3D Aware Correction and Completion of Depth Maps in Piecewise Planar Scenes

Ali K. Thabet[1], Jean Lahoud[1], Daniel Asmar[2], Bernard Ghanem[1]

[1]King Abdullah University of Science and Technology (KAUST), Saudi Arabia
[2]American University of Beirut (AUB), Lebanon

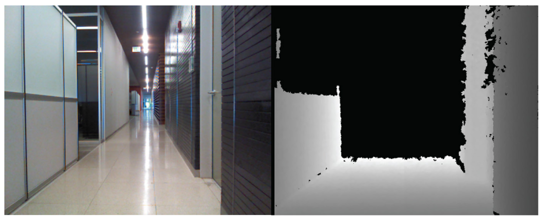
Abstract. RGB-D sensors are popular in the computer vision community, especially for problems of scene understanding, semantic scene labeling, and segmentation. However, most of these methods depend on *reliable* input depth measurements, while discarding unreliable ones. This paper studies how reliable depth values can be used to *correct* the unreliable ones, and how to *complete* (or extend) the available depth data beyond the raw measurements of the sensor (i.e. infer depth at pixels with unknown depth values), given a prior model on the 3D scene. We consider piecewise planar environments in this paper, since many indoor scenes with man-made objects can be modeled as such. We propose a framework that uses the RGB-D sensor's noise profile to adaptively and robustly fit plane segments (e.g. floor and ceiling) and iteratively complete the depth map, when possible. Depth completion is formulated as a discrete labeling problem (MRF) with hard constraints and solved efficiently using graph cuts. To regularize this problem, we exploit 3D and appearance cues that encourage pixels to take on depth values that will be compatible in 3D to the piecewise planar assumption. Extensive experiments, on a new large-scale and challenging dataset, show that our approach results in more accurate depth maps (with 20% more depth values) than those recorded by the RGB-D sensor. Additional experiments on the NYUv2 dataset show that our method generates more 3D aware depth. These generated depth maps can also be used to improve the performance of a state-of-the-art RGB-D SLAM method.

1 Introduction

Active RGB-D sensors are capable of directly capturing 3D structure from a scene, thus, avoiding the difficult task of inferring this information from the scene's appearance alone. Sensors like the MS Kinect are popular in computer vision applications, since they are relatively cheap, accessible, and well supported/maintained by manufacturers and third-party developers. There is a large amount of recent work that exploits RGB-D data to generalize and shed light on traditional vision problems, including semantic scene labeling [22], segmentation [13], scene understanding [1, 19, 11], object detection [20], visual SLAM (Simultaneous Localization And Mapping) [26, 18], among others.

Most of the RGB-D methods above rely on the fact that the input depth image is largely comprised of pixels with *reliable* (accurate) depth measurements.

This assumption might not always be valid, as most sensors only provide reliable measurements up to a maximum distance d_{rel} (usually of 3-4.5 meters). Given this constraint, most RGB-D based methods either record images where the scene is within the d_{rel} range away from the sensor or disregard all pixels whose depth values exceed d_{rel}. This unreliability in depth measurement arises due to several reasons, including limitations in the depth sensor itself (e.g., the projected IR signal of a Kinect decays with squared distance and it is unable to image objects that are too close or too far) and the nature of the scene itself (e.g., IR absorbing or reflective surfaces). In Figure 1 (*left*), we show a sample RGB-D image pair taken in an indoor office scene, with all the unreliable and unknown depth values set to zero (black pixels). Clearly, not all pixels are assigned depth values. Disregarding unreliable pixels limits the range and impedes the generality of methods using RGB-D data. This issue is even more pressing when general indoor scenes are considered, such as in open areas, office spaces, reasonably sized rooms, museums, etc. In these cases, much of the scene is at a distance larger than the maximum reliable depth value d_{rel}. We show empirical evidence of this in Figure 1 (*right*). We histogram the average percentage of pixels discarded in an image because they were deemed unreliable ($d_{rel} = 4.5$ meters) for a typical recording of a walk-through inside an office/lab environment. For this recording, roughly 60% of pixels in each Kinect depth map are deemed unreliable. Discarding these pixels limits subsequent processing or learning modules, including RGB-D semantic labeling, scene understanding, and visual SLAM methods.



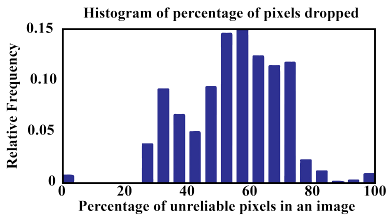

Histogram of percentage of pixels dropped
Relative Frequency
0.15
0.10
0.05
0
0
20
40
60
80
100
Percentage of unreliable pixels in an image

Fig. 1. Left: A sample RGB-D image pair. This is a single frame out of roughly 1000 that were recorded while walking through an indoor piecewise planar scene (*office corridor*). **Right:** The percentage of pixels per image of this walk-through that are unreliable because their depth values are above a certain threshold d_{rel} or are unknown.

Interestingly, many of these unreliable pixels are projections of 3D scene points belonging to *objects* (e.g., floor, ceiling, walls, cabinets, etc.) that have extensions within the reliable range of depth pixels. So, it is important to study how reliable depth values can be used to judge and even *correct* unreliable ones, when a particular prior model is assumed on the 3D scene. In fact, this could also be used to *complete* (or extend) the depth values beyond the raw measurements of the sensor itself (i.e., infer depth at pixels with unknown depth). This paper studies this problem for 3D piecewise planar scenes and proposes a novel framework that makes use of both appearance and 3D cues from a *single* RGB-D image pair to correct and complete the depth image. This framework can enable subsequent scene processing and understanding even if the initial RGB-D data is deemed unreliable. Here, we define a piecewise planar scene to be one that comprises a set of intersecting planes (realistically plane segments). Piecewise planar scenes are valid descriptions of many indoor scenes containing man-made objects (e.g., building interiors, offices, homes, museums, etc.). This scene prior has been used in previous work for the purpose of stereo vision or 3D reconstruction [9, 10], semantic labelling [22], and scene understanding from a single RGB-D image [19, 8] or a single RGB image [16, 14, 15].

