A Contour Completion Model for Augmenting Surface Reconstructions

Nathan Silberman[1], Lior Shapira [2], Ran Gal[2], Pushmeet Kohli[3]

[1]Courant Institute, New York University,
[2] Microsoft Research

Abstract. The availability of commodity depth sensors such as Kinect has enabled development of methods which can densely reconstruct arbitrary scenes. While the results of these methods are accurate and visually appealing, they are quite often incomplete. This is either due to the fact that only part of the space was visible during the data capture process or due to the surfaces being occluded by other objects in the scene. In this paper, we address the problem of completing and refining such reconstructions. We propose a method for scene completion that can infer the layout of the complete room and the full extent of partially occluded objects. We propose a new probabilistic model, Contour Completion Random Fields, that allows us to complete the boundaries of occluded surfaces. We evaluate our method on synthetic and real world reconstructions of 3D scenes and show that it quantitatively and qualitatively outperforms standard methods. We created a large dataset of partial and complete reconstructions which we will make available to the community as a benchmark for the scene completion task. Finally, we demonstrate the practical utility of our algorithm via an augmented-reality application where objects interact with the completed reconstructions inferred by our method.

Keywords: 3D Reconstruction, Scene Completion, Surface Reconstruction, Contour Completion

1 Introduction

The task of generating dense 3D reconstruction of scenes has seen great progress in the last few years. While part of this progress is due to algorithmic improvements, large strides have been made with the adoption of inexpensive depth cameras and the fusion of color and depth signals. The combined use of depth and colour signals has been successfully demonstrated for the production of large-scale models of indoor scenes via both offline [1] and online [2] algorithms. Most RGB+D reconstruction methods require data that show the scene from a multitude of viewpoints and are not well suited for input sequences which contain a single-view or limited number of viewpoints. Moreover, these reconstruction methods are hindered by occlusion as they make no effort to infer the geometry of parts of the scene that are not visible in the input sequences. Consequently, the resulting 3D models often contain gaps or holes and do not capture certain basic elements of a scene, such as the true extent and shape of objects and scene surfaces.

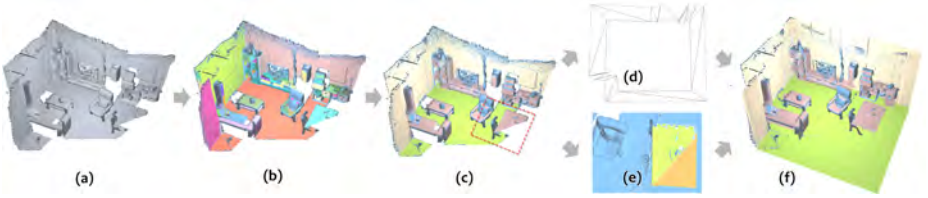


Fig. 1. Overview of the pipeline: Given a dense but incomplete reconstruction (a) we first detect planar regions (b) and classify them into ceiling (not shown), walls (tan), floor (green) and interior surfaces (pink) (c) Non-planar regions are shown in light blue. Next, we infer the scene boundaries(d) and shape of partially occluded interior objects, such as the dining table (e) to produce a complete reconstructed scene (f). Additional visualizations are found in the supplementary material.

Accurate and complete surface reconstruction is of special importance in Augmented Reality (AR) applications which are increasingly being used for both entertainment and commerce. For example, a recently introduced gaming platform [3] asks users to scan an interior scene from multiple angles. Using the densely reconstructed model, the platform overlays graphically generated characters and gaming elements. Furniture retailers (such as IKEA) enable customers to visualize how their furniture will look when installed without having to leave their homes. These applications often require a high fidelity dense reconstruction so that simulated physical phenomenon, such as lighting, shadow and object interactions (e.g. collisions) can be produced in a plausible fashion. Unfortunately, such reconstructions often require considerable effort on the part of the user. Applications either demand that users provide video capture of sufficient viewpoint diversity or operate using an incomplete model of the scene.

Our goal is to complement the surface reconstruction for an input sequence that is limited in viewpoint, and "fill in" parts of the scene that are occluded or not visible to the camera. This goal is driven by both a theoretical motivation and a practical application. Firstly, basic scene understanding requires high level knowledge of how objects interact and extend in 3D spaces. While most scene understanding research is concerned with semantics and pixel labeling, relatively little work has gone into inferring object or surface extent, despite the prevalence and elemental nature of this faculty in humans. Second, online surface reconstruction pipelines such as KinectFusion [2] are highly suitable for AR applications, and could benefit from a scene completion phase, integrated into the pipeline.

