Depth Extraction from Video Using Non-parametric Sampling

Kevin Karsch
University of Illinois at Urbana-Champaign

Ce Liu
Microsoft Research New England

Sing Bing Kang
Microsoft Research

Abstract. We describe a technique that automatically generates plausible depth maps from videos using non-parametric depth sampling. We demonstrate our technique in cases where past methods fail (non-translating cameras and dynamic scenes). Our technique is applicable to single images as well as videos. For videos, we use local motion cues to improve the inferred depth maps, while optical flow is used to ensure temporal depth consistency. For training and evaluation, we use a Kinect-based system to collect a large dataset containing stereoscopic videos with known depths. We show that our depth estimation technique outperforms the state-of-the-art on benchmark databases. Our technique can be used to automatically convert a monoscopic video into stereo for 3D visualization, and we demonstrate this through a variety of visually pleasing results for indoor and outdoor scenes, including results from the feature film *Charade*.

1 Introduction

While many reconstruction techniques for extracting depth from video sequences exist, they typically assume moving cameras and static scenes. They do not work for dynamic scenes or for stationary, purely rotating, or strictly variable focal length sequences. There are some exceptions, e.g., [1], which can handle some moving objects, but they still require camera motion to induce parallax and allow depth estimation.

In this paper, we present a novel solution to generate depth maps from ordinary 2D videos; our solution also applies to single images. This technique is applicable to arbitrary videos, and works in cases where conventional depth recovery methods fail (static/rotating camera; change in focal length; dynamic scenes). Our primary contribution is the use of a non-parametric "depth transfer" approach for inferring temporally consistent depth maps without imposing requirements on the video (Sec 2 and 3), including a method for improving the depth estimates of moving objects (Sec 3.1). In addition, we introduce a new, ground truth stereo RGBD (RGB+depth) video dataset[1] (Sec 4). We also describe how we synthesize stereo videos from ordinary 2D videos using the results of our technique (Sec 5).

Several reconstruction methods for single images of real, unknown scenes have also been proposed. One of the first methods, introduced by Hoiem et al. [2],

[1] Our dataset is publicly available at http://kevinkarsch.com/depthtransfer

Fig. 1: Our technique takes a video sequence (*top row*) and automatically estimates per-pixel depth (*bottom row*). Our method does not require any cues from motion parallax or static scene elements; these videos were captured using a stationary camera with multiple moving objects.

created convincing reconstructions of outdoor images by assuming an image could be broken into a few planar surfaces; similarly, Delage et al. developed a Bayesian framework for reconstructing indoor scenes [3]. Saxena et al. devised a supervised learning strategy for predicting depth from a single image [4], which was further improved to create realistic reconstructions for general scenes [5], and efficient learning strategies have since been proposed [6]. Better depth estimates have been achieved by incorporating semantic labels [7], or more sophisticated models [8]. Repetitive structures can also be used for stereo reconstruction from a single image [9]. Single-image shape from shading is also possible for known (a priori) object classes [10,11]. We not only focus on depth from a single image, but also present a framework for using temporal information for enhanced and time-coherent depth when multiple frames are available.

One application of these methods is 2D-to-3D conversion; however, many existing techniques require user interaction to refine depth and stereo estimates [12,13,14]. An exception is the contemporaneous work of Konrad et al., which uses non-parametric depth sampling to automatically convert monocular images into stereoscopic images [15]. Our technique extends their inference procedure, and works for videos as well.

Liu et al. [16] showed that arbitrary scenes can be semantically labelled through non-parametric learning. Given an unlabeled input image and a database with known per-pixel labels (e.g., sky, car, tree, window), their method works by transferring the labels from the database to the input image based on SIFT features. We build on this work by transferring depth instead of semantic labels. Furthermore, we show that this "transfer" approach can be applied in a continuous optimization framework (Sec 2.1), whereas their method used a discrete optimization approach (MRFs).

2 Non-parametric depth estimation

We leverage recent work on non-parametric learning [16], which avoids explicitly defining a parametric model and requires fewer assumptions as in past methods (e.g., [4,5,7]). This approach also scales better with respect to the training data size, requiring virtually no training time. Our technique imposes no requirements on the video, such as motion parallax or sequence length, and can even be applied

