3D Reconstruction Using an n-Layer Heightmap

David Gallup[1], Marc Pollefeys[2], and Jan-Michael Frahm[1]

[1] Department of Computer Science, University of North Carolina
{gallup,jmf}@cs.unc.edu
[2] Department of Computer Science, ETH Zurich
marc.pollefeys@inf.ethz.ch

Abstract. We present a novel method for 3D reconstruction of urban scenes extending a recently introduced heightmap model. Our model has several advantages for 3D modeling of urban scenes: it naturally enforces vertical surfaces, has no holes, leads to an efficient algorithm, and is compact in size. We remove the major limitation of the heightmap by enabling modeling of overhanging structures. Our method is based on an an n-layer heightmap with each layer representing a surface between full and empty space. The configuration of layers can be computed optimally using a dynamic programming method. Our cost function is derived from probabilistic occupancy, and incorporates the Bayesian Information Criterion (BIC) for selecting the number of layers to use at each pixel. 3D surface models are extracted from the heightmap. We show results from a variety of datasets including Internet photo collections. Our method runs on the GPU and the complete system processes video at 13 Hz.

1 Introduction

Automatic large-scale 3D reconstruction of urban environments is a very active research topic with broad applications including 3D maps like Google Earth and Microsoft Bing Maps, civil planning, and entertainment. Recent approaches have used LiDAR scans, video, or photographs, acquired either from ground, aerial, or satellite platforms [1,2,3,4,5]. In this work, we focus on reconstructions from street-level video, which has higher resolution than aerial data, and video cameras are significantly less expensive than active sensors like LiDAR.

To process in a reasonable time, computational efficiency must be considered when modeling wide-area urban environments such as entire cities, since millions of frames of video are required for even a small town [3]. Even if a (cloud) computing cluster is used, efficiency is of great concern since usage of such systems is billed according to processing time. In addition to computational efficiency, the models need to be compact in order to efficiently store, transmit, and render them.

Gallup et al. [6] introduced a method, which uses a heightmap representation to model urban scenes. See Figure 1 for an example. The method takes depthmaps as input and fits a heightmap to a volume of occupancy votes. In contrast to other volumetric methods [7], the heightmap model has several advantages. First, it enforces that walls and facades are strictly flat and vertical,

M. Goesele et al. (Eds.): DAGM 2010, LNCS 6376, pp. 1–10, 2010.

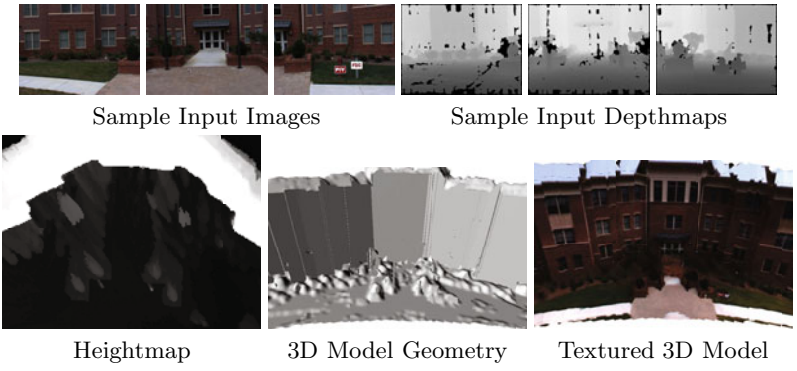

Fig. 1. Our system uses a heightmap model for 3D reconstruction. Images courtesy of Gallup et al.[6].

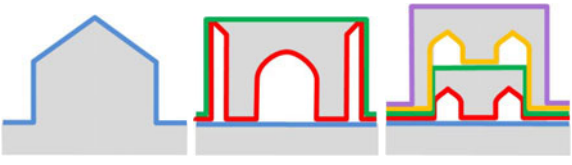

Fig. 2. Examples of n-layer heightmaps

since they appear as discontinuities in the heightmap. Second, the heightmap represents a continuous surface and does not allow for holes. Third, because the height estimate is supported by the entire vertical column, no regularization is necessary, leading to a highly parallel and efficient computation. Fourth, heightmaps can be stored and transmitted efficiently using depthmap coding algorithms.

However, the major limitation of the method is the inability to model overhanging structures. Thus awnings, eves, balconies, doorways, and arches are either filled in or missed entirely. While some loss of detail is to be expected in exchange for a robust and compact representation, this is a major weakness.