Our method assumes a partial dense reconstruction of the scene that is represented as a voxel grid where each voxel can be occupied, free or its state in unknown. We use the KinectFusion [2] method to compute this reconstruction which also assigns a surface normal and truncated signed distance function (TSDF) [4] to the voxels. Given this input, our method first detects planar surfaces in the scene and classifies each one as being part of the scene layout (floor, walls

ceiling) or part of an internal object. We then use the identities of the planes to extend them by solving a labeling problem using our Contour-completion random field (CCRF) model. Unlike pairwise Markov random fields, which essentially encourage short boundaries, the CCRF model encourages the discontinuities in the labeling to follow detected contour primitives such as lines or curves. We use this model to complete both the floor map for the scene and estimate the extents of planar objects in the room. This provides us with a watertight 3D model of the room. Finally we augment the original input volume account for our extended and filled scene. The stages of our algorithm are demonstrated in figure 1.

Our Contributions: This paper makes the following technical contributions: (1) We introduce the Contour Completion Random Field (CCRF), a novel algorithm for completing or extending contours in 2D scenes. (2) We describe an algorithm for inferring the extent of a scene and its objects. (3) We demonstrate how to efficiently re-integrate the inferred scene elements into a 3D scene representation to allow them to be used by a downstream application.

The rest of the paper is organized as follows. We discuss the relationship of our work to the prior art in dense reconstruction in section 2. In section 3 we describe how we detect and classify planar surfaces in the scene. In section 4 we discuss how we perform scene completion by inferring the extent of scene boundaries and extending internal planes using our novel contour completion MRF. The procedure for augmenting the original (TSDF) based reconstruction with new values is discussed in section 5. In section 6 we perform qualitative and quantitative evaluation of our methods and in section 7 discuss our results, limitations and future work.

2 Related Work

KinectFusion [2] is a real-time dense surface mapping and tracking algorithm. It maintains an active volume in GPU memory, updated regularly with new depth observations from a Kinect camera. Each depth frame is tracked using the iterative closest-point algorithm, and updates the reconstruction volume which is represented as a truncated signed-distance function (TSDF) grid. At any point the TSDF volume may be rendered (using ray casting) or transformed into an explicit surface using marching cubes [5] (or similar algorithm). Our method extends this pipeline by accessing the TSDF volume at certain key frames, and augmenting it. As new depth observations reveal previously occluded voxels, our augmentations are phased out. Our scene completion algorithm may be used to augment other surface reconstruction methods such as [6,7] with little change. However, most of these methods have different motivation, and are not real-time. In SLAM++ [8], a database of 3D models is used during the reconstruction process. This helps improve tracking, reduces representation size, and augments the reconstructed scene. However, this method requires intensive preparation time for each new scene in order to capture the various repeated objects found in each one.

In recent years there have been many papers on scene understanding from a single image. Barinova et al [9] geometrically parse a single image to identify edges, parallel lines and vanishing point, and a level horizon. Hoiem et al [10] estimate the layout of a room from a single image, by using a generalized box detector. This approach is not well suited to completing either highly occluded scenes or non-rectangular objects. Ruiqi and Hoiem [11] attempt to label occluded surfaces using a learning-based approach. This method is limited by the number of classes which can be learned, and does not infer the extent of the hidden objects. A more recent paper by the same authors [12] is more relevant to our approach. They detect a set of supporting horizontal surfaces in a single-image depth frame, and their extent is deduced using the features at each point and surrounding predictions. However, they do not attempt to infer the full scene layout and limit themselves to support-related objects.

Min Kim et al [13] augment and accelerate reconstruction of complex scenes by carefully scanning and modeling objects which repeat in the scene, in advance. These objects (e.g. office chairs, monitors) are then matched in the noisy and incomplete point cloud of the larger scene, and used to augment it. This approach is less suited to capturing a new environment, in which we cannot model the repeating objects independently (given that there are any). Kim et al[14] jointly estimate the 3D reconstruction of a scene, and the semantic labels associated with each voxel, on a coarse 3D volume. Their method works best with simple occlusions and does not extend the volume to estimate the overall shape of the room. Zheng et al [15] employ physical and geometrical reasoning to reconstruct a scene. Their method relies on successful object segmentation, and a Manhattan-world assumption. They fill occluded parts of objects by extrapolating along the major axes of the scene. However, none of these methods can handle previously unseen object and surfaces with non-linear boundaries.

3 Plane Detection and Classification

We now describe how our method detects the dominant planes from the partial voxel based reconstruction. We denote the space of all possible 3D planes by $\mathcal{H}$, and the set of planes present in the scene by H. Let the set of all 3D points visible in the scene be denoted by $\mathcal{P} = \{p_1, ...p_N\}$. We estimate the most probable labeling for H by minimizing the following energy function:

$$H* = \arg\min_{H \subset \mathcal{H}} \sum_{i=1}^{N} f_i(H) + \lambda |H|. \quad (1)$$

where λ is a penalty on the number of planes and f_i is a function that penalizes the number of points not explained by any plane: $f_i(H) = \min\{\min_{h \in H} [\delta(p_i, h), \lambda_b]\}$, where the function δ returns a value of 0 if point p_i falls on plane h and is infinity otherwise. Minimizing the first term alone has the trivial degenerate solution where we include a plane for every point p_i in the set H. However, this situation is avoided by the second terms of the energy which acts as a regularizer and adds a penalty that linearly increases with the cardinality of H.