2 Related Work

This paper addresses the problem of correcting unreliable depth values and inferring unknown ones in RGB-D data of piecewise planar scenes. In data of this kind, large groups of contiguous pixels have either unreliable or unknown depth values. The most related work in the literature that address a similar problem (depth enhancement) is categorized as: **(1)** hole-filling (depth inpainting) methods or **(2)** depth upsampling methods.

Methods of category **(1)** attempt to infer depth at pixels that are not assigned depth values. These pixels tend to be projections of parts of surfaces that are IR absorbing, reflective, or too close to the RGB-D sensor. These pixels are small in number and tend not to cluster in the same portion of a depth image. Most hole-filling methods interpolate (or propagate) unknown depths from depths in pixel neighborhoods. To this end, joint bilateral filtering (JBF) has been extensively used to fill holes in depth images, especially due to its relatively attractive runtime [21, 4]. Moreover, colorization techniques are also used to fill in unknown depth [23], as done in compiling the popular NYUv2 dataset [25]. In [6], the problem is formulated as a continuous Markov Random Field (MRF), where the latent variables are the depth values of all pixels, the unary (data) term is dependent on the known depth values, and the binary term encourages similar looking pixels in a local neighborhood to have similar depth values. In [5], a foreground depth model is assumed to be available and depth layers are inferred using a discrete MRF. Many other methods of this category exist (e.g., [27]); however, they all suffer from the same drawback that makes them infeasible and inappropriate for the problem of depth correction and completion tackled in this paper. Hole-filling (depth inpainting) methods assume that there is a strong correlation between depth discontinuities and image edges and that pixels with similar appearance have similar 3D geometry. In general, this assumption is a useful cue for interpolation but it does not always hold, especially in scenes where large portions of a depth image need to be filled.

Methods of category **(2)** attempt to generate a high resolution depth image from a low-resolution one and (usually) a registered high resolution RGB image. The low-resolution depth image is assumed to be complete and comprised of reliable depth values. Since a plethora of such methods abound in the literature, we mention a representative few here. In fact, JBF and MRF labeling are common techniques employed by methods of this category [24, 21, 6]. Similar to the problem of hole-filling, local assumptions of depth smoothness (except at color discontinuities) are made to propagate depth values from the low resolution image to a local neighborhood of unknown values in the higher resolution image. These assumptions break down in the case of large contiguous holes, which makes upsampling methods unsuitable for the problem addressed in this paper.

All previous methods apply local priors to depth values in RGB-D data, but they tend not to take into account the global 3D structure of the scene for unreliable depth correction and completion. These methods do *not* ensure that the processed depth maps lead to a 3D point cloud that has a compatible 3D structure (i.e. its structure does not lead to any 3D contradictions or impossibilities). Assuming a piecewise planar scene allows us to regularize the completion and correction process globally as well as locally. This regularization disallows certain depth assignments that lead to a 3D structure that is not compatible or does not respect the underlying planar assumption.

Contributions: They are three fold. **(i)** Unlike other depth enhancement methods, this paper addresses the problem of correction and completion of unreliable and unknown depth values in a single RGB-D image pair. To the best of our knowledge, this is the first work that makes use of local *and* global priors on the overall 3D scene to enable 3D aware depth prediction and correction. Here, the global prior on the scene is piecewise planarity. **(ii)** Instead of discarding unreliable depth information, we make use of its noisy (probabilistic) structure to perform adaptive depth smoothing and adaptive robust plane fitting. We model the depth correction/completion process as a discrete MRF (Markov Random Field), a labeling problem that can be efficiently solved using iterative interactive graph cuts. The unary and binary terms of the MRF go beyond traditional definitions to stress appearance and 3D cues that encourage a 3D structure compatible with the piecewise planar assumption. **(iii)** To evaluate our proposed approach and validate the importance of depth correction and completion, we compile a challenging, large-scale dataset with ground truth depth. Additionally, we qualitatively evaluate our method on the NYUv2 dataset. Also, we illustrate the merit of our solution in a real-world application, namely RGB-D SLAM. We show that replacing raw depth maps with our corrected and completed ones substantially improves the performance of a state-of-the-art visual SLAM approach.

3 Proposed Method

In this work, two images generated by an RGB-D sensor (the MS Kinect) are taken as input, where both RGB and depth sensors are assumed to be calibrated (i.e., their intrinsic and extrinsic parameters are estimated beforehand). Denote the RGB and depth images as $\mathbf{I}_c \in \mathbb{R}^{M\times N}$ and $\mathbf{I}_d \in \mathbb{R}^{M\times N}$ respectively. In the rest of the paper, we denote $\mathbf{I}_d$ as the *raw* depth image, since it contains the unprocessed depth values measured by the sensor directly.

3.1 Pre-Processing

Before correcting and/or completing depth values in $\mathbf{I}_d$, we adaptively smooth them using a precomputed Kinect noise profile (see **Supplementary Material**) and we robustly extract planar segments from the smoothed 3D data. These segments constitute primitives for the piecewise planar scene, which will subsequently be filtered, grouped, and possibly extended.

