

A Depth Restoration Occlusionless Temporal Dataset

Daniel Rotman and Guy Gilboa

Electrical Engineering Department, Technion - Israel Institute of Technology
Technion City, Haifa 32000, Israel

{drotman@tx, guy.gilboa@ee}.technion.ac.il

Abstract

Depth restoration, the task of correcting depth noise and artifacts, has recently risen in popularity due to the increase in commodity depth cameras. When assessing the quality of existing methods, most researchers resort to the popular Middlebury dataset; however, this dataset was not created for depth enhancement, and therefore lacks the option of comparing genuine low-quality depth images with their high-quality, ground-truth counterparts. To address this shortcoming, we present the Depth Restoration Occlusionless Temporal (DROT) dataset. This dataset offers real depth sensor input coupled with registered pixel-to-pixel color images, and the ground-truth depth to which we wish to compare. Our dataset includes not only Kinect 1 and Kinect 2 data, but also an Intel R200 sensor intended for integration into hand-held devices. Beyond this, we present a new temporal depth-restoration method. Utilizing multiple frames, we create a number of possibilities for an initial degraded depth map, which allows us to arrive at a more educated decision when refining depth images. Evaluating this method with our dataset shows significant benefits, particularly for overcoming real sensor-noise artifacts.

1. Introduction

1.1. Depth Sensor Evolution

With the introduction of depth sensors a door was opened to a wide variety of possibilities. Research into the abilities of these sensors blossomed in many fields, including: computer vision, robotics, natural user interaction and many more. At first, the depth sensors were lacking in multiple ways, particularly in spatial and temporal resolution. Methods were developed to attempt to overcome these disadvantages by using a coupled color camera to guide and assist the depth sensor. With the introduction of the Kinect 1 and 2 cameras, which included high resolution and 30 FPS depth sensor, it seemed that these methods were no longer necessary.

Recently with the advance of technology to the realm of hand-held devices, we have begun to re-examine depth frame enhancement's usefulness in our mobile oriented world. As part of this renewed interest, Intel released the hand-held R200 depth sensor (among others). As a result, the Kinect camera, a power consuming and not mobile-friendly device, may possibly become outdated even though the R200 depth quality is much below that of the Kinect. We find ourselves again in need of software oriented solutions to improve a highly noisy depth sensor that is pushing the limits of our technological abilities.

In this work we present a dataset for evaluation of RGB guided depth restoration, and offer a new depth restoration method which specifically overcomes modern depth camera limitations.

1.2. RGB Based Restoration

There are many existing methods for improving Depth data based on coupled RGB cameras.

One of the building blocks that lies at the center of a number of depth correction algorithms is the Bilateral filter, presented in [16] This filter is used to improve depth using the color information by weighting neighbor depth values according to spatial distance and color similarity. In [3] a blending function is added to take color into consideration on edges while ignoring it on smooth areas so as to avoid depth shadowing.

In [12] the authors propose depth evolution using PDE and anisotropic diffusion. Pixels that contain depth data are considered heat sources and their information is propagated to the areas in the image with less known depth values. Defining the correction of the image as a progress in time, they perform the diffusion procedure using diffusion conductance described as a Gaussian weighted color similarity function.

In [4] a Markov Random Field (MRF) formulation is proposed, formulating a quadratic smoothness term and neighbor weighting factors based on color similarity. They use a two layer MRF to represent depth images, where one layer corresponds to the color, and the other to the depth.

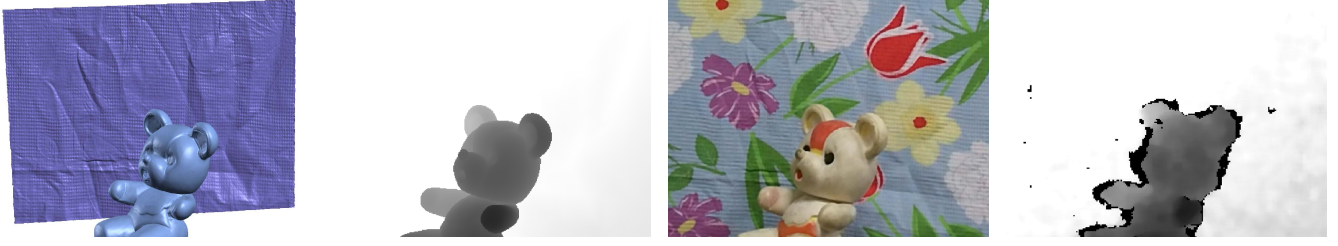


Figure 1. DROT - a dataset for depth restoration

They define a depth measurement potential to weigh against known depth values, a depth smoothness prior for the continuity of object depth values, and weighting factors to avoid depth smoothness over color image borders. In [8] the authors introduce a second order smoothness prior to overcome a tendency to estimate depth as fronto-parallel surfaces.