Lemma 1. *The energy function defined in equation (1) is a supermodular set function.*

Proof in the supplemental material

Computing the Optimal H Minimization of a super-modular function is an NP-hard problem even when the set of possible elements is finite (which is not true in our case). We employ a greedy strategy, starting from an empty set and repeatedly adding the element that leads to the greatest energy reduction. This method has been observed to achieve a good approximation [16]. We begin by using the Hough transform [17] to select a finite set of planes. In our method, each 3D point and its surface normal votes for a plane equation parameterized by its azimuth θ, elevation ψ and distance from the origin ρ. Each of these votes is accrued in an accumulator matrix of size $A \times E \times D$ where A is the number of azimuth bins, E is the number of elevation bins and D is the number of distance bins [1]. After each point has voted, we run non-maximal suppression to avoid accepting multiple planes that are too similar.

Once we have a set of candidate planes we sort them in descending order by the number of votes they have received and iteratively associate points to each plane. A point can be associated to a plane if it has not been previously associated to any other plane and if its planar disparity and local surface normal difference are small enough [2]. As an additional heuristic, each new plane and its associated points are broken into a set of connected components ensuring that planes are locally connected.

Semantic Labeling Once we have a set of planes, we classify each one independantly into one of four semantic classes: Floor, Wall, Ceiling and Internal. To do so, we train a Random Forest Classifier to predict each plane's class using the ground truth labels and 3D features from [18], which capture attributes of each plane including its height in the room, size and surface normal distribution. Planes classified as one of Floor, Wall and Ceiling will be used for inferring the floor plan and scene boundaries (section 4.6), whereas Internal planes will be extended and filled in a subsequent step (section 4.7).

4 Scene Completion

Given the set of detected and classified planes we infer the true extent of the scene, *ie.* obtain a water-tight room structure, and extend interior planes based on evidence from the scene itself.

4.1 Completion as a Labeling Problem

We now describe how to estimate the boundaries of planes as seen from a top-down view. We formulate boundary completion as a pixel labeling problem. Consider a

[1] We use A=128, E=64 and D is found dynamically by spacing bin edges of size 5cm apart between the max and minimum points

[2] Planar disparity threshold=.1, angular disparity threshold = .1

set $\mathcal{S}$ of nodes that represent grid locations in the top-down view of the scene. We assume that a partial labeling of nodes $i \in \mathcal{S}$ in the grid can be observed and is encoded by variables $y_i; i \in \mathcal{S}$ where $y_i = 1$, $y_i = 0$ and $y_i = -1$ represent that i belongs to the plane, does not belong to the plane, and its membership is uncertain respectively. Given $\mathbf{y}$, we want to estimate the true extent of the plane which we denote by $\mathbf{x}$. Specifically, we will use the binary variable x_i to encode whether the plane covers the location of node i in the top-view. $x_i = 1$ represents that node i belongs to the plane while $x_1 = 0$ represents that it does not.

The traditional approach for pixel labeling problems is to use a pairwise Markov Random Field (MRF) model. The energy of any labeling $\mathbf{y}$ under the pairwise MRF model is defined as: $E(\mathbf{x}) = \sum_{i \in \mathcal{S}} \phi_i(x_i) + \sum_{ij \in \mathcal{N}} \phi_{ij}(x_i, x_j)$, where ϕ_i encode the cost of assigning a label x_i and ϕ_{ij} are pairwise potentials that encourage neighboring ($\mathcal{N}$) nodes to take the same label, and K is a constant. The unary potential functions force the estimated labels $\mathbf{x}$ to be consistent with the observations $\mathbf{y}$, *ie.* $\phi_i(x_i) = \inf$ if $y_i \neq -1$ and $x_i \neq y_i$, and $\phi_i(y_i) = 0$ for all other cases, while the pairwise potentials take the form an Ising model. The Maximum a Posteriori (MAP) labeling under the model can be computed in polynomial time using graph cuts. However, the results are underwhelming as the pairwise model does not encode any information about how boundaries should be completed. It simply encourages a labeling that has a small number of discontinuities.

4.2 Contour Completion Random Field

Unlike the standard MRF which penalizes the number of transitions in the labeling, our Contour Completion Random Field (CCRF) model adds a penalty based on the least number of curve primitives that can explain all the transitions. We implement this by introducing higher order potentials in the model. These potentials are defined over overlapping sets of edges where each set follows some simple (low-dimensional) primitive curve shape such as a line or a circle. Formally, the energy function for the CCRF model can be written as:

$$E(\mathbf{x}) = \sum_{i \in \mathcal{S}} \phi_i(x_i) + \sum_{g \in \mathcal{G}} \Psi_g(\mathbf{x}) \tag{2}$$

where Ψ_g are our curve completion potentials, and $\mathcal{G}$ is a set where Each curve g represents a set of nodes (edges) that follow a curve. The curve completion potentials have a diminishing returns property. More formally,

$$\Psi_g(\mathbf{x}) = F\Big(\sum_{ij \in \mathcal{E}_g} \psi_{ij}(x_i, x_j) \Big), \tag{3}$$

where $\mathcal{E}_g$ is the set of edges that defines the curve or edge group g. F is a non-decreasing concave function. In our experiments, we defined F as an upper-bounded linear function *ie.* $F(t) = \min\{\lambda * t, \theta\}$ where λ is the slope of the function and θ is the upper-bound. It can be seen that once a few edges are

cut $t \geq \frac{\theta}{\lambda}$, the rest of the edges in the group can be cut without any penalty. This behavior of the model does not prevent the boundary in the labeling from including large number of edges as long as they belong to the same group (curve). The exact nature of these groups are described below.