Adaptive Depth Smoothing: To reduce the effect of sensor noise while preserving depth discontinuities, the projected 3D points are smoothed using the JBF defined in Eq (1). This filter makes use of both the depth image $\mathbf{I}_d$ and RGB image $\mathbf{I}_c$, as in [5]. Since this filtering method is unaware of the underlying 3D scene, we only use it to smooth existing depth data but not fill unknown depth values in $\mathbf{I}_d$.

$$\hat{\mathbf{I}}_d(\mathbf{p}) = \frac{1}{k} \sum_{\mathbf{q} \in \Omega} \mathbf{I}_d(\mathbf{q}) F(\mathbf{p}, \mathbf{q}) G(\mathbf{I}_d(\mathbf{p}), \mathbf{I}_d(\mathbf{q})) H(\mathbf{I}_c(\mathbf{p}), \mathbf{I}_c(\mathbf{q})) \tag{1}$$

In this filter, the smoothed depth at pixel $\mathbf{p}$ is a positive weighted sum of the raw depth values in the neighborhood Ω around $\mathbf{p}$. Each weight is a product of three similarity measures between $\mathbf{p}$ and its neighbor $\mathbf{q}$: within-image spatial closeness (defined by F), similarity in raw depth (defined by G), and similarity in appearance (defined by H). Similar to other methods, we take these three functions to be Gaussian. We model $G(\mathbf{I}_d(\mathbf{p}), \mathbf{I}_d(\mathbf{q}))$ as a Gaussian function $\mathcal{N}(\mathbf{I}_d(\mathbf{p}) - \mathbf{I}_d(\mathbf{q}), \sigma_d(\mathbf{I}_d(\mathbf{p})))$, where the standard deviation is depth dependent. By doing this, the JBF is made adaptive to varying depths, which specifically allows for more suitable smoothing at larger depths.

Adaptive and Robust Plane Fitting: After depth-adaptive smoothing, we aim to detect and fit 3D planes through the 3D projections of pixels in $\hat{\mathbf{I}}_d$ with known depth. Similar to previous methods, we use RANSAC for robust plane fitting. However, the criterion for a pixel to be an inlier (e.g. having the distance of its 3D projection to the plane be less than a threshold) should incorporate the noise in the sensor's depth measurements. Otherwise, fewer consistent planar segments can be detected at points that are farther away.

We start the RANSAC process at pixels with smoothed depth less than d_{rel}. We model the actual depth $d(\mathbf{p})$ at pixel $\mathbf{p}$ probabilistically as a Gaussian centered around the depth value $\hat{\mathbf{I}}_d(\mathbf{p})$ with a depth-varying standard deviation $\sigma_d(\hat{\mathbf{I}}_d(\mathbf{p}))$. Since the depth camera is calibrated, the 3D point corresponding to $\mathbf{p}$ is computed as $\mathbf{x}(\mathbf{p}) = d(\mathbf{p})\mathbf{K}^{-1}\tilde{\mathbf{p}} = d(\mathbf{p})\mathbf{t}(\mathbf{p})$, where $\tilde{\mathbf{p}}$ is $\mathbf{p}$ in homogenous coordinates and $\mathbf{K}$ is the camera matrix. It is easily shown that $\mathbf{x}(\mathbf{p})$ is a Gaussian random variable (centered around the observed 3D projection $\hat{\mathbf{I}}_d(\mathbf{p})\mathbf{t}(\mathbf{p})$). Similarly, the distance $D(\mathbf{x}(\mathbf{p})|(\mathbf{n}, n_0))$ between $\mathbf{x}(\mathbf{p})$ and a 3D plane parameterized by a unit normal $\mathbf{n}$ (pointing towards $\mathbf{x}(\mathbf{p})$) and offset n_0 also has a Gaussian distribution: $D(\mathbf{x}(\mathbf{p})|(\mathbf{n}, n_0)) \sim \mathcal{N}(\mu_D, \sigma_D^2)$, where

$$\mu_D = \hat{\mathbf{I}}_d(\mathbf{p})\mathbf{n}^T\mathbf{t}(\mathbf{p}) + n_0 \quad \text{and} \quad \sigma_D^2 = \sigma_d^2(\hat{\mathbf{I}}_d(\mathbf{p}))\|\mathbf{n} \circ \mathbf{t}(\mathbf{p})\|_2^2 \tag{2}$$

Note that $\circ$ is the Hadamard product. As expected, the variance of $D(\mathbf{x}(\mathbf{p})|(\mathbf{n}, n_0))$ increases with depth and varies with the relative orientation of the plane with the camera plane. By representing this distance probabilistically, we replace the usual RANSAC inlier condition $D(\mathbf{x}(\mathbf{p})|(\mathbf{n}, n_0)) \leq a$ with $p(D(\mathbf{x}(\mathbf{p})|(\mathbf{n}, n_0)) \leq a) \geq \theta$. The latter probability can be computed straightforwardly using the CDF of a Gaussian. In this paper, we take $a = 2$cm and $\theta = 0.8$. Also, normals of the observed 3D projections $\hat{\mathbf{I}}_d(\mathbf{p})\mathbf{t}(\mathbf{p})$ can be used to refine the inliers. Using this probabilistic rule allows a plane model $(\mathbf{n}, n_0)$ that is fit with 3D projections from pixels with reliable depth to extend into farther pixels with less reliable depth. This extension would not be possible and plane fragmentation would ensue, if the conventional RANSAC inlier condition is used.

Once the proposed plane-fitting method converges to a set of 3D plane equations and corresponding pixel inliers, we project the 3D projections of the inliers unto their respective planes and update $\hat{\mathbf{I}}_d$ to reflect this projection. This point-to-plane projection changes the depth values acquired by the sensor and, in the majority of cases, corrects their values when a piecewise planar scene is assumed. If a fitted plane exists such that the vast majority of projected points lie in the halfspace designated by its normal and the principal angle between its normal and the normal of the image plane is negative (clockwise), then this fitted plane is designated as the *floor*. A similar rule determines the ceiling plane, if it exists.

3.2 Depth Completion as an MRF

After pre-processing, the raw depth image $\mathbf{I}_d$ is transformed into $\hat{\mathbf{I}}_d$ and each pixel with a known depth value is assigned a label corresponding to the fitted plane it belongs to. We denote the resulting label image as $\mathbf{L}_0 \in \{0, 1, \ldots, l\}^{M \times N}$, where the 0 label designates pixels of unknown depth, 1 the floor (if it exists), and 2 the ceiling (if it exists). In this section, we describe how the initial label image $\mathbf{L}_0$ is relabelled through an iterative process that makes use of appearance and 3D cues from $\mathbf{I}_c$ and $\hat{\mathbf{I}}_d$. We denote the label image at iteration k as $\mathbf{L}_k$. As we will see, a consequence of this process is the iterative extension/completion of each plane label and the constriction of the 0 label. In other words, pixels with unknown depth values (labelled 0) can be assigned a plane label, if deemed likely from an appearance and 3D reasoning point of view. We formulate the relabeling process as an iterative *interactive graph cuts* problem, where regional (unary) and boundary (binary) terms are used to relabel all pixels in the image while enforcing the labels of pixels that already have plane labels.