Unlike the above methods, that can create a linear combination of depth values, in [15] the authors propose to use a weighted histogram:

$$H(p, d) = \sum_{q \in S} g_1(I_p - I_q) g_2(p - q) g_3(d - D_q) \quad (1)$$

Where g_1, g_2 and g_3 are similarity metrics often chosen as gaussian functions. Here, a histogram is weighted according to the similarity of the intensity (or color) and proximity and closeness of depth value (g_3 can be taken as a binary function to achieve a “true” histogram). Then the depth values are chosen as:

$$\tilde{D}_p = \arg \max_d \{H(p, d)\} \quad (2)$$

This enforces that depth remains consistent with the depth values that were observed.

In [18], the authors propose a method to overcome low depth temporal resolution. They use high accuracy optical flow [2] estimated between color images and then apply it to the depth image to create an initial estimation. The optical flow is a flow field w that describes the observed motion between frames $I_p(t) \approx I_{p+w}(t+1)$. A reasonable assumption given that I and its corresponding D are aligned pixel-to-pixel, is that:

$$D_p(t) \approx D_{p+w}(t+1) \quad (3)$$

Which in practice can be seen as warping one depth image to look like another given the RGB optical flow.

1.3. Camera Calibration

In the process of creating our dataset, we utilized a number of fundamental and classic models and calculations, which due to space restraints we mention only briefly: The pinhole camera model [9] can be used to move between

point clouds and depth images once the parameters were learned from a camera calibration process [24]. Multiple cameras require the use of transformation between coordinate systems, which can be calculated with 3D point correspondences and the Kabsch algorithm [10]

2. Previous Datasets

Today, the availability of a suitable dataset for depth restoration is very limited. Most researchers must resort to testing the quality of their methods on the Middlebury dataset [19]. This dataset was originally intended as a stereo ground truth dataset and so, for every setting, it consists of two RGB images and the matching “true” depth. The obvious problem is that in order to run restoration algorithms, one must have both high quality depth images for ground truth, and low quality depth images to run the algorithm on - which are lacking in the Middlebury dataset.

Currently, researchers take the Middlebury depth, degrade it, and then attempt to reconstruct the original depth. In most studies, the method used to corrupt the depth is to simply downsample the given depth, thus creating a very limited model of corrupted depth, while others attempt to contaminate the depth data in ways that aren’t at all indicative of how depth sensor data looks. For example in [14] the authors added zero mean white Gaussian noise and randomly invalidated 13% of the pixels. Another example is [13] where the authors synthetically create holes in the depth images which are not related to the structure in the image. In these cases The resulting “degraded” depth images look nothing like a depth image captured by a real sensor.

Some more advanced methods attempted to model sensor artifacts and noise. For example, Time of Flight (ToF) depth sensors are known to suffer from measurement noise and [7] while the Kinect 1 camera suffers from invalid depth values mainly on edges (Fig. 3). Given this, the authors in [23] proposed a number of ways to emulate depth degradation. They modeled ToF depth noise by under-sampling and inserting random signal dependent noise, and they modeled Kinect 1 sensor input by manually editing depth edge pixels to have a value of “invalid” (Fig. 2).

Besides the drawback that all the manually edited depth

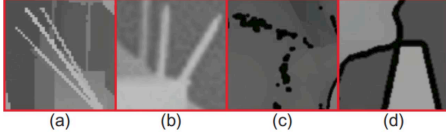


Figure 2. Depth degradation techniques (taken from [23]). (a) under-sampling, (b) undersampling with signal-dependant noise, (c) random missing, and (d) structural missing.

images mentioned above don’t really reproduce depth sensor data, none of the methods take into account problematic spatial precision on edges which is very common in Kinect 1 sensors due to its structured light technology [1]. These artifacts can appear with edges which should be smooth and instead show up as wavy. As can be seen in Fig. 3, the depth from Kinect 1 is often inconsistent with object boundaries - this can be a major pitfall for methods that rely on the fact that no background-colored pixels will have a valid depth value of the foreground.

All of this doesn’t even begin to describe the artifacts that appear in the R200 RealSense depth camera. Besides serious structural inconsistencies, there are also outlier depth values (Fig. 2), something not seen in either Kinect 1 or 2 and not emulated in any of the manually degraded depth examples. The goal of this work is to create a baseline for depth restoration algorithms to allow for testing on real state-of-the-art sensor data, thus raising the performance of these algorithms to the point where they are practical in real situations.

In a previous study, the authors improved upon the prevailing mode and used a real sensor dataset with highly accurate depth ground truth [6]. While the dataset was an improvement on datasets used in other studies, it was limited in scope, containing just three frames with many occlusions and lacking a temporal element.

We proposed to develop an encompassing dataset that could perform as a thorough qualitative tool. Our main contributions regarding this dataset are as follows:

We provide over 100 frames of depth with ground truth data for each depth sensor, Kinect 1, 2 and RealSense. We create an instance as seen from the RGB sensor and also as seen from the IR sensor. This allows for both upsampling



Figure 3. From left to right, Kinect 1, 2 and RS. The Kinect 1 sensor features invalid (black) depth values, and crooked edges. The Kinect 2 has false intermediate depth values (on the right side of the object). The RS shows depth artifacts with erroneous values (outlier white pixel on left bottom).