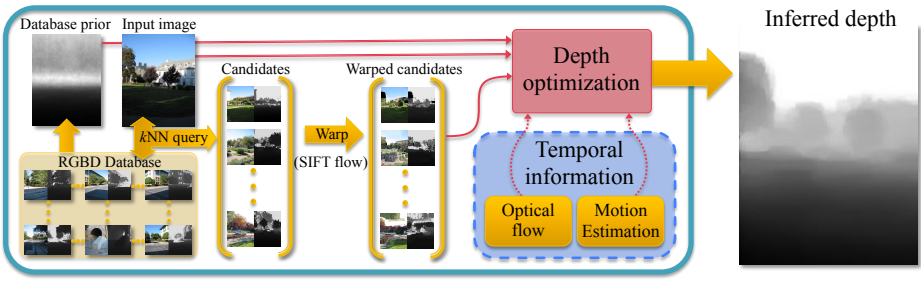

Fig. 2: Our pipeline for estimating depth. Given an input image, we find matching candidates in our database, and warp the candidates to match the structure of the input image. We then use a global optimization procedure to interpolate the warped candidates (Eq. 1), producing per-pixel depth estimates for the input image. With temporal information (e.g. extracted from a video), our algorithm can achieve more accurate, temporally coherent depth.

to a single image. We first describe our depth estimation technique as it applies to single images below, and in Sec 3 we discuss novel additions that allow for improved depth estimation in videos.

2.1 Depth estimation via continuous label transfer

Our depth transfer approach, outlined in Fig 2, has three stages. First, given a database RGBD images, we find candidate images in the database that are "similar" to the input image in RGB space. Then, a warping procedure (SIFT Flow [17]) is applied to the candidate images and depths to align them with the input. Finally, an optimization procedure is used to interpolate and smooth the warped candidate depth values; this results in the inferred depth.

Our core idea is that scenes with similar semantics should have roughly similar depth distributions when densely aligned. In other words, images of semantically alike scenes are expected to have similar depth values in regions with similar appearance. Of course, not all of these estimates will be correct, which is why we find several candidate images and refine and interpolate these estimates using a global optimization technique that considers factors other than just absolute depth values.

RGBD database. Our system requires a database of RGBD images and/or videos. We have collected our own RGBD video dataset, as described in Sec 4; a few already exist online, though they are for single images only.[2]

Candidate matching and warping. Given a database and an input image, we compute high-level image features (we use GIST [18] and optical flow features, see the supplementary file) for each image or frame of video in the database as well as the input image. We then select the top K ($= 7$ in our work) matching

[2] Examples: Make3D range image dataset (http://make3d.cs.cornell.edu/data.html), B3DO dataset (http://kinectdata.com/), NYU depth datasets (http://cs.nyu.edu/~silberman/datasets/), RGB-D dataset (http://www.cs.washington.edu/rgbd-dataset/), and our own (http://kevinkarsch.com/depthtransfer).

Fig. 3: SIFT flow warping. (a) SIFT features are calculated and matched in a one-to-many fashion, which defines ψ. (b) ψ is applied to achieve dense scene alignment.

frames from the database, but ensure that each video in the database contributes no more than one matching frame. This forces matching images to be from differing viewpoints, allowing for greater variety among matches. We call these matching images *candidate images*, and their corresponding depths *candidate depths*.

Because the candidate images match the input closely in feature space, it is expected that the overall semantics of the scene are roughly similar. We also make the critical assumption that the distribution of depth is comparable among the input and candidates. However, we want pixel-to-pixel correspondences between the input and all candidates, as to limit the search space when inferring depth from the candidates.

We achieve this pixel-to-pixel correspondence through SIFT flow [17], which matches per-pixel SIFT features to estimate dense scene alignment. Using SIFT flow, we estimate warping functions $\psi_i, i \in \{1, \ldots, K\}$ for each candidate image; this process is illustrated in Fig 3. These warping functions map pixel locations from a given candidate's domain to pixel locations in the input's domain. The warping functions can be one-to-many.

Depth optimization. Each warped candidate depth is deemed to be a rough approximation of the input's depth map. Unfortunately, such candidate depths may still contain inaccuracies and are often not spatially smooth. Instead, we generate the most likely depth map by considering all of the warped candidates, optimizing with spatial regularization in mind.

Let $\mathbf{L}$ be the input image and $\mathbf{D}$ the depth map we wish to infer. We minimize

$$-\log(P(\mathbf{D}|\mathbf{L})) = E(\mathbf{D}) = \sum_{i \in \text{pixels}} E_\text{t}(\mathbf{D}_i) + \alpha E_\text{s}(\mathbf{D}_i) + \beta E_\text{p}(\mathbf{D}_i) + \log(Z), \quad (1)$$

where Z is the normalization constant of the probability, and α and β are parameters ($\alpha = 10, \beta = 0.5$). For a single image, our objective contains three terms: data (E_t), spatial smoothness (E_s), and database prior (E_p).