Determining Background Pixels: Clearly, not all pixels in $\hat{\mathbf{I}}_d$ are projections of 3D points that belong to the l fitted planes. Due to sensor limitations and scene structure, other planes might exist in the 3D scene but they are not imaged at all. To allow pixels not to belong to the l fitted planes, we construct a *background* label, denoted as $(l+1)$. Since no depth information exists for background pixels, we label them according to how their projection rays (3D rays connecting the pixels to the camera's optical center) relate to the fitted 3D planes, as follows. A pixel is given an $(l + 1)$ label if **(i)** it cannot belong to any of the l fitted planes (i.e. its projection ray does not intersect any of the planes) or **(ii)** the intersection between its projection ray and each plane j is at least $d_{\max}$ far from the closest known 3D point of plane j. In our experiments, we take $d_{\max} = 1$ meter. Condition **(ii)** assumes that the farther away a point is from the observed points on the fitted planes, the more likely it belongs to the background label. Background pixels are labeled as $(l+1)$ and added to $\mathbf{L}_0$. Figure 2 shows an example of pixels labelled as background in a sample depth frame.

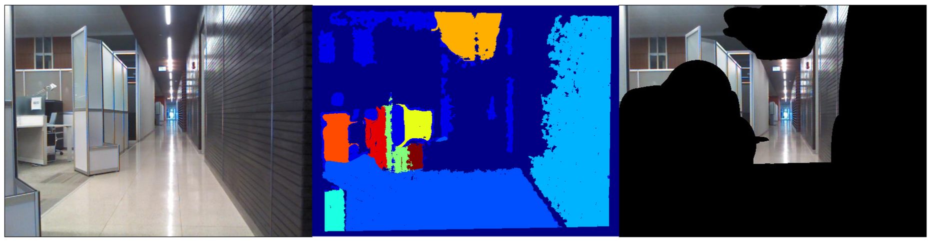


Fig. 2. Left: Original RGB image of a piecewise planar scene. **Middle:** Initial labeling of plane segments using our robust plane extraction in 3D (background *not* included). **Right:** The original RGB image showing only the pixels that are initially labelled in $\mathbf{L}_0$ as background.

Discriminative Appearance Model: We discriminatively model the appearance of each label (i.e. pixels whose 3D points belong to the same fitted plane). All labelled pixels in the image are represented using a set of low-level image features that describe a pixel's color (local color histogram), neighborhood structure (HOG features), and texture information (LBP features). Using PCA, dimensionality reduction is performed to maintain 90% variance in the labelled data. A discriminative appearance model is formed by training a multi-class RBF SVM classifier on all labelled pixels. This model is a vector scoring function $h(\mathbf{z}) \in \mathbb{R}^{l+1}$, where $h_i(\mathbf{z})$ is the SVM score of labelling feature vector $\mathbf{z}$ as i, for all $i \in \mathcal{L}$ such that $\mathcal{L} = \{1, \dots, l+1\}$.

Vanishing Lines: Apart from appearance, other cues exist that shed light on the 3D structure of the scene. A widely used cue is the existence of vanishing line segments. This cue has been extensively used in scene recognition and understanding from single RGB images, especially in indoor piecewise planar scenes [14]. In an image, vanishing points are usually extracted through a process of clustering line segments that vanish to the same point. However, in our case, we can explicitly compute certain vanishing points without any need for clustering or line detection. In indoor scenes, many plane segments tend to be perpendicular to each other (e.g. the *floor* and *walls*), thus, many parallel 3D lines in the scene (belonging to the same or different planes) tend to be perpendicular to another plane in the scene. Therefore, we can easily estimate the vanishing point of 3D parallel lines that are perpendicular to fitted plane i by simply constructing at least two such lines (parallel to the normal of plane i) and projecting them unto the RGB image $\mathbf{I}_c$. We denote these vanishing points as $\mathcal{V}_1 = \{\mathbf{v}_i\}_{i=1}^{l}$, where $\mathbf{v}_i$ is the vanishing point of parallel 3D lines that are perpendicular to plane i. We also use a method similar to [2] to obtain another set of vanishing points $\mathcal{V}_2$ that is disjoint from $\mathcal{V}_1$. We maintain a record of the clustered line segments that vanish to points in $\mathcal{V}_1$ and $\mathcal{V}_2$. Obviously, the process of extracting line segments and clustering vanishing points in $\mathcal{V}_2$ is prone to error, but it provides an additional 3D cue that can be harnessed in the relabeling process.

MRF Formulation: Given the label image $\mathbf{L}_0$, we now aim to label all pixels in $\hat{\mathbf{I}}_d$, especially those with unknown depth values. We model this labelling problem with a discrete Markov Random Field (MRF), where $\mathcal{L} = \{1, \dots, l+1\}$ is the set of discrete labels, $\mathcal{P}$ is the set of all pixels, and $\mathcal{E}$ is the set of all connections defining local 8-connected neighborhoods around each pixel. We seek a labeling $\mathbf{f}^*$ that minimizes the energy in Eq (3).

$$E(\mathbf{f}) = \sum_{\mathbf{p}\in\mathcal{P}} U(\mathbf{f}_p|\mathbf{p}) + \lambda \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{E}} B(\mathbf{p},\mathbf{q}) \tag{3}$$

Here, $U(\mathbf{f}_p|\mathbf{p})$ defines the unary (or data) term, which quantifies the cost incurred when pixel $\mathbf{p}$ is assigned to label $\mathbf{f}_p \in \mathcal{L}$. Alone, this term treats pixels in $\hat{\mathbf{I}}_d$ independently, so a binary (or smoothness) term $B(\mathbf{p},\mathbf{q})$ is added for regularization, with tradeoff coefficient λ. This term quantifies the cost of assigning neighboring pixels $\mathbf{p}$ and $\mathbf{q}$ to different labels, i.e. $\mathbf{f}_p \neq \mathbf{f}_q$. This energy can be minimized efficiently using graph cuts [3]. Since some pixels in $\hat{\mathbf{I}}_d$ are already assigned to fitted planes with non-background labels, we use a version of graph cuts (popularly known as *interactive graph cuts*) to guarantee that the labels of these pixels, after optimization, remain the same. Other optimization methods could be used to solve the normalized version of this problem [12].