Figure 4. System rig. Sensors from top to bottom: Kinect 2 sensor, R200 RS sensor, 3.1 M DAVID CAM and Kinect 1 sensor.

depth images (when aligning low resolution depth to high resolution color) and also a comparative result on the original depth images without undergoing transformations. Our dataset contains no occlusions using our complete 3D scene capturing scheme. We integrate a temporal element into our dataset allowing the evaluation of methods which use temporal information to improve depth data as we present in 4.

3. Dataset Creation

We aimed to create a dataset with real sensor data and depth ground truth. Our objectives were that the dataset have temporal changes, but at the same time contain no occlusions in the ground truth data. In order to obtain this, we needed a complete understanding of the surrounding 3D structure of the given scene. We used a SLS-2 David Scanner equipped with a high-quality K132 projector, a monochrome 3.1 M DAVID CAM camera and glass calibration panels - a system which uses Structured Light to capture 3D data [20]. Because the David Scanner captures 3D data from one viewpoint only, it was necessary to scan an object multiple times to obtain complete structural information. The David scanner software uses a 3D point cloud registration algorithm to combine multiple point clouds to receive a final complete representation of an object in 3D space [21, 22].

The main problem with creating a temporal dataset was that the David scanner can take multiple seconds to capture a single depth viewpoint. In addition, we didn’t want the structured light emitting from the projector to contaminate the color that we wished to capture with our depth sensors. The conclusion we came to was to create temporal videos using stop-motion, i.e. filming a video one frame at a time and moving the objects manually between frames to create the illusion of motion. Our methodology was as follows:

1. Perform a complete scan of all the objects in the scene receiving a complete 3D representation of each object.
2. Scan the background of the scene.
3. For each frame, capture depth sensor data and scanner data.



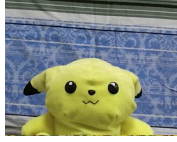


	Vid 1	Vid 2	Vid 3	Vid 4	Vid 5
# Frames	30	21	30	20	11
Motion	Turn, Parallel	Parallel	Turn	Parallel	Turn, Parallel, Perpendicular
Preview					

Table 1. Information on the videos in the dataset.

- Using partial information from scanner data from each frame, register the complete 3D scans to their correct location using a point cloud registration algorithm.
- Transform scanner data from scanner space to each depth sensor space.
- Render ground truth depth frames from depth sensor scans.
- Create Depth-RGB aligned pairs for every depth sensor.

In the following subsections we will expand on some of these points.

3.1. Scanning Complete 3D Objects

Each stop motion video must include at least one object undergoing motion during the length of the video. In order to have the ground truth depth information of the object from any view that the depth sensors observed it, we needed a complete 3D scan of the object.

We scanned the objects in advance using the David scanner. By capturing multiple viewpoints, we were able to reconstruct the 3D object surface using the 3D scanners software which includes 3D point cloud registration and mesh surface estimation. The registration process used color information as well as structure, and so in some difficult cases, color markers were used to help guide the registration process.

For every object approximately twenty scans from different angles were used. In some cases of black and other dark colors appearing on the objects, a number of different shutter speeds were used in order to be able to effectively capture all types of materials present. The process resulted in a complete 3D point cloud with surface mesh information in the scanner coordinate system.

3.2. Create 3D Scene

The scanner captured a high quality but partial view of the object and background. Using this incomplete information, we were able to then register the previously scanned complete objects into the current scanned frame, resulting in the complete 3D structure of the whole scene. The high precision of the scanner allowed for the objects to be inserted into their correct location and then be rendered from

the different viewpoints of the depth cameras. Registration of the 3D objects was performed using the point cloud registration algorithm provided by the David scanner software. Though the details of the registration algorithms are not available to the public, they are most likely based on the creator’s relevant publications [21, 22], and though we treated the process as a black box in this work, the quality of the scans attainable by the David scanner are verified to be up to 0.1% of the scan size - in our case, approximately 2mm.

3.3. Aligning Scans to Sensors

Alignment of the scans to the viewpoint of each camera proves to be a very difficult task, especially when very high accuracy is desired. In order to limit errors to the magnitude of a pixel, we chose to perform the alignment of the scans to the depth sensors in an end-to-end fashion to reduce accumulative errors. Point to point correspondences were chosen between the texture map of the 3D scan and the viewpoint of the depth cameras using a classic checkerboard pattern. What differentiated this process from regular stereo camera calibration was that instead of 2D points corresponding to 2D points, we had 3D points (due to the texture map correspondences to the 3D mesh) to 2D points matches. We chose not to use the depth from the sensors in this process since it is less reliable especially when aiming for pixel precision.

We optimized over the six degrees of freedom of rotation and translation, using the camera parameters obtained prior in laboratory conditions. Using the Nelder-Mead Simplex optimization, we arrived at the optimized parameters which gave a minimum to the squared distance between projected points.