The data term measures how close the inferred depth map $\mathbf{D}$ is to each of the warped candidate depths, $\psi_j(C^{(j)})$. This distance measure is defined by ϕ, a robust error norm (we use an approximation to the $L1$ norm, $\phi(x) = \sqrt{x^2 + \epsilon}$, with $\epsilon = 10^{-4}$). We define the data term as

$$E_\text{t}(\mathbf{D}_i) = \sum_{j=1}^{K} w_i^{(j)} \Big[\phi(\mathbf{D}_i - \psi_j(C_i^{(j)})) + \gamma \big[\phi(\nabla_x \mathbf{D}_i - \psi_j(\nabla_x C_i^{(j)})) + \phi(\nabla_y \mathbf{D}_i - \psi_j(\nabla_y C_i^{(j)})) \big] \Big], \quad (2)$$

where $w_i^{(j)}$ is a confidence measure of the accuracy of the j^{th} candidate's warped depth at pixel i (more details in the supplementary file), and K (= 7) is the total number of candidates. We measure not only absolute differences, but also relative depth changes, i.e., depth gradients. The latter terms of Eq 2 enforce similarity among candidate depth gradients and inferred depth gradients, weighted by γ (= 10).

We encourage spatial smoothness, but more so in regions where the input image has small intensity gradients:

$$E_{\mathrm{s}}(\mathbf{D}_i) = s_{x,i}\phi(\nabla_x \mathbf{D}_i) + s_{y,i}\phi(\nabla_y \mathbf{D}_i). \tag{3}$$

The depth gradients along x and y ($\nabla_x \mathbf{D}, \nabla_y \mathbf{D}$) are modulated by soft thresholds (sigmoidal functions) of image gradients in the same directions ($\nabla_x \mathbf{L}, \nabla_y \mathbf{L}$), namely, $s_{x,i} = (1 + e^{(||\nabla_x \mathbf{L}_i|| - 0.05)/.01})^{-1}$ and $s_{y,i} = (1 + e^{(||\nabla_y \mathbf{L}_i|| - 0.05)/.01})^{-1}$; see the supplemental file for further explanation.

We also incorporate assumptions from our database that will guide the inference when pixels have little or no influence from other terms (due to weights w and s):

$$E_{\mathrm{p}}(\mathbf{D}_i) = \phi(\mathbf{D}_i - \mathcal{P}_i). \tag{4}$$

We compute the prior, $\mathcal{P}$, by averaging all depth images in our database.

This is an unconstrained, non-linear optimization, and we use iteratively reweighted least squares to minimize our objective function (details in the supplementary file).

3 Improving depth estimation for videos

Generating depth maps frame-by-frame without incorporating temporal information often leads to temporal discontinuities; past methods that ensure temporal coherence rely on a translating camera and static scene objects. Here, we present a framework that improves depth estimates and enforces temporal coherence for *arbitrary video sequences*. That is, our algorithm is suitable for videos with moving scene objects and rotating/zooming views where conventional SFM and stereo techniques fail. (Here, we assume that zooming induces little or no parallax.)

Our idea is to incorporate temporal information through additional terms in the optimization that ensure (a) depth estimates are consistent over time and (b) that moving objects have depth similar to their contact point with the ground. Each frame is processed the same as in the single image case (candidate matching and warping), except that now we employ a global optimization (described below) that infers depth for the entire sequence at once, incorporating temporal information from all frames in the video. Fig 4 illustrates the importance of these additional terms in our optimization.

We formulate the objective for handling video by adding two terms to the single-image objective:

$$E_{\mathrm{video}}(\mathbf{D}) = E(\mathbf{D}) + \sum_{i \in \mathrm{pixels}} \nu E_{\mathrm{c}}(\mathbf{D}_i) + \eta E_{\mathrm{m}}(\mathbf{D}_i), \tag{5}$$

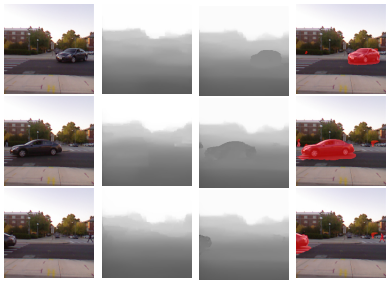

Fig. 4: Importance of temporal information. Left: input frames. Mid-left: predicted depth without temporal information. Note that the car is practically ignored here. Mid-right: predicted depth with temporal information, with the depth of the moving car recovered. Right: detected moving object.

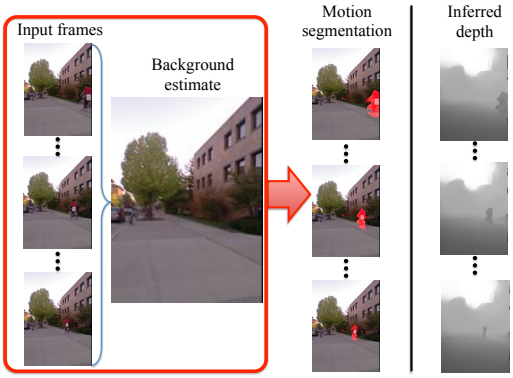

Fig. 5: We apply median filtering on the stabilized images to extract the background image. A pixel is deemed to be in motion if there is sufficient relative difference from the background image.

where E_{c} encourages temporal coherence while E_{m} uses motion cues to improve the depth of moving objects. The weights ν and η balance the relative influence of each term ($\nu = 100, \eta = 5$).