Unary (data) Term: Although interactive graph cuts is well-known and has been used for various labelling problems in the past, the quality of the final labeling is mainly determined by how appropriate and informative the unary and binary terms are. Here, the unary term is inversely proportional to the likelihood of a pixel belonging to a fitted plane. This term compares the appearance of a pixel to the discriminative appearance model of each plane and prevents label assignments that are incompatible in 3D under the piecewise planar assumption. In general, we set $U(i|\mathbf{p}) = -h_i(\mathbf{z}(\mathbf{p}))$ for each pixel $\mathbf{p} \in \mathcal{P}$ and each label $i \in \mathcal{L}$. This assumes that points belonging to the same plane look similar, which is usually a valid assumption. In what follows, we use 3D cues (from both $\hat{\mathbf{I}}_d$ and $\hat{\mathbf{I}}_c$) to regularize the labelling further.

To enforce the hard constraints, we follow a similar strategy as in [3], where $U(i|\mathbf{p}) = K \gg 0$ for each $i \in \mathcal{L}\backslash\{l+1\}$ and $\mathbf{p}$ such that $\mathbf{L}_0(\mathbf{p}) = j \neq i$. This large cost K prevents a pixel that is already labelled in $\mathbf{L}_0$ to switch labels. This enforcement is done for all labels except background $(l+1)$, which we allow to constrict or expand. Moreover, we set $U(i|\mathbf{p}) = K$ for any pixel $\mathbf{p}$ whose projection ray does not intersect plane i. Also, we enforce that all intersections between projection rays and planes occur *above* the floor plane (if it exists) and *below* the ceiling plane (if it exists). This prohibits label assignments that lead to 3D points, which tunnel into the floor or pierce the ceiling. For these pixels, we set $U(1|\mathbf{p}) = K$ and/or $U(2|\mathbf{p}) = K$. Lastly, the unary term should not lead to label assignments that are contradictory in 3D. As such, we set $U(i|\mathbf{p}) = K$ for any pixel $\mathbf{p}$, which belongs to a line segment that vanishes to $\mathbf{v}_i \in \mathcal{V}_1$. This discourages $\mathbf{p}$ from belonging to a fitted plane, if it lies on a line perpendicular to that plane. In this case, the points belonging to the intersection of two perpendicular planes will be assigned to only one of the two.

Binary (smoothness) Term: To allow for interactions between neighboring pixels, we define a binary term $B(\mathbf{p}, \mathbf{q})$, which encourages label smoothness among pixel neighbors that have similar appearance and/or that are likely to belong to the same plane in 3D. In general, we set $B(\mathbf{p}, \mathbf{q}) = \exp(-\Delta_c(\mathbf{p}, \mathbf{q})\mathbf{\Sigma}_c^{-1}\Delta_c(\mathbf{p}, \mathbf{q}))$, where $\Delta_c(\mathbf{p}, \mathbf{q}) = \mathbf{I}_c(\mathbf{p}) - \mathbf{I}_c(\mathbf{p})$. The covariance matrix $\mathbf{\Sigma}_c$ is estimated from $\mathbf{I}_c$. Moreover, we make use of vanishing points to discourage pixels lying on the same vanishing line segment to belong to different planes. So, we set $B(\mathbf{p}, \mathbf{q}) = K$ when pixels $\mathbf{p}$ and $\mathbf{q}$ belong to the same line segment that vanishes to a point in $\mathcal{V}_1 \cup \mathcal{V}_2$. In our experiments and as suggested in [3], we set $K = 1 + \max_{\mathbf{p}\in\mathcal{P}} \sum_{\mathbf{q}:(\mathbf{p},\mathbf{q})\in\mathcal{E}} B(\mathbf{p}, \mathbf{q})$.

Iterative Solution: Once all unary and binary terms are computed for all pixels and labels, we solve the labeling problem using interactive graph cuts [3]. Upon convergence, we use the final labeling $\mathbf{f}^*$ to determine the depth value of each pixel with label $\mathbf{f}_p^* \in \{1, \ldots, l\}$. This is done by intersecting the projection ray of $\mathbf{p}$ with fitted plane $\mathbf{f}_p^*$. The depth of a pixel labeled as background (i.e. $\mathbf{f}_p^* = l+1$) remains unknown. Effectively, the original label image $\mathbf{L}_0$ has been relabelled to produce a modified version $\mathbf{L}_1$. Many pixels that had unknown depth values in $\mathbf{L}_0$ have been assigned depth values in $\mathbf{L}_1$. These depth values encourage conformity to appearance models of existing planes and non-contradictory 3D layouts in piecewise planar scenes. In fact, the relabeling process can be rerun with $\mathbf{L}_1$ replacing $\mathbf{L}_0$. Obviously, the plane equations can be refined, the appearance model for each label needs to be retrained, and the unary and binary terms should reflect the changes in hard constraints. The resulting iterative relabeling process converges at iteration k when the change in label image $\delta(\mathbf{L}_k, \mathbf{L}_{k-1})$ is negligible. As such, our proposed approach corrects (through robust plane fitting and projection) and completes (through appearance and 3D aware relabeling) raw depth values recorded by an RGB-D sensor in a piecewise planar scene. Figure 3 shows the end-to-end approach applied to a sample RGB-D image pair.