3.4. Rendering Ground Truth Depth

Given the 3D point cloud with mesh information in image frame coordinates we rendered the triangles of the mesh using the depth values of the vertices to arrive at a depth image as seen from the depth sensor’s view point.

3.5. Creating Depth-RGB Aligned Pairs

An issue rose when projecting low resolution depth onto high resolution color, in the form of sparse depth points as

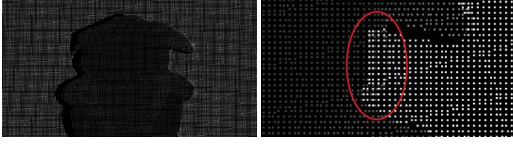


Figure 5. Projection from low to high resolution causes a depth values to be known only on a grid. In red circle: object and background transformed to the same area.

can be seen in Fig. 5. These points are neither on a grid, nor in fact are they projected in a monotonic way. What this means is that to the same area, pixels from different depths can be copied due to the stereo nature of the rig - these are areas which the depth sensor has knowledge about what is behind the object due to the slight difference in location. Because of the grid-like shape of the projection we find ourselves with inter-spliced depth values as can be seen in (Fig. 5 right).

What we do in this case, is to use the method described in [11] where the authors describe an operator to discern which points are visible in 3D point clouds. Though our problem seems different at first, we can use this operator to filter out the pixels which are sparsely copied behind the object.

3.6. Dataset Precision

After completing the process above, we wished to test the precision of our created dataset.

Since the dataset is intended for RGB based depth restoration, it is crucial that the RGB and depth ground truth be aligned perfectly. We believe that assessing the exact precision of the dataset is sadly not possible, since we didn't have information about the real object locations - this information was exactly what we were trying to calculate in our dataset. The calibration processes returned a mean reprojection error of approximately 0.3 pixels, and the optimization returned a reprojection error of approximately 1 pixel, but this doesn't indicate the precision of the whole process. We resorted to testing the precision by eye in the following way: using the ground truth data created, it is easy to segment objects from background by simple depth thresholding. In this way we can see visually the distance between object border in ground truth relative to RGB. We performed the same test on the Middlebury dataset in order to have comparative results.

As can be seen in Fig. 6 there exists a small error on the borders of objects which is comparable to the error that can be seen in the Middlebury dataset. We estimate this error to be within 1-1.5 pixels dependent on object distance.

4. Temporal Depth Restoration

We now present our method that utilizes the temporal aspect to perform improved depth restoration.



Figure 6. Dataset Precision. Foreground vs. Background, our dataset (top) and Middlebury dataset (bottom) [19]. On object borders we can observe the alignment error of the dataset. The border of green pixels observed around the cones in the background of the Middlebury dataset seems of the same order as the error seen in our dataset.

4.1. Formulation

Given a series of matching RGB and Depth frames, we denote the weighted mode depth restoration method from 2 as a function:

$$\tilde{D}(t) = WM(I(t), D(t)) \quad (4)$$

Where the t parameter describes the temporal placement in the series. Additionally, we describe the optical flow based depth warping in 3 as a function as well:

$$D(t) = OFW(I(t), I(t+1), D(t+1)) \quad (5)$$

We can now substitute 5 into 4, omitting the RGB inputs to simplify the mathematical notation:

$$\tilde{D}(t) = WM(OFW(D(t+1))) \quad (6)$$

This means using a “warped depth” as an initial guess for the weighted mode process. For more generality we can extend this idea to an arbitrary time difference:

$$\tilde{D}^n(t) = WM(OFW(D(t+n))) \quad (7)$$

Where the superscript represents the weighted mode result when using as an initial guess the optical flow warped depth from n frames after. Note that \tilde{D}^0 is the original weighted mode method.

4.2. Multiple Frame Consideration

Instead of creating multiple $\tilde{D}(t)$ estimations and attempting to combine them, we integrate the idea into the weighted mode method. We will describe a multiple weighted mode as:

$$\tilde{D}^{0 \dots n}(t) = WM(D(t), D(t+1), \dots, D(t+n)) \quad (8)$$

Where there's an implied use of the *OFW* function when using a future depth frame. The superscript $0 \dots n$ implies that the current depth frame and the next n depth frames were used in the multiple weighted mode. This multiple weighted mode can be performed by applying an addition to the original histogram definition in 1. We wish to fill the depth value histogram with pixels not only from one depth image but from a number of images corresponding to optical flow warped estimations of the current depth frame. In practice we correct the definition by:

$$H_G(p, d) = \sum_{i=0}^n \sum_{q \in S} g_1(I_p - I_q) g_2(p - q) g_3(d - D_q(t + i)) \quad (9)$$

Where $D_q(t + i)$ represents the depth value at pixel q in optical flow warped estimation i .

4.3. Motivation

When dealing with structured light technology, there's the issue of problematic spatial precision on edges as discussed above. In this case, sometimes an additional guess for the initial depth can be closer to the truth than the original. We take for example the fingers of a hand: Kinect 1 is not reliable at separating the fingers in the depth frame at a reasonable distance. Instead a single mass is seen which has flickering pixels (flickering temporally between background, foreground and invalid) where the depth has high uncertainty due to the small object size of the fingers. The histogram nature of the weighted mode means that taking multiple initial guess into account will allow better statistical interpretation of each pixel and so a higher probability of correctness.