We model temporal coherence first by computing per-pixel optical flow for each pair of consecutive frames in the video (using Liu's publicly available code [19]). We define the *flow difference*, ∇_{flow}, as a linear operator which returns the change in the flow across two corresponding pixels, and model the coherence term as

$$E_{\mathrm{c}}(\mathbf{D}_i) = s_{t,i}\phi(\nabla_{flow}\mathbf{D}_i). \tag{6}$$

We weight each term by a measure of flow confidence, $s_{t,i} = (1 + e^{-(||\nabla_{flow}\mathbf{L}_i||-0.05)/.01})^{-1}$, which intuitively is a soft threshold on the reprojection error. Minimizing the weighted flow differences has the effect of temporally smoothing inferred depth in regions where optical flow estimates are accurate.

To handle motion, we detect moving objects in the video (Sec 3.1) and constrain their depth such that these objects touch the floor. Let m be the binary motion segmentation mask and $\mathcal{M}$ the depth in which connected components in the segmentation mask contact the floor. We define the motion term as

$$E_{\mathrm{m}}(\mathbf{D}_i) = m_i\phi(\mathbf{D}_i - \mathcal{M}_i). \tag{7}$$

3.1 Motion segmentation

Differentiating moving and stationary objects in the scene can be a useful cue when estimating depth. Here we describe our algorithm for detecting objects in motion in non-translating movies (i.e., static, rotational, and variable focal length videos).

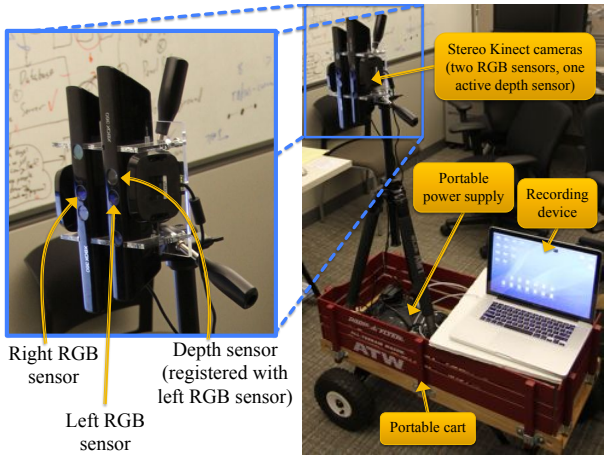

Fig. 6: Our stereo-RGBD collection rig consists of two side-by-side Microsoft Kinects. The rig is mobile through the use of an uninterruptible power supply, laptop, and rolling mount.

First, to account for dynamic exposure changes throughout the video, we find the image with the lowest overall intensity in the sequence and perform histogram equalization on all other frames in the video. We use this image as to not propagate spurious noise found in brighter images. Next, we use RANSAC on point correspondences to compute the dominant camera motion (modeled using homography) to align neighboring frames in the video. Median filtering is then used on the stabilized images to extract the background B (ideally, without all the moving objects).

In our method, the likelihood of a pixel being in motion depends on how different it is from the background, weighted by the optical flow magnitude which is computed between stabilized frames (rather than between the original frames). We use relative differencing (relative to background pixels) to reduce reliance on absolute intensity values, and then threshold to produce a mask:

$$m_{i,k} = \begin{cases} 1 \text{ if } ||flow_{i,k}|| \frac{||W_{i,k} - B_i||^2}{B_i} > \tau \\ 0 \text{ otherwise,} \end{cases} \quad (8)$$

where $\tau = 0.01$ is the threshold, and $W_{i,k}$ is the k^{th} pixel of the i^{th} stabilized frame (i.e., warped according to the homography that aligns W with B). This produces a motion estimate in the background's coordinate system, so we apply the corresponding inverse homography to each warped frame to find the motion relative to each frame of the video. This segmentation mask is used (as in Eq 7) to improve depth estimates for moving objects in our optimization. Fig 5 illustrates this technique.

4 Dataset

In order to train and test our technique on video input, we collected a dataset of over 200 stereoscopic video sequences with corresponding depth values for

Fig. 7: Results obtained on four different sequences sequences captured with a rotating camera and/or variable focal length. In each 3×3 block of images, we show the input frames (*top*), inferred depth (*middle*) and inferred 3D anaglyph (*bottom*). Notice that the sequences are time-coherent and that moving objects are not ignored.

one of the two stereoscopic views. These sequences come from four different buildings in two cities and contain substantial scene variation (e.g., hallways, rooms, foyers). Each clip is filmed with camera viewpoints that are either static or slowly rotated. Our dataset primarily contains one or more persons walking through a scene, sitting or socializing.