4.3 Defining Edge Groups

We consider two types of edge groups: straight lines and parabolas. While previous work has demonstrated the ability of the hough transform [17] to detect other shapes, such as circles and ellipses, such high parameter shapes require substantially more memory and computation and we found lines and parabolas sufficiently flexible to capture most of the cases we encountered.

To detect lines, we used a modified Hough transform to not only detect lines in the image, but also the direction of the transition (the plane to free space or vice-versa). We use an accumulator with 3 parameters: ρ, the distance from the origin to the line, θ, the angle between the vector from the origin to the line and the X axis, and a quaternary variable d, which indicates the direction of the transition (both bottom-top and left-right directions) [3]. Following the accumulation of votes, we run non-maximal suppression and create an edge group for each resulting line.

The standard Hough transform for parabolas requires 4 parameters. To avoid the computational and memory demands of such a design, we introduce a novel and simple heuristic detailed in the supplemental material.

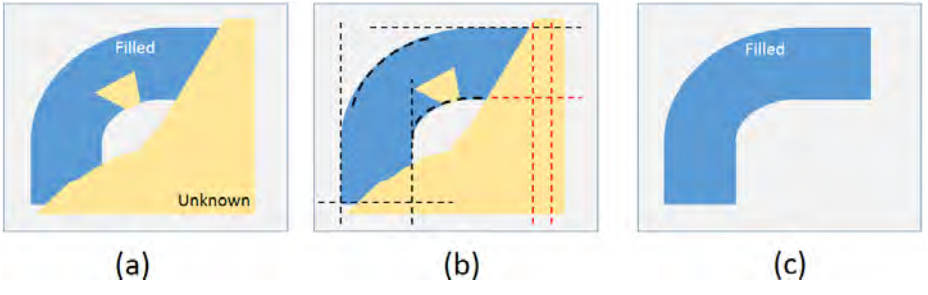


Fig. 2. Contour Completion Random Field: (a) A top-down view of a partially occluded plane (b) We detect lines and parabolas along the contour of the known pixels (stippled black lines), and hallucinate parallel lines (in red) (c) We apply CCRF inference to extend the plane.

[3] We use 400 angular bins for θ and evenly spaced bins for ρ 1 unit apart. The minimum number of votes allowed was set to 10.

4.4 Hierarchical Edge Groups

While using detected lines or curves may encourage the correct surface boundaries to be inferred in many cases, in others, there is no evidence present in the image of how a shape should be completed. For example see the right side of the shape in figure 2 and the synthetic examples in figure 3(b). Motivated by the gestalt laws of perceptual grouping, we attempt to add edge groups whose use in completion would help provide for shapes that exhibited simple closure and symmetry. More specifically, for each observed line detected, we add addition parallel edge groups on the occluded side of the shape.

It is clear that defining edge groups that completely cover another edge group would lead to double counting. To prevent this, we modify the formulation to ensure that only one group from each hierarchy of edge groups is counted. To be precise, our CCRF model allows edge groups to be organized hierarchically so that a set of possible edges have a parent and only a single child per parent may be active. This formulation is formalised as:

$$E(\mathbf{x}) = \sum_{i \in \mathcal{S}} \phi_i(x_i) + \sum_{g \in \mathcal{G}} \min_{k \in c(g)} \Psi_k(\mathbf{x}) \tag{4}$$

where $c(g)$ denotes the set of child edge groups for each parent g.

To summarize, our edge groups are obtained by fitting lines and parabolas to the input image thus encouraging transitions that are consistent with these edges. As indicated in Equation 4, not all edge groups can be active simultaneously and in particular, any line used to hallucinate a series of edges is considered the parent to its child hallucinated lines. Consequently, we constrain only a single hallucinated line to be active (at most) at a time.

4.5 Inference with Hierarchical Edge Groups

Inference under higher order potentials defined over edges groups was recently shown to be NP-hard even for binary random variables by Jegelka *et al.* [19]. They proposed a special purpose iterative algorithm for performing approximate inference in this model. Later, Kohli *et al.* [20] proposed an approximate method that could deal with multi-label variables. Their method transformed edge group potentials into a sum of pairwise potentials with the addition of auxiliary variables that allowed the use of standard inference method like graph cuts. However, both these algorithms are unsuitable for CCRF because of the special hierarchical structure defined over our edge groups.