It's important to note that in the formula of 9 only $I(t)$ is considered, and not other RGB frames either warped or not. The reason for this is that in RGB based depth restoration we treat the RGB data as highly reliable and attempt to correct according to it. It would be unwise to attempt to correct depth when basing ourselves on altered RGB frames.

However, it is possible to take the RGB into consideration. When using warped depth frames we can warp the color as well, and then in the weighted mode histogram we can weigh the depth value according to the error between warped and regular RGB, adding to 9 an additional weight:

$$g_4(I_q(t) - OFW(I_q(t + i))) \quad (10)$$

This would allow for detection of when the assumptions of optical flow have failed (occlusions, discontinuities) and we can weigh the depth pixels accordingly as less reliable.

Beyond this, using multiple depth frames would allow us to upgrade the original idea in [18] to use scene flow instead

of optical flow. Scene flow [17], similar to optical flow, is the perceived motion, but instead of in two dimensions, the scene flow field is three dimensional. This would allow for correct depth estimation under motion perpendicular to the depth camera.

And finally, since the use of optical flow warp is for an initial guess only, then we don't need the motion to be too precise. If we downsample the images before calculating optical flow we can cut run times drastically. Upsampling the resulting flow field gives results which many times are just as useful as the full resolution flow field in regard to depth optical flow warp. Experimental results showed that run time could be cut by an order of magnitude just down-sampling by 1/4, and the difference in depth estimation was negligible.

5. Evaluation

5.1. Error Metrics

The commonly used metrics for depth correction are Root Mean Square Error (RMSE) and Mean Absolute Distance (MAD).

These metrics are inherently flawed when attempting to score depth values. One of the most important attributes of a depth image is the structural build of object versus background, and it is imperative that depth values aren't contorted from a correct value just to satisfy the error metric.

For example, as can be seen in Fig 8, the bilateral filter results in gradual transitions between object and background. This type of behavior can be counter-productive for applications performing object recognition and background segmentation. On the other hand, the weighted mode and multiple weighted mode return sharp transitions. And as can be seen in Table 3, the bilateral filter received a better MAD and RMSE score, though most applications would prefer definitive depth values like in the weighted mode.

Therefore we use the Tukey biweight estimation metric [5]:

$$Tukey(A, B) = \min \left\{ 1 - \left(1 - (|A - B| / d_{max})^2 \right)^3, 1 \right\} \quad (11)$$

calculated for each pixel A versus its corresponding ground truth B . The metric for an entire image is the average pixel Tukey distances.

This weight penalizes depth differences up to a point, d_{max} , and then gives a relatively constant error. In this case the depth algorithms will be penalized for a medium error the same as for a large error. This makes sense, since as far as structural depth is concerned, there's no difference - in both cases the object was missed, but with the Tukey metric, algorithms aren't tempted to stretch depth values.

In our experimentation $d_{max} = 5cm$ since we decide that any error larger than this value is equally erroneous.

5.2. Experimentation

Our database includes five videos of varying lengths with a sum of 112 frames detailed in Table 1. For all the frames, we captured information using three depth sensors: Kinect 1, 2 and RealSense R200, and for each camera, we created an instance aligned to the RGB sensor and to the depth sensor. Altogether, taking into account that an instance of the dataset includes a matching RGB, depth and ground truth, we had 672 of these instances. The dataset is freely downloadable at: <http://visl.technion.ac.il/databases/drot2016/>

We ran a number of basic depth upsampling and filtering methods to demonstrate the usability of this dataset. In Table 3 are the results of these methods and our $\tilde{D}^{0...4}$ when applied on the first frame of Vid 2 (we opted to use Vid 2 for integrity, since the parameters of our method were learned from the first frame in Vid 1), and in Fig 8 are the visual results. For analysis of the entire performance, we present in Table 2 the percentages in which a specific algorithm outperformed the rest. The table is divided to analyze the performance on different slices of the dataset. We show the out-performance percentage for: all instances, divided by video, divided by camera view and divided by error metric.

To evaluate the temporal depth restoration method, g_1 and g_2 were taken as Gaussian functions, g_3 was taken as a boolean function returning a value of 1 if its argument was 0, and g_4 was not used. We optimized the parameters of the algorithm using a Genetic Algorithm optimization process on the RMSE of the first frame of Vid 1 of the Kinect 1 camera from the RGB view, which was chosen arbitrarily being the first frame of our dataset. Once the weighted mode pa-