4 Experimental Results

In this section, we evaluate the effectiveness of our method in enhanceing/correcting and completing depth measurements obtained by the Kinect. For a quantitative comparison between the depth maps generated by our method and the raw depth

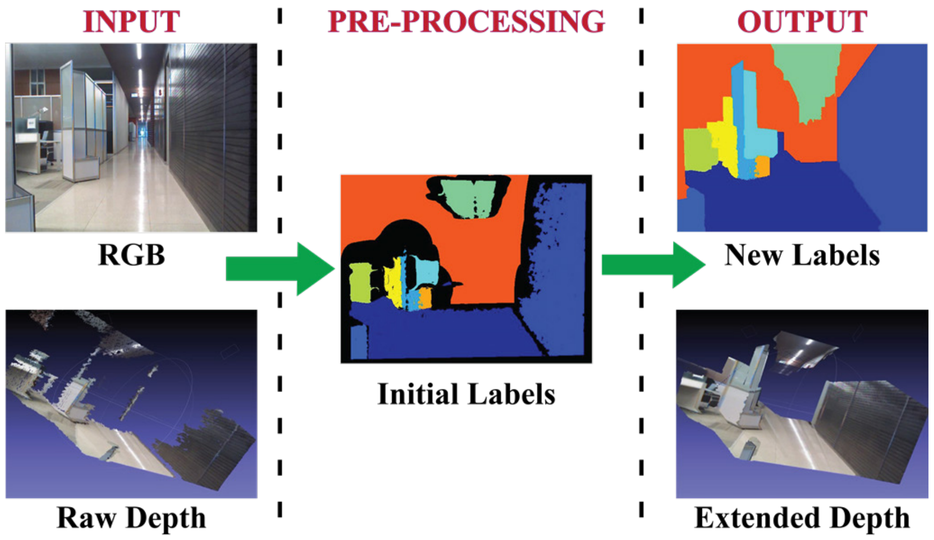


Fig. 3. End-to-end process. *Input:* The system takes as input an RGB-D image pair. *Pre-processing:* The input depth is smoothed using a JBF, after which, plane segments are extracted using the adaptive/robust RANSAC method described in this section. Using the extracted plane segments, an initial label image is created. In addition, background pixels are extracted and added as a new label. In the initial label image, each label is given a distinct color, and orange is used here to describe background pixels. These initial labels are fed to the Graph Cut solver. *Output:* The final output is a complete set of labels, which are converted to new depth values. We can observe how the depth range is extended from the original 3D point cloud to the final result of our correction/completion process.

maps obtained from the sensor, we generate a large-scale 3D point cloud of a typical indoor scene, which we use to generate ground truth depth maps. Additionally, we show some quantitative results on the NYUv2 dataset. We also show how our method can be used in to enhance applications like RGB-D SLAM.

4.1 Dataset Compilation

To create the ground truth set, we scanned a large indoor area using 2 Kinect sensors mounted on a moving platform. The devices were pointed at different fields of view to avoid possible interference of their light patterns. We used 2 devices to be able to cover large areas in the scene, but each device reconstructs its view independently from the other. The relative movement of the platform was constrained to a fixed translation along a predefined direction perpendicular to the floor normal, and a rotation of 0 or $\pm 90°$ around the floor normal; such restrictions make the registration between frames trivial. A total of 700 frames were recorded, corresponding to 220 meters of stretch inside a typical office space. In order to ensure data precision, we select from each frame depth values that only lie within a 0.8 - 4 meters reliable range. Figure 4 shows 2 different views of the final point cloud, with a rough size of 63 million points, as well as a triplet of color, raw depth, and ground truth depth for the same image.

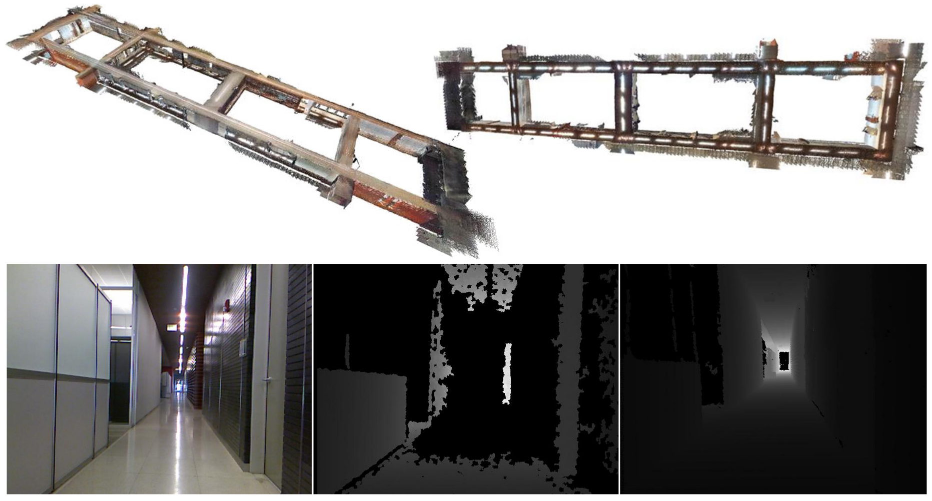


Fig. 4. Top: Different views of our large 3D reconstructed dataset. The point cloud covered a physical area of 220 meters and comprised around 63 million points. It contains challenging images of reflective and transparent surfaces, where the Kinect fails to estimate depth values. The depth images obtained are comprised of large gaps with unknown depth as shown in Figure 1. Bottom: RGB, raw depth, and ground truth depth. We notice the large amount of points missing in the raw depth image, due to the sensor's inability to process large dark areas and reflective surfaces. The ground truth depth frame shows a complete version of the view by back-projecting the large 3D point cloud aligned to this frame's view.

4.2 Quantitative Comparison

Using the ground truth set, we test the accuracy of our approach and compare it to the raw data provided by the Kinect. The result of our application is a new set of depth frames that improve the raw Kinect data in 2 ways, first by correcting depth values, particularly at large depths, and second by increasing the number of available depth pixels. In order to test for the first hypothesis, we compare the enhanced and raw depth with the ground truth 3D points. The comparison is done by converting the depth values of the enhanced and raw frames into 3D points, and calculating errors as Euclidean distances between closest points of these frames and the 3D point cloud of the scene. In order to do this computation in an efficient manner, we build a K-D tree for the ground truth point cloud once, and query the tree using the 3D points of each frame. Figure 5 *Left* plots these errors for both the raw depth data of the RGB-D sensor and our approach. Our method increases the accuracy of the depth values by an average of 0.5m. We show a wide range of qualitative results on the **Supplementary Material**.