To capture data, we use two side-by-side, vertically mounted Microsoft Kinects shown in Fig 6 (positioned about 5cm apart). We collected the color images from both Kinects and only the depth map from the left Kinect.

We also collected outdoor data with our stereo device. However, because the Kinect cannot produce depth maps outdoors due to IR interference from the sunlight, we could not use these sequences for training. We did not apply stereo to extract ground truth depth because of reliability issues. We did, however, use this data for testing and evaluation purposes.

5 Application: Automatic stereoscopic view synthesis

In recent years, 3D[3] videos have become increasingly popular. Many feature films are now available in 3D, and increasingly more personal photography devices are

[3] The presentation of stereoscopic (left+right) video to convey the sense of depth.

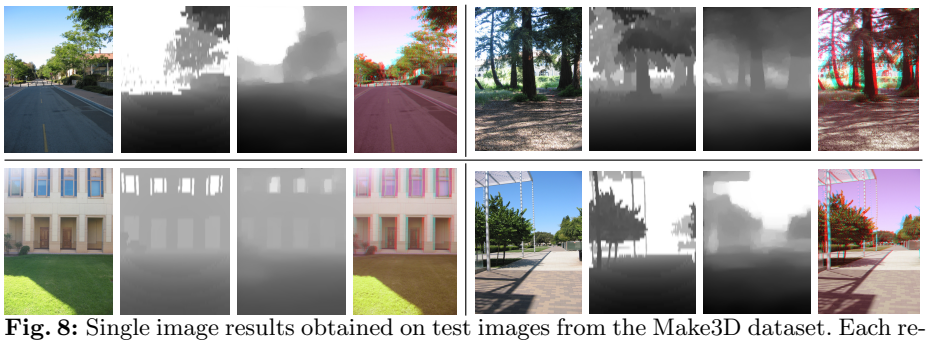

Fig. 8: Single image results obtained on test images from the Make3D dataset. Each result contains the following four images (from left to right): original photograph, ground truth depth from the dataset, our inferred depth, and our synthesized anaglyph image. The depth images are shown in log scale. Darker pixels indicate nearby objects (black is roughly 1m away) and lighter pixels indicate objects farther away (white is roughly 80m away). Each pair of ground truth and inferred depths are displayed at the same scale.

now equipped with stereo capabilities (from point-and-shoots to attachments for video cameras and SLRs). Distributing user-generated content is also becoming easier. Youtube has recently incorporated 3D viewing and uploading features, and many software packages have utilities for handling and viewing 3D file formats, e.g., Fujifilm's FinePixViewer.

As 3D movies and 3D viewing technology become more widespread, it is desirable to have techniques that can convert legacy 2D movies to 3D in an efficient and inexpensive way. Currently, the movie industry uses expensive solutions that tend to be manual-intensive. For example, it was reported that the cost of converting (at most) 20 minutes of footage for the movie "Superman Returns" was $10 million[4].

Our technique can be used to automatically generate the depth maps necessary to produce the stereoscopic video (by warping each input frame using its corresponding depth map). To avoid generating holes at disocclusions in the view synthesis step, we adapt and extend Wang et al.'s technique [20]. They developed a method that takes as input a single image and per-pixel disparity values, and intelligently warps the input image based on the disparity such that highly salient regions remain unmodified. Their method was applied only to single images; we extend this method to handle video sequences as well. Details of our view synthesis technique are given in the supplementary file.

6 Results

We use the Make3D range image dataset to evaluate our single image depth estimation technique. Of the 534 images in the Make3D dataset, we use 400 for testing and 134 for training (the same as was done before, e.g., [4,5,8,7]). We report error for three common metrics in Table 1. Denoting $\mathbf{D}$ as estimated depth and $\mathbf{D}^*$ as ground truth depth, we compute *relative* (**rel**) *error*

[4] See http://en.wikipedia.org/wiki/Superman_Returns.

Method	rel	$\log_{10}$	RMS
Depth MRF [4]	0.530	0.198	16.7
Make3D [5]	0.370	0.187	N/A
Feedback Cascades [8]	N/A	N/A	15.2
Semantic Labels [7]	0.375	**0.148**	N/A
Depth Transfer (ours)	**0.361**	**0.148**	**15.1**

Table 1: Comparison of depth estimation errors on the Make3D range image dataset. Using our single image technique, our method achieves state of the art results in each metric (**rel** is relative error, **RMS** is root mean squared error; details in text).