In this paper we adopt the heightmap approach and improve upon it in the following ways: First we introduce a multi-layer representation to handle overhanging structures. Second, the cost function for heightmap estimation is derived from probability. We address the overhanging structure problem by extending the method of [6] to an n-layer heightmap. Each layer represents a surface between full and empty space. Some examples are shown in Figure 2. The positions of the n layers at each heightmap pixel can be computed optimally using dynamic programming. We also include the Bayesian Information Criterion (BIC) as a model selection penalty to use additional layers only when necessary. In [6], the cost function for heightmap estimation was defined in an ad-hoc manner. We show how this cost function can be derived from probabilities. This derivation also allows us to incorporate the BIC in a principled way.

As in [6], our method also runs on the GPU, and the complete system can process video at 13 Hz. We have demonstrate our approach on several challenging street-side video sequences. Results show a clear improvement over [6], particularly on overhanging structures and trees.

Another data source for urban 3D reconstruction is images downloaded from photo sharing websites such as Flickr. In this case data acquisition is free but is subject to the interests of the website community, and thus datasets are usually limited to popular tourist locations. Camera poses can be computed using techniques such as Snavely et al. [8] and the more recent methods of [9,10]. Dense stereo and surface modeling were achieved by Goesele et al. [11] and recently by Furukawa et al. [12]. We apply our extended heightmap approach to 3D reconstruction from community photo collections as well. Our approach is much simpler and faster, and yet results are surprisingly good.

2 Related Work

Recent approaches employ simplified geometries to gain robustness [13,1,14,15]. Cornelis et al.[13] produce compact 3D street models using a ruled surface model. Similar to the heightmap model, this assumes that walls and facades are vertical. Furukawa et al.[14] presented a Manhattan-world model for stereo, where all surfaces have one of three orthogonal surface normals. Sinha et al.[15] employ a general piecewise-planar model, and Gallup et al.[6] uses a more general piecewise-planar model that can also handle non-planar objects. Our approach uses a simplified geometry and is far more general than [13,15], and more efficient and compact than [14,6]. It effectively models buildings and terrain, but also naturally models cars, pedestrians, lamp posts, bushes, and trees.

In our approach we use the probability occupancy grid of the scene from the robotics literature [16,17]. The occupancy of each voxel is computed by bayesian inference, and our derivation is similar to that of Guan et al.[18]. We model the measurement distribution as a combination of normal and uniform distrubutions in order to better handle outliers. Robustness to outliers is critical since our input measurements are stereo depthmaps.

Dense 3D reconstruction for photo collections has first been explored by Goesele et al.[11] and by Furukawa et al.[19]. Images on the web come from a variety of uncontrolled settings, which violate many of the assumptions of stereo such as brightness constancy. Goesele et al. and Furukawa et al. take great care to select only the most compatible images, starting from points of high certainty and growing outward. Our approach on the other hand relies on the robustness of the heightmap model and results in a much simpler and faster algorithm.

Merrell et al. [20] proposed a depthmap fusion from video employing the temporal redundancy of the depth computed for each frame. It obtains a consensus surface by enforcing visibility constraints. The proposed heightmap fusion in contrast does not require a confidence measure due to the benefits of the vertical column regularization.

3 Method

The proposed n-layer heightmap generalizes the single layer heightmap. A single layer heightmap defines a surface, which is the transition from occupied space to empty space. In an n layer heightmap, each layer defines a transition from full to empty or vice versa. The number of layers needed to reconstruct a scene can be determined with a vertical line test. For any vertical line, the number of surfaces that the line intersects is the number of layers in the scene. In our approach, the user must give the number of layers beforehand, although model selection may determine that fewer layers are sufficient.

The input to our method is a set of images with corresponding camera poses and their depthmaps. The depth measurements from each camera are used to determine the occupancy likelihood of each point in space, and an n-layer heightmap is fit. Using a heightmap for ground-based measurements has the advantage that the estimated parameter, height, is perpendicular to the dominant direction of measurement noise. This is ideal for urban reconstruction where vertical walls are of particular interest.

We will now present our novel method for reconstructing scenes using an n-layer heightmap. This method consists of the following steps:

- Layout the volume of interest.
- Construct the probabilistic occupancy grid over the volume.
- Compute the n-layer heightmap.
- Extract mesh and generate texture maps.

The volume of interest for heightmap computation is defined by its position, orientation, size, and resolution. Heightmap computation assumes the vertical direction is known, which can be extracted from the images