Inspired from [20], we transformed the higher-order curve completion potential (3) to the following pairwise form:

$$\Psi_g^p(\mathbf{x}) = T + \min_{h_g, \mathbf{z}} \Big\{ \sum_{ij \in \mathcal{E}_g} \theta_{ij}((x_i + x_j - 2z_{ij})h_g - 2(x_i + x_j)z_{ij} + 4z_{ij}) - Th_g \Big\}. \tag{5}$$

where h_g is the binary auxiliary corresponding to the group g, and $z_{ij}, \forall ij \in \mathcal{E}_g$ are binary auxiliary variables corresponding to the edges that constitute the edge

group g. However, this transformation deviates from the energy of the hierarchical CCRF (equation 4) as it allows multiple edge groups in the hierarchy to be all active at once.

To enure that only one edge group is active in each edge group hierarchy, we introduce a series of constraints on the binary auxiliary variables corresponding to the edge groups. More formally, we minimize the following energy :

$$E(\mathbf{x}) = \sum_{i \in \mathcal{S}} \phi_i(x_i) + \sum_{g \in \mathcal{G}} \Psi_g^p(\mathbf{x}) \qquad \text{s.t.} \forall g, \qquad \sum_{k \in c(g)} h_k \leq 1 \tag{6}$$

where $c(g)$ denotes the set of child edge groups for each parent edge group g. The minimum energy configuration of this formulation is equivalent to that of hierarchical CCRF (equation 4).

In order to find the MAP solution, we now need to minimize the constrained pairwise energy function (equation 6). We observe that on fixing the values of the group auxiliary variable (h's) the resulting energy becomes unconstrained and submodular, and thus, can be minimized using graph cuts. We use this observation to do inference by exhaustively searching over the space of edge group auxiliary variables and minimizes the rest of the energy using graph cuts. However, we can make the algorithm even more efficient by not allow the activity of a child edge group to be explored if its parent is not already active. In other words, we start by exhaustively searching over the auxiliary variables of the parent edge groups (at the top of the hierarchy), and if a group variable is found to be active, we check if its child variables can be made active instead of it.

4.6 Inferring Scene Boundaries

To extend and fill the scene boundaries, we begin by projecting the free space of the input TSDF and the Wall planes (predicted by our classifier) onto the floor plane. Given a 2D point cloud induced by these projections, we discretize the points to form a projection image illustrated by figure 2 where each pixel y_i takes on the value of free space, wall or unknown. To infer the full scene layout, we apply the CCRF (Equation 2) to infer the values of the unknown pixels. In this case, we consider free space to be the area to be expanded ($y_i = 1$) and the walls to be the surrounding area to avoid being filled ($y_i = 0$). We first detect the lines and curves of the walls to create a series of edge groups. Next, we set $\phi_i(x_i = 1) = \infty$ if $y_i = 0$ and $\phi_i(x_i = 0) = \infty$ if $y_i = 1$. Finally, we add a slight bias [6] to assigning free space $\phi_i(x_i = 0) = \epsilon$ [4].

4.7 Extending Planar Surfaces

Once the scene boundary has been completed, we infer the full extent of internal planar surfaces. For each internal plane, we project the TSDF onto the detected 2D plane as follows. First we find a coordinate basis for the plane using PCA