		[16]	[12]	[4]	[8]	[15]	Ours $\tilde{D}^{4...0}$
All		23%	2%	13%	7%	9%	46%
Video	1	56%	6%	0%	0%	6%	33%
	2	28%	6%	11%	11%	0%	44%
	3	6%	0%	6%	6%	11%	72%
	4	28%	0%	33%	11%	6%	22%
	5	0%	0%	17%	6%	22%	56%
View	K1 RGB	40%	0%	13%	7%	0%	40%
	K1 IR	27%	7%	7%	7%	0%	53%
	K2 RGB	20%	0%	40%	0%	13%	27%
	K2 IR	47%	7%	13%	27%	7%	0%
	RS RGB	0%	0%	7%	0%	13%	80%
	RS IR	7%	0%	0%	0%	20%	73%
Metric	MAD	23%	0%	20%	7%	3%	47%
	RMSE	23%	0%	20%	10%	3%	43%
	Tukey	23%	7%	0%	3%	20%	47%

Table 2. Percentage of instances in which the method outperformed the others, per category. From top to bottom, over all the instances, divided by video, divided by depth sensor and view-point, and divided by metric. The highest percentage in each row is emphasized in bold.

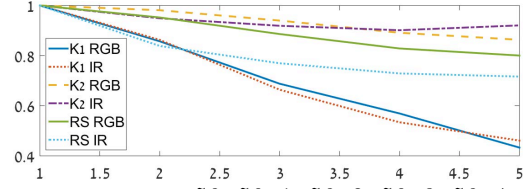


Figure 7. The RMSE of $\tilde{D}^0, \tilde{D}^{0...1}, \tilde{D}^{0...2}, \tilde{D}^{0...3}, \tilde{D}^{0...4}$ marked by the x axis from 1 to 5. Though the values are discrete, they were connected by a continuous line in order to portray the rate of improvement as the number of depth frames used increases. Each series of values was divided by the highest value so as to be able to place all graphs on one set of axis (hence, the largest value is 1).

rameters were set, we used them for all experimentation of the multiple weighted mode.

In addition, we performed some evaluations of the complete set of estimations $\tilde{D}^{0...1}, \tilde{D}^{0...2}, \tilde{D}^{0...3}, \tilde{D}^{0...4}$. The comparisons to the normal weighted mode \tilde{D}^0 are given in Table 4 and Figure 7.

6. Results and Discussion

The dataset provided offers the ability to perform comprehensive evaluation on depth restoration methods. Indeed the dataset is so large, that researchers who wish to limit themselves to a specific camera or even view are still left with a substantial amount of data to use as evaluation or learning.

From observing the results of depth restoration methods, we come to a number of conclusions: Table 2 shows that our multiple weighted mode outperforms the other methods on nearly all slices of the dataset. Besides good metric results, the multiple weighted mode is also superior in that it overcomes depth artifacts such as can be seen in the RealSense rows in Fig. 8. A clear improvement can be seen in Fig. 7 as more depth frames are considered. The multiple weighted mode takes advantage of the statistical nature of the histogram of the weighted mode by collecting multiple initial estimations.

The low results on the Kinect 2 are possibly due to ToF “fake depth” values on object intersections. ToF cameras on object borders many times return a value which is an average (for example, halfway between the object and the background). Since this is prominent in a relatively large area on the border, the weighted mode may consider it the true depth and propagate it inwards. A possible way to overcome this is to invalidate depth values on a section of object borders, or weighting them with a low confidence.

7. Conclusions

We have presented a novel large-scale depth restoration dataset, which allows comprehensive evaluation on depth restoration methods using real depth sensor data and high-

	[16]	[12]	[4]	[8]	[15]	Ours $\tilde{D}^{4...0}$
K1 RGB	144.7 , 176.24 , 0.86	178.91, 205.5, 0.91	153.15, 179.92, 0.89	170.6, 198.2, 0.9	225.6, 253.75, 0.87	228.35, 257.19, 0.84
K1 IR	120.2 , 160.61 , 0.71	135.56, 181.8, 0.7	122.9, 163.38, 0.71	131.23, 175.13, 0.71	154.92, 205.28, 0.69	145.3, 198.01, 0.65
K2 RGB	79.5, 106.6, 0.64	77.66, 103.68, 0.63	77.07 , 100.81 , 0.64	77.55, 102.49, 0.63	88.6, 125.6, 0.62	83.55, 124.47, 0.56
K2 IR	76.13, 100.83, 0.64	71.3, 94.46, 0.62	72.4, 94.89, 0.62	71.1 , 91.29 , 0.63	107.41, 147.44, 0.64	124.35, 169.04, 0.66
RS RGB	67.84, 195.57, 0.51	63.16, 125.23, 0.52	60.19, 97.55, 0.52	63.43, 127.24, 0.52	64.88, 173.88, 0.51	51.76 , 71.91 , 0.47
RS IR	62.02, 111.72, 0.51	67.24, 163.09, 0.51	65.42, 140.17, 0.51	64.03, 144.55, 0.52	56.93, 77.93, 0.51	50.3 , 67.61 , 0.47

Table 3. Metric results on a single frame; in each square the MAD, RMSE and Tukey metrics respectively are presented. The best performance for each view per metric is emphasized in bold.