Dataset	rel	$\log_{10}$	RMS	PSNR
Building 1†	0.196	0.082	8.271	15.6
Building 2	0.394	0.135	11.7	15.6
Building 3	0.325	0.159	15.0	15.0
Building 4	0.251	0.136	15.3	16.4
Outdoors††	N/A	N/A	N/A	15.2
All	0.291	0.128	12.6	15.6

Table 2: Error averaged over our stereo-RGBD dataset. †Building used for training (results for Building 1 trained using a hold-one-out scheme). ††No ground truth depth available.

$\frac{|\mathbf{D}-\mathbf{D}^*|}{\mathbf{D}^*}$, $\mathbf{log}_{10}$ *error* $|\log_{10}(\mathbf{D}) - \log_{10}(\mathbf{D}^*)|$, and *root mean squared* (**RMS**) *error* $\sqrt{\sum_{i=1}^{N}(\mathbf{D}_i - \mathbf{D}_i^*)^2/N}$. Error measures are averaged over all pixels/images in the test set. Our estimated depth maps are computed at 345×460 pixels (maintaining the aspect ratio of the Make3D dataset input images).

Our method is as good as or better than the state-of-the-art for each metric. Note that previously no method achieved state-of-the-art results in more than one metric. Our estimated depth is good enough to generate compelling 3D images, and representative results are shown in Fig 8. In some cases, our method even produces better depth estimates than the ground truth (with low resolution and sensor errors). Thin structures (e.g., trees and pillars) are usually recovered well; however, fine structures are occasionally missed due to spatial regularization (such as the poles in the bottom-right image of Fig 8).

As further evaluation, a qualitative comparison between our technique and the publicly available version of Make3D is shown in Fig 9. Unlike Make3D, our technique is able to extract the depth of the runner throughout the sequence.

Our technique works well for videos of many types scenes and video types (Figs 1, 4, 5, 7, 10; more examples are in the supplemental files). We use the dataset we collected in Sec 4 to validate our method for indoor scenes and videos (we know of no other existing methods/datasets to compare to). This dataset contains ground truth depth and stereo images for four different buildings (referred to as Buildings 1, 2, 3, and 4), and to demonstrate generalization, *we only use data from Building 1 for training*. We still generate results for Building 1 by holding each particular example out of the training set during inference.

We show quantitative results in Table 2 and qualitative results in Fig 10. We calculate error using the same metrics as in our single image experiments, and to make these results comparable with Table 1, we globally rescale the ground truth and inferred depths to match the range of the Make3D database (roughly 1-81m). As expected, the results from Building 1 are the best, but our method still achieves reasonable errors for the other buildings as well.

Fig 10 shows a result from each building in our dataset (top left is Building 1). As the quantitative results suggest, our algorithm performs very well for this building. In the remaining examples, we show results of videos captured in the other three buildings, all which contain vastly different colors, surfaces and

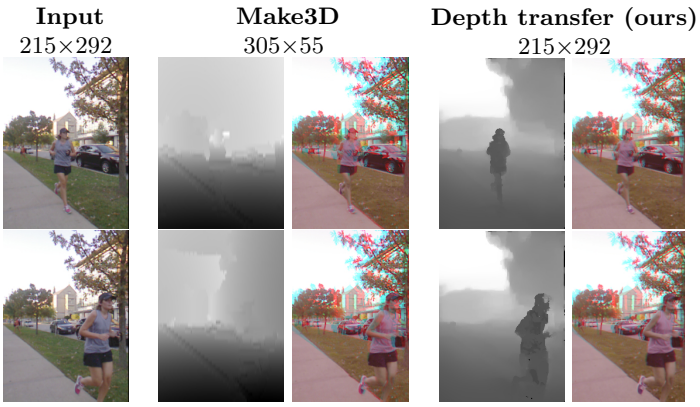

Fig. 9: Comparison between our technique and the publicly available version of Make3D (http://make3d.cs.cornell.edu/code.html). Make3D depth inference is trained to produce depths of resolution 55 × 305 (bilinearly resampled for visualization), and we show results of our algorithm at the input native resolution. The anaglyph images are produced using the technique in Sec 5. Depths displayed at same scale.