[4] ϵ=1e-6

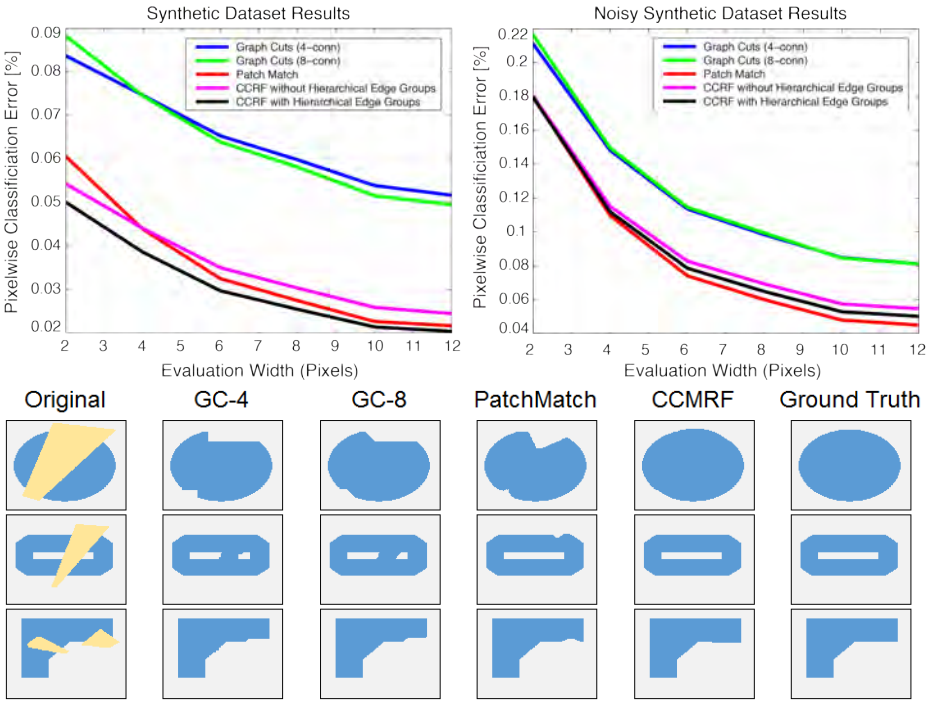


Fig. 3. (Top) Pixel-wise classification error on our synthetic dataset. The plot demonstrates performance as a function of the width of the evaluation region. (Bottom) Some examples from our dataset of synthetic images showing classification results for occluded pixels (yellow) using four methods.

and estimate the major and and minor axes of the plane, M and N, respectively. Next, we create an image of size $2N+1 \times 2M+1$ where the center pixel of the image corresponds to the centroid of the plane. We sample a grid along the plane basis of size $2N+1 \times 2M+1$ where the TSDF values sampled in each grid location are used to assign each of the image's pixels. If the sampled TSDF value is occupied, y_i is set to 1, if its free space y_i is set to 0 and if its unknown, y_i is set to -1. In practice, we also sample several voxels *away* from the plane (along the surface normal direction). This heuristic has the effect of reducing the effects of sensor noise and error from plane fitting.

Once Y has been created, we detect all lines and parabolas in the image and hallucinate the necessary lines to create our edge groups. Next, we assign the local potentials in the same manner as described in Section 4.6.

5 Augmenting the Original Volume

The result of our scene completion is a water-tight scene boundary, and extended interior planes. As the final step in our pipeline we augment the original TSDF we imported. For the scene boundary we simplify the resutling polyline representing

the boundary, and sample points along this boundary from floor to ceiling height. For the interior planes we sample points in the extended parts of the planes. For each sampled point (sampled densely as required, in our case γ) we traverse a bresenham-line in the volume from the voxel closest to the point, and in two directions, its normal and the inverse to its normal. For each encountered voxel, we update the TSDF value with the distance to the surface. If the dimensions of the original volume do not suffice to hold the new scene boundaries, we create a new larger volume and copy the original TSDF to it, before augmenting it.

The augmented TSDF, in the originating surface reconstruction pipeline, is continuously updated with new evidence (e.g. as the user moves). Augmented areas are phased out as the voxels which they filled become known.

6 Experiments

We evaluate our method in several ways. First, we demonstrate the effectiveness of the CCRF model by comparing it to several baselines for performing in-painting of binary images. Second, we compare our results to a recently introduced method [12] that performs extent-reasoning. Third, we demonstrate qualitative results on a newly collected set of indoor scenes.

6.1 Synthetic validation

We generated a dataset of synthetic images inspired by shapes of planar surfaces commonly found in indoor scenes. The dataset contained 16 prototypical shapes meant to resemble objects like desks, conference tables, beds, kitchen counters, etc. The set of 16 protypes were first divided into training and test sets. Each of the 8 training and testing prototypes were then randomly rotated and occluded 10 times resulting in 80 total training and 80 test images.

Since we are primarily concerned with how these methods perform in predicting the boundaries of the input images, we use the evaluation metric of [21] in which we only evaluate the correctness of pixels immediately near the boundary areas. We computed evaluation scores for various widths of the evaluation region. Quantitative results can be found in Figure 3. We compare against several baseline methods that are commonly used for in-painting binary images. These include both 4 connected and 8 connected graph cuts and a non-parametric patch-matching algorithm for in-painting. The ideal parameters for each algorithm were fine tuned on the training set and then applied to the test set.

6.2 Single Frame Scene Completion

We compare our approach to the work of Ruiqi and Hoiem [12] to demonstrate its applicability to single frame scene completion. While we produce a binary label indicating the voxels that are occupied, [12] output a heat map. Therefore, for each frame of the NYU V2 dataset [18] we computed the filled voxels using our regular pipeline. Since KinectFusion was not designed for single image inputs, a small number of images failed to be fused and were ignored during evaluation.

For the rest we used the metric of [12] and report the accuracy using the same false positive rate as in our method.

[12] achieved 62.6% accuracy compared to our 60.6%. This demonstrates that while our method was not meant for single frame completion, and does not use RGB data, it achieves comparable results. For visual comparisons see figure 4.