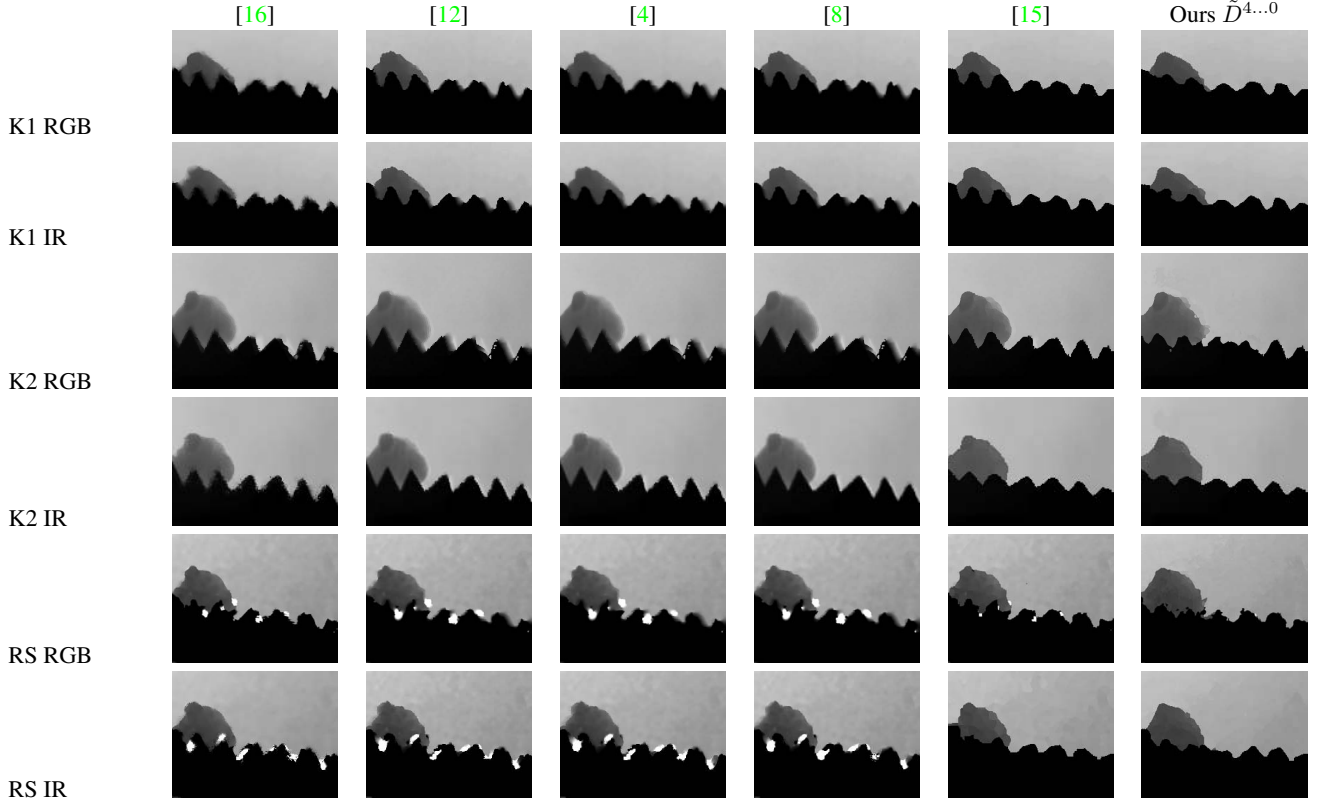


Figure 8. Visual results

	\tilde{D}^0	$\tilde{D}^{0...1}$	$\tilde{D}^{0...2}$	$\tilde{D}^{0...3}$	$\tilde{D}^{0...4}$
K1 RGB	243.74, 281.95, 0.87	204.74, 252.75, 0.79	159.13, 213.27, 0.73	133.15, 184.53, 0.72	115.11 , 162.25 , 0.71
K1 IR	120.25, 182.76, 0.56	106.18, 167.22, 0.54	93.92, 151.04, 0.53	87.41, 140.02, 0.53	82.77 , 130.87 , 0.54
K2 RGB	145.36, 195.22, 0.7	110.54, 158.56, 0.63	89.7, 132.01, 0.62	81.78, 116.64, 0.64	79.27 , 110.94 , 0.65
K2 IR	173.03, 221.61, 0.76	158.42, 209.63, 0.72	145.71, 196.69, 0.7	139.2, 187.56, 0.7	125.27 , 171.65 , 0.69
RS RGB	51.58, 71.26, 0.49	50.92, 68.86, 0.49	49.68, 63.6, 0.48	48.68, 60.03, 0.48	48.03 , 57.6 , 0.49
RS IR	54.08, 79.82, 0.5	52.93 , 72.34 , 0.5	62.08, 225.39, 0.49	60.3, 223.05, 0.49	59.89, 228.91, 0.49

Table 4. Metric results on a single frame. The best performance for each view per metric is emphasized in bold.

quality ground truth. The different technologies of depth sensors in the dataset give a representative picture of true depth noise and artifacts. The temporal aspect of the dataset makes it as general as possible, and the lack of occlusions promise a true evaluation for depth restoration. Our new temporal depth restoration method using the multiple weighted mode has been shown to perform outstandingly. The method especially performs well on the noise riddled RealSense camera which will most likely become the standard for depth cameras as technology moves toward mobile devices.