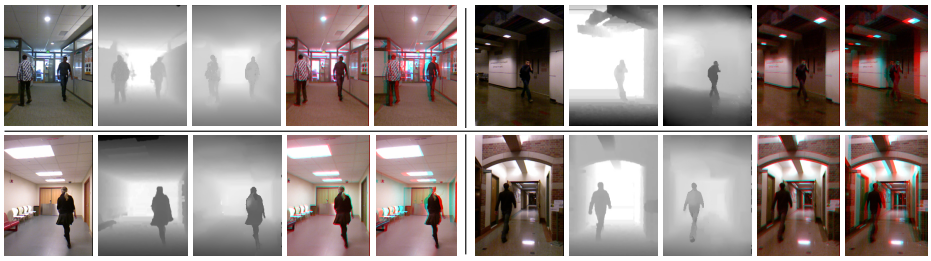

Fig. 10: Video results obtained on test images for each building in our stereo-RGBD dataset (buildings 1-4, from left to right and top to bottom). For each result (from left to right): original photograph, ground truth depth, our inferred depth, ground truth anaglyph image, and our synthesized anaglyph image. Because the ground truth 3D images were recorded with a fixed interocular distance (roughly 5cm), we cannot control the amount of "pop-out," and the 3D effect is subtle. However, this is a parameter we can set using our automatic approach to achieve a desired effect, which allows for an enhanced 3D experience. Note also that our algorithm can handle multiple moving objects (*top*). Additional results are shown in the supplemental files.

Fig. 11: Several clips from the feature film *Charade*. Each result contains (from top to bottom): the original frames, estimated depth, and estimated anaglyph automatically generated by our algorithm. Some imperfections in depth are conveniently masked in the 3D image due to textureless or less salient regions.

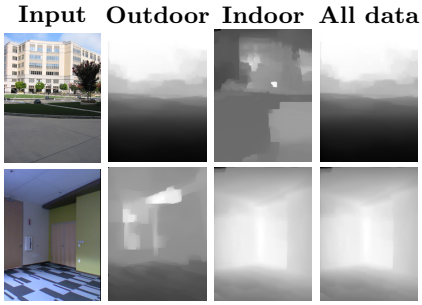

Fig. 12: Effect of using different training data for indoor and outdoor images. While the results are best if the proper dataset is used, we also get good results even if we combine all the datasets.

Fig. 13: Example failure cases. Top row: thin or floating objects (pole and basketball) are ignored. Bottom row: input image is too different from training set.

structures from the Building 1. Notice that even for these images our algorithm works well, as evidenced by the estimated depth and 3D images.

Our algorithm also does not require video training data to produce video results. We can make use of static RGBD images (e.g. Make3D dataset) to train our algorithm for video input, and we show several outdoor video results in Figs 1, 4, 5 (more in the supplemental files). Even with static data from another location, our algorithm is usually able to infer accurate depth and stereo views.

Since we collected ground truth stereo images in our dataset, we also compare our synthesized right view with actual right view. We use peak signal-to-noise ratio (**PSNR**) to measure the quality of the reconstructed views, as shown in Table 2. We could not acquire depth outdoors, and we use this metric to compare our outdoor and indoor results.

We also demonstrate that our algorithm may be suitable for feature films in Fig 11. More diverse quantities of training are required to achieve commercial-quality conversion; however, even with a small amount of data, we can generate plausible depth maps and create convincing 3D sequences automatically.

Recently, Youtube has released an automatic 2D-to-3D conversion tool, and we compared our method to theirs on several test sequences. Empirically, we noticed that the Youtube results have a much more subtle 3D effect. Both results are available online at `http://kevinkarsch.com/depthtransfer`.

Our algorithm takes roughly one minute per 640×480 frame (on average) using a parallel implementation on a quad-core 3.2GHz processor.

7 Discussion

Our results show that our algorithm works for a large variety of indoor and outdoor sequences using a practical amount of training data. Note that our algorithm works for arbitrary videos, not just those with no parallax. However, videos with arbitrary camera motion and static scenes are best handled with techniques such as [1]. In Fig 12, we show that our algorithm requires some similar data in order to produce decent results (i.e., training with outdoor images

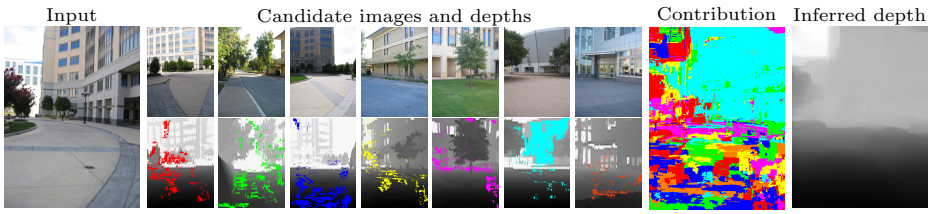

Fig. 14: Candidate contribution for depth estimation. For an input image (*left*), we find top-matching candidate RGBD images (*middle*), and infer depth (*right*) for the input using our technique. The contribution image is color-coded to show the sources; red pixels indicate that the left-most candidate influenced the inferred depth image the most, orange indicates contribution from the right-most candidate, etc.

for an indoor query is likely to fail). However, our algorithm can robustly handle large amounts of depth data with little degradation of output quality. The only issue is that more data requires more comparisons in the candidate search.