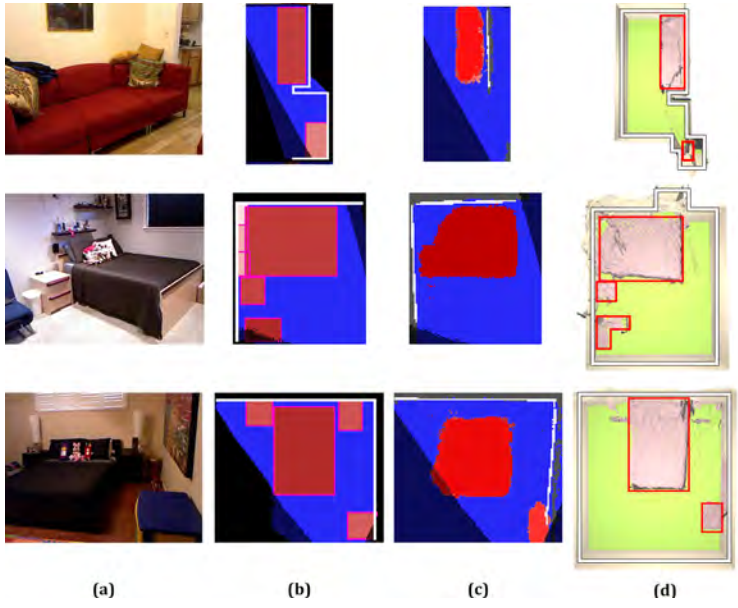


Fig. 4. Comparison with [12]: (a) Frames from the NYU2 dataset (b) Ground truth support extent (c) Predicted extent from [12] (d) A top down view of our completed scene where inferred scene boundaries are highlighted in white and extended interior objects are highlighted in red.

6.3 Qualitative Analysis

Using a Kinect Sensor attached to a notebook computer, we captured more than 100 scenes. Each scene is a collection of color and depth frames ranging from hundreds to thousands of frames. Each of these sequences was input into the KinectFusion pipeline, where for a large number of scenes we performed multiple reconstructions, sharing a starting frame, but varying in number of frames processed. This gave us a baseline for a qualitative and progressive evaluation of our results.

For each such scene we were able to evaluate the original augmented and original reconstruction at each step. As more evidence is revealed, extended planes are replaced with their physical counterparts, and the scene boundaries are

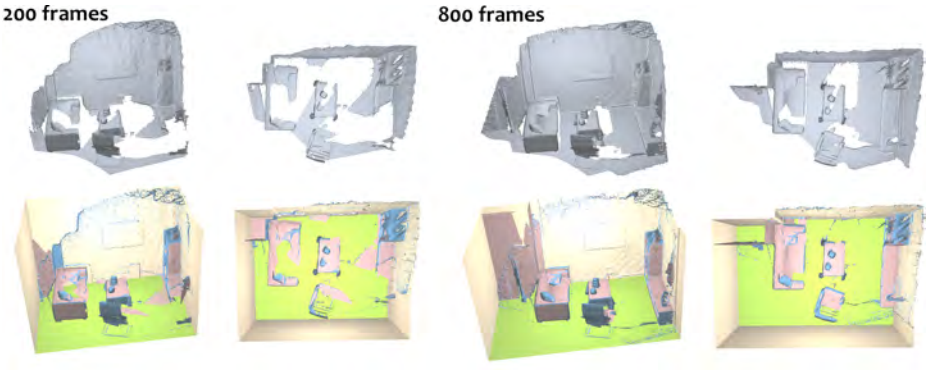
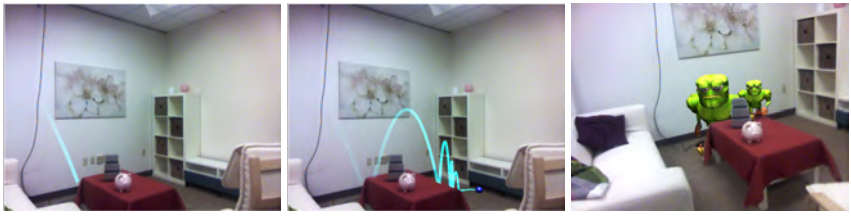


Fig. 5. Progressive surface reconstruction and scene completion. After 200 frames large parts of the room are occluded, and filled in by the completion model. After 800 previously occluded areas are observed and replace the completion, and overall the structure of the room is more accurate. For a more detailed figure please see supplementary material.

updated to reflect reality. A sample progression can be seen in figure 5. Additional results can be seen in figure 7(top row) and the supplemental material.

We implemented a complete AR system in the Unity 3D framework, in which a user is able to navigate a captured scene, place virtual objects on horizontal supporting planes, and throw balls which bounce around the scene. As seen in figure 6, with scene completion, we are able to place figures on occluded planes, and bounce balls realistically off completed surfaces. Video captured results can be seen in the supplemental materials.



(a) (b) (c)

Fig. 6. We created an augmented reality application to demonstrate our results. (a) a ball bouncing on an occluded floor falls through (b) a ball bouncing on the floor in a completed scene bounces realistically (c) Virtual objects may be placed on any surface, including occluded planes in the completed scene. The scene shown is the same as in figure 5. *Video clips of this application can be seen in the supplemental material.*

7 Discussion and Future Work

We have presented a complete and novel pipeline for surface reconstruction augmentation via scene completion. Our novel contour completion MRF algorithm is able to extend occluded surfaces in a believable manner, paving way for enhanced AR applications. In this work we have focused on extending planar surfaces and estimating scene boundaries, however we recognize the value in augmenting volumetric (non-planar) objects and intend to address it in future work. Moreover, in this work we do not rely on previously seen objects, nor on repetition in the scene, both of which we intend to employ in future work. Our current pipeline is implemented in Matlab and does not achieve online performance (processing a volume takes up to a minute). In the future we intend to integrate our pipeline directly into KinectFusion or an equivalent system.