Acknowledgments

We acknowledge support by the Magnet program of the OCS, Israel Ministry of Economy, in the framework of Omek Consortium.

References

- [1] M. R. Andersen, T. Jensen, P. Lisouski, A. K. Mortensen, M. K. Hansen, T. Gregersen, and P. Ahrendt. Kinect depth sensor evaluation for computer vision applications. *Technical Report Electronics and Computer Engineering*, 1(6), 2015. 2
- [2] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *European Conference on Computer Vision (ECCV)*, volume 3024 of *Lecture Notes in Computer Science*, pages 25–36. Springer, May 2004. 1.2
- [3] D. Chan, H. Buisman, C. Theobalt, and S. Thrun. A noise-aware filter for real-time depth upsampling. In *Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications-M2SFA2 2008*, 2008. 1.2
- [4] J. Diebel and S. Thrun. An application of markov random fields to range sensing. In *NIPS*, volume 5, pages 291–298, 2005. 1.2, 5.2, 5.2, 5.2
- [5] G. Drozdov, Y. Shapiro, and G. Gilboa. Robust recovery of heavily degraded depth measurements. In *International Conference on 3D Vision (3DV)*. IEEE, 2016 in press. 5.1
- [6] D. Ferstl, C. Reinbacher, R. Ranftl, M. R  ther, and H. Bischof. Image guided depth upsampling using anisotropic total generalized variation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 993–1000, 2013. 2
- [7] S. B. Gokturk, H. Yalcin, and C. Bamji. A time-of-flight depth sensor-system description, issues and solutions. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW'04. Conference on*, pages 35–35. IEEE, 2004. 2
- [8] A. Harrison and P. Newman. Image and sparse laser fusion for dense scene reconstruction. In *Field and Service Robotics*, pages 219–228. Springer, 2010. 1.2, 5.2, 5.2, 5.2
- [9] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 1.3
- [10] W. Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976. 1.3
- [11] S. Katz, A. Tal, and R. Basri. Direct visibility of point sets. *ACM Transactions on Graphics (TOG)*, 26(3):24, 2007. 3.5
- [12] J. Liu and X. Gong. Guided depth enhancement via anisotropic diffusion. In *Advances in Multimedia Information Processing-PCM 2013*, pages 408–417. Springer, 2013. 1.2, 5.2, 5.2, 5.2
- [13] S. Liu, Y. Wang, J. Wang, H. Wang, J. Zhang, and C. Pan. Kinect depth restoration via energy minimization with tv 21 regularization. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 724–724. IEEE, 2013. 2
- [14] S. Lu, X. Ren, and F. Liu. Depth enhancement via low-rank matrix completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3390–3397, 2014. 2
- [15] D. Min, J. Lu, and M. N. Do. Depth video enhancement based on weighted mode filtering. *Image Processing, IEEE Transactions on*, 21(3):1176–1190, 2012. 1.2, 5.2, 5.2, 5.2
- [16] G. Petschnigg, R. Szeliski, M. Agrawala, M. Cohen, H. Hoppe, and K. Toyama. Digital photography with flash and no-flash image pairs. In *ACM transactions on graphics (TOG)*, volume 23, pages 664–672. ACM, 2004. 1.2, 5.2, 5.2, 5.2
- [17] J. Quiroga, T. Brox, F. Devernay, and J. Crowley. Dense semi-rigid scene flow estimation from rgbd images. In *European Conference on Computer Vision*, pages 567–582. Springer, 2014. 4.3
- [18] D. Rotman, O. Cohen, and G. Gilboa. Frame rate reduction of depth cameras by rgb-based depth prediction. In *International Conference on the Science of Electrical Engineering (ICSEE)*. IEEE, 2016 in press. 1.2, 4.3
- [19] D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–195. IEEE, 2003. 2, 6
- [20] S. Winkelbach, S. Molkenstruck, and F. M. Wahl. Low-cost laser range scanner and fast surface registration approach. In *Pattern Recognition*, pages 718–728. Springer, 2006. 3
- [21] S. Winkelbach, M. Rilk, C. Sch  nfelder, and F. M. Wahl. Fast random sample matching of 3d fragments. In *Pattern Recognition*, pages 129–136. Springer, 2004. 3, 3.2
- [22] S. Winkelbach and F. M. Wahl. Pairwise matching of 3d fragments using cluster trees. *International Journal of Computer Vision*, 78(1):1–13, 2008. 3, 3.2
- [23] J. Yang, X. Ye, K. Li, C. Hou, and Y. Wang. Color-guided depth recovery from rgb-d data using an adaptive autoregressive model. *Image Processing, IEEE Transactions on*, 23(8):3443–3458, 2014. 2, 2
- [24] Z. Zhang. A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(11):1330–1334, 2000. 1.3