This robustness is likely due to the features we use when determining candidate images as well as the design of our objective function. In Fig 14, we show an example query image, the candidates retrieved by our algorithm, and their contribution to the inferred depth. By matching GIST features, we detect candidates that contain features consistent with the query image, such as building facades, sky, shrubbery, and similar horizon location. Notice that the depth of the building facade in the input comes mostly from another similarly oriented building facade (teal), and the ground plane and shrubbery depth come almost solely from other candidates' ground and tree depths.

In some cases, our motion segmentation misses or falsely identifies moving pixels. This can result in inaccurate depth and 3D estimation, although our spatio-temporal regularization (Eqs. 3, 6) helps to overcome this. Our algorithm also assumes that moving objects contact the ground, and thus may fail for airborne objects (see Fig 13).

Due to the serial nature of our method (depth estimation followed by view synthesis), our method is prone to propagating errors through the stages. For example, if an error is made during depth estimation, the result may be visually implausible. It would be ideal to use knowledge of how the synthesized views should look in order to correct issues in depth.

8 Concluding Remarks

We have demonstrated a fully automatic technique to estimate depths for videos. Our method is applicable in cases where other methods fail, such as those based on motion parallax and structure from motion, and works even for single images and dynamics scenes. Our depth estimation technique is novel in that we use a non-parametric approach, which gives qualitatively good results, and our single-image algorithm quantitatively outperforms existing methods. Using our technique, we also show how we can generate stereoscopic videos for 3D viewing from conventional 2D videos. Specifically, we show how to generate time-coherent, vi-

sually pleasing stereo sequences using our inferred depth maps. Our method is suitable as a good starting point for converting legacy 2D feature films into 3D.

Acknowledgement

We would like to thank Tom Blank for his critical help in creating our dual-Kinect data collection system.

References

1. Zhang, G., Jia, J., Hua, W., Bao, H.: Robust bilayer segmentation and motion/depth estimation with a handheld camera. IEEE TPAMI (2011) 603–617
2. Hoiem, D., Efros, A., Hebert, M.: Automatic photo pop-up. In: ACM SIGGRAPH. (2005)
3. Delage, E., Lee, H., Ng, A.: A dynamic Bayesian network model for autonomous 3D reconstruction from a single indoor image. In: CVPR. (2006)
4. Saxena, A., Chung, S., Ng, A.: Learning depth from single monocular images. NIPS (2005)
5. Saxena, A., Sun, M., Ng, A.: Make3D: Learning 3D scene structure from a single still image. IEEE TPAMI **31** (2009) 824–840
6. Batra, D., Saxena, A.: Learning the right model: Efficient max-margin learning in laplacian crfs. In: CVPR. (2012)
7. Liu, B., Gould, S., Koller, D.: Single image depth estimation from predicted semantic labels. In: CVPR. (2010)
8. Li, C., Kowdle, A., Saxena, A., Chen, T.: Towards holistic scene understanding: Feedback enabled cascaded classification models. NIPS (2010)
9. Wu, C., Frahm, J.M., Pollefeys, M.: Repetition-based dense single-view reconstruction. CVPR (2011)
10. Han, F., Zhu, S.C.: Bayesian reconstruction of 3D shapes and scenes from a single image. In: IEEE HLK. (2003)
11. Hassner, T., Basri, R.: Example based 3D reconstruction from single 2D images. CVPR Workshop on Beyond Patches (2006)
12. Guttmann, M., Wolf, L., Cohen-Or, D.: Semi-automatic stereo extraction from video footage. In: ICCV'09. (2009) 136–142
13. Ward, B., Kang, S.B., Bennett, E.P.: Depth Director: A system for adding depth to movies. IEEE Comput. Graph. Appl. **31** (2011) 36–48
14. Liao, M., Gao, J., Yang, R., Gong, M.: Video stereolization: Combining motion analysis with user interaction. IEEE Transactions on Visualization and Computer Graphics **18** (2012) 1079–1088
15. Konrad, J., Wang, M., Ishwar, P.: 2d-to-3d image conversion by learning depth from examples. In: 3DCINE. (2012)
16. Liu, C., Yuen, J., Torralba, A.: Nonparametric scene parsing: Label transfer via dense scene alignment. CVPR (2009)
17. Liu, C., Yuen, J., Torralba, A.: SIFT Flow: Dense correspondence across scenes and its applications. IEEE TPAMI **33** (2011) 978–994
18. Oliva, A., Torralba, A.: Modeling the shape of the scene: A holistic representation of the spatial envelope. IJCV **42** (2001) 145–175
19. Liu, C.: Beyond pixels: Exploring new representations and applications for motion analysis. PhD thesis, MIT (2009)
20. Wang, O., Lang, M., Frei, M., Hornung, A., Smolic, A., Gross, M.: StereoBrush: Interactive 2D to 3D conversion using discontinuous warps. In: SBIM. (2011)