Failure cases may occur in different stages along our pipeline. In some instances we fail to detect or correctly classify planes. Sometimes the noise level is too high, hampering the CCRF from fitting good contours. In some cases we receive faulty volumetric information from the Kinect sensor, mostly due to reflectance and transparencies in the scene. Some examples can be seen in figure 7 bottom row.

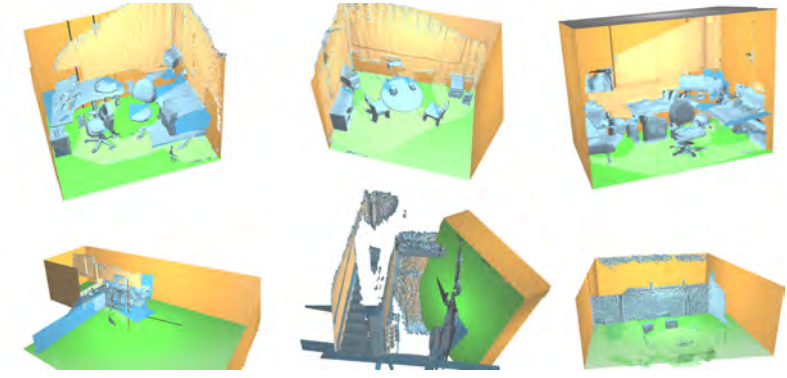


Fig. 7. Example scenes: darker colors represent completed areas, hues signify plane classification. The top row contains mostly successful completions. The bottom row contains failure cases, on the left the room is huge due to many reflective surfaces, the middle image is of a staircase, vertical in nature, which leads to incorrect plane classifications. And on the right a high ceiling and windows cause misclassification of the floor plane.

References

1. Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D.: Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In: ISER. Volume 20. (2010) 22–25

2. Newcombe, R.A., Davison, A.J., Izadi, S., Kohli, P., Hilliges, O., Shotton, J., Molyneaux, D., Hodges, S., Kim, D., Fitzgibbon, A.: Kinectfusion: Real-time dense surface mapping and tracking. In: ISMAR. (2011)
3. Qualcomm: Vuforia. Website (2011) https://www.vuforia.com/.
4. Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: CGI. (1996) 303–312
5. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. In: Proceedings of the 14th annual conference on Computer graphics and interactive techniques. SIGGRAPH (1987)
6. Furukawa, Y., Curless, B., Seitz, S.M., Szeliski, R.: Reconstructing building interiors from images. In: ICCV. (2009)
7. Furukawa, Y., Curless, B., Seitz, S., Szeliski, R.: Manhattan-world stereo. CVPR (2009)
8. Salas-Moreno, R.F., Newcombe, R.A., Strasdat, H., Kelly, P.H., Davison, A.J.: Slam++: Simultaneous localisation and mapping at the level of objects. In: CVPR. (June 2013)
9. Barinova, O., Lempitsky, V., Tretiak, E., Kohli, P.: Geometric image parsing in man-made environments. In: ECCV. (2010)
10. Hoiem, D., Hedau, V., Forsyth, D.: Recovering free space of indoor scenes from a single image. (2012)
11. Guo, R., Hoiem, D.: Beyond the line of sight: Labeling the underlying surfaces. In: ECCV. (2012)
12. Guo, R., Hoiem, D.: Support surface prediction in indoor scenes. In: ICCV. (2013)
13. Kim, Y.M., Mitra, N.J., Yan, D.M., Guibas, L.: Acquiring 3d indoor environments with variability and repetition. ACM Transactions on Graphics **31**(6) (2012)
14. soo Kim, B., Kohli, P., Saverese, S.: 3d scene understanding by voxel-crf. In: ICCV. (2013)
15. Zheng, B., Zhao, Y., Yu, J.C., Ikeuchi, K., Zhu, S.C.: Beyond point clouds: Scene understanding by reasoning geometry and physics. In: CVPR. (2013)
16. Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functionsi. Mathematical Programming **14**(1) (1978) 265–294
17. Duda, R.O., Hart, P.E.: Use of the hough transformation to detect lines and curves in pictures. Communications of the ACM **15**(1) (1972) 11–15
18. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: ECCV. (2012)
19. Jegelka, S., Bilmes, J.: Submodularity beyond submodular energies: Coupling edges in graph cuts. In: CVPR. (2011)
20. Kohli, P., Osokin, A., Jegelka, S.: A principled deep random field model for image segmentation. In: CVPR. (2013)
21. Kohli, P., Torr, P.H., et al.: Robust higher order potentials for enforcing label consistency. IJCV **82**(3) (2009)