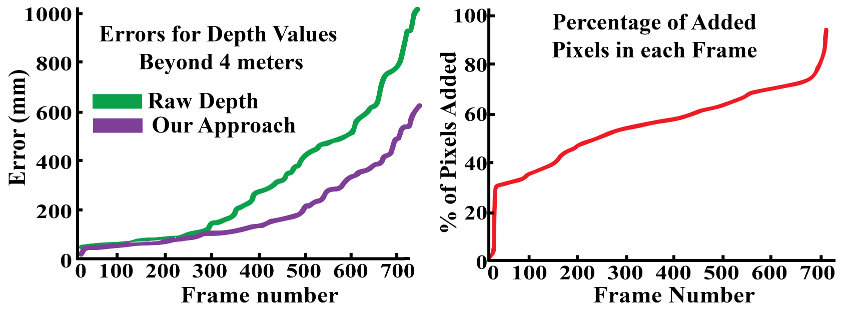


Fig. 5. Left: Comparison of raw depth values and depth values generated by our method w.r.t. ground truth for a large-scale piecewise planar scene. The plot shows the error per frame in increasing order of error. We notice the significantly lower errors obtained by applying our method. This improvement comes from a combination of correction and completion of the raw depth. **Right:** Percentage of depth values added to a single image. Our method is adding an average of 50% of depth pixels, and sometimes as much as 80%.

Since our method not only corrects but also extends the depth range of the sensor data; and thus increasing the number of depth pixels available, we also look at the average pixel increase as an additional measure of performance. Figure 5 *Right* shows a plot of percentage pixel increase per frame, calculated as the percentage of pixels added by the our correction/completion with respect to the original raw data available. The mean increase is around 40000 pixels, with standard deviation of around 20000 pixels. We note at this point that although an increase is present in more than 80% of the cases, the amount of increase always scene-dependent, thus resulting in high standard deviations.

4.3 Result on NYUv2 Dataset

In addition to testing our proposed method on the set we compiled, we also make use of images from the popular NYUv2 dataset [25]. We choose images of piecewise planar environments (such as corridors or hallways) and apply our method on the raw depth images. We compare our corrected and completed depth map to that generated by the hole-filling (colorization) method in [23] used to colorize depth in NYUv2 . We show some examples in Figure 6, where we render 3D views of the raw depth data and the enhanced depth obtained by both techniques. Since the colorization technique is unaware of the 3D structure of the scene, depth points that were added by this method were not constrained to belong to any plane, leading to significant errors. As pointed out in Figure 6, these errors come from bad depth predictions at large gaps as well as noise created ad depth discontinuities. Our method does not suffer from these drawbacks due to its 3D aware nature while estimating unknown depth. Further results are available in the **Supplementary Material**.

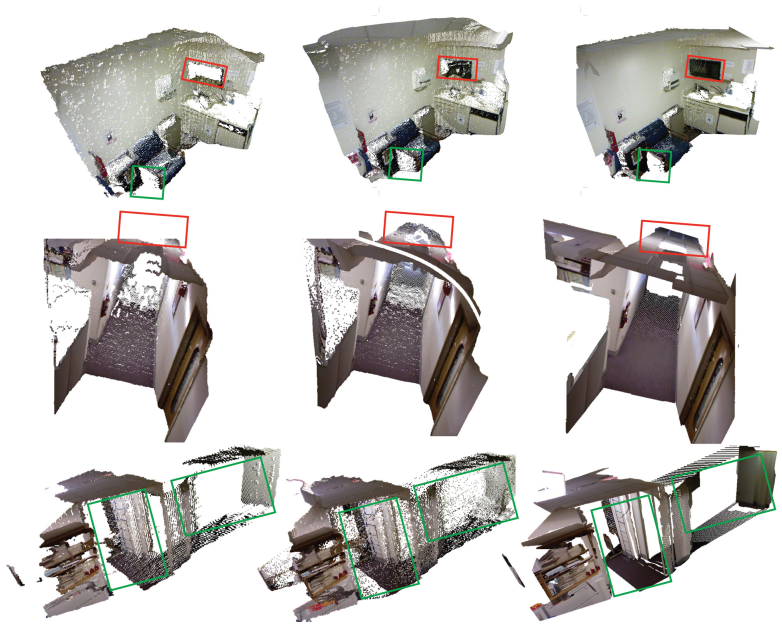


Fig. 6. Left to right: 3D renderings of the raw depth, depth after applying the colorization method in [23], depth from our proposed method. The depth colorization method creates unwanted artifacts at large depth gaps (red boxes), as well as a lot of noise at depth discontinuities (green boxes). these problems are not present in our method, since it provides a more 3D aware approach to estimating unknown depth.

4.4 RGB-D SLAM

In order to substantiate the usage of the corrected depth data over regular raw sensor data, we test the performance of a well-known application, the RGBD SLAM, by using the enhanced depth. We tackle the problem of localization in feature-poor environments such as hallways. Such environments are challenging because of the low number of features (SIFTs or SURFs) within the range of the Kinect sensor that could be tracked to estimate the sensor motion. Moreover, using ICP algorithm to estimate the motion in those environments faces convergence to local minima solutions due to the lack of 3D cues. We use a method similar to the one used by [7] with SIFT features. This method can be complemented by other methods, such as RGBD ICP by [17], but the goal is to assess the advantages of providing the corrected depth as an input to any application that uses RGBD data. We compare our results to the ground truth data by measuring two errors: the drift in the sensor motion and that of the 3D point locations. The error in 3D point locations is computed as the distance between the predicated 3D point location and the true location where the predicted point location is calculated after movements of 2 meters throughout the hallway. Results are shown in Figure 7. The mean error in point locations is 682mm when using the raw depth and 436mm when using the corrected depth data. For translation, the average drift is 440mm when using raw depth and 283mm when using the our ones. Figure 7 also shows the difference in trajectory between the 2 methods, showing significant improvement when using the corrected/completed depth data.

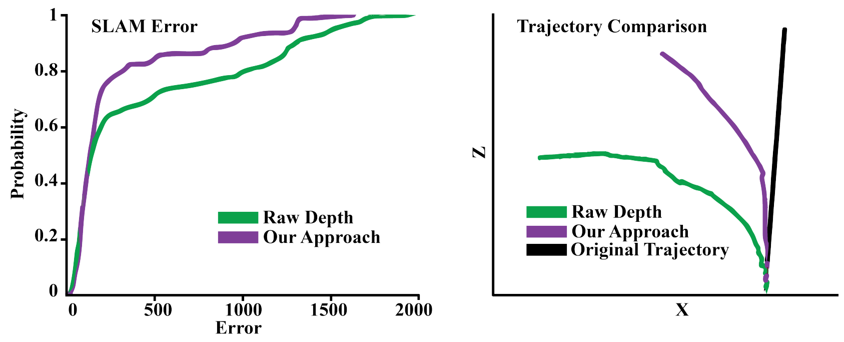


Fig. 7. RGB-D SLAM comparison. **Left:** A cumulative density distribution of the frame-to-frame SLAM error when using raw depth and our depth on 700 images. **Right:** We see in this plot the trajectory estimation of RGB-D SLAM using the raw depth (green), and using our depth (magenta), compared to the original trajectory (black). The improvement (after enhancement) in drift is substantial: 283mm as opposed to 440mm with the raw depth.

5 Conclusion

In this paper we presented a method to use the complete set of depth values provided by an RGB-D sensor to better represent Manhattan environment scenes. We showed that by properly analyzing the sensor error at large depth, we can correct the data given, and segment planar labels from the scene. By applying our method to a new large-scale ground truth data set, we show that our framework provides more accurate depth maps, having a larger number of pixels (20% average increase) than those recorded by the RGB-D sensor. In addition, we qualitatively showed the power of our technique on the NYUv2 dataset. We also applied our methodology to improve the performance of RGB-D SLAM.

Acknowledgement. Research reported in this publication was supported by competitive research funding from King Abdullah University of Science and Technology (KAUST).

References

1. Barron, J.T., Malik, J., Berkeley, U.C.: Intrinsic Scene Properties from a Single RGB-D Image. CVPR (2013)
2. Bazin, J., Seo, Y.: Globally optimal line clustering and vanishing point estimation in manhattan world. CVPR (2012)
3. Boykov, Y., Funka-Lea, G.: Graph Cuts and Efficient N-D Image Segmentation. IJCV 70(2), 109–131 (2006)
4. Camplani, M., Salgado, L.: Efficient spatio-temporal hole filling strategy for kinect depth maps. SPIE (2012)
5. Cheung, S.c.S.: Layer Depth Denoising and Completion for Structured-Light RGB-D Cameras. CVPR (2013)
6. Diebel, J., Thrun, S.: An application of markov random fields to range sensing. NIPS (2005)
7. Endres, F., Hess, J., Engelhard, N., Sturm, J., Cremers, D., Burgard, W.: An evaluation of the RGB-D SLAM system. ICRA (2012)
8. Flint, A., Murray, D., Reid, I.: Manhattan scene understanding using monocular, stereo, and 3D features. ICCV (2011)
9. Furukawa, Y., Curless, B., Seitz, S., Szeliski, R.: Manhattan-world stereo. CVPR (2009)
10. Furukawa, Y., Curless, B., Seitz, S.M., Szeliski, R.: Reconstructing building interiors from images. ICCV (2009)
11. Gallup, D., Frahm, J.M., Pollefeys, M.: Piecewise planar and non-planar stereo for urban scene reconstruction. CVPR (2012)
12. Ghanem, B., Ahuja, N.: Dinkelbach NCUT: An efficient framework for solving normalized cuts problems with priors and convex constraints. IJCV 89(1), 40–55 (2010)
13. Gupta, S., Arbel, P., Malik, J., Berkeley, B.: Perceptual Organization and Recognition of Indoor Scenes from RGB-D Images. CVPR (2013)
14. Hedau, V., Hoiem, D., Forsyth, D.: Recovering the spatial layout of cluttered rooms. CVPR (2009)
15. Hedau, V., Hoiem, D., Forsyth, D.: Thinking inside the box: Using appearance models and context based on room geometry. ECCV (2010)
16. Hedau, V., Hoiem, D., Forsyth, D.: Recovering free space of indoor scenes from a single image. CVPR (2012)
17. Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D.: RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. IJRR 31(5), 647–663 (2012)
18. Hu, G., Huang, S., Zhao, L.: A robust RGB-D SLAM algorithm. IROS (2012)
19. Jia, Z., Gallagher, A., Saxena, A., Chen, T.: 3D-Based Reasoning with Blocks, Support, and Stability. CVPR (2013)
20. Kim, B.s., Arbor, A., Savarese, S.: Accurate Localization of 3D Objects from RGB-D Data using Segmentation Hypotheses. CVPR (2013)
21. Kopf, J., Cohen, M.: Joint bilateral upsampling. SIGGRAPH (2007)
22. Koppula, H.S., Anand, A., Joachims, T., Saxena, A.: Semantic Labeling of 3D Point Clouds for Indoor Scenes. NIPS (2011)
23. Levin, A., Lischinski, D., Weiss, Y.: Colorization using optimization. SIGGRAPH (2004)
24. Park, J., Kim, H., Brown, M.S., Kweon, I.: High quality depth map upsampling for 3D-TOF cameras. ICCV (2011)

25. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from rgbd images. ECCV (2012)
26. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of RGB-D SLAM systems. IROS. (2012)
27. Wang, L., Jin, H., Yang, R., Gong, M.: Stereoscopic inpainting: Joint color and depth completion from stereo images. CVPR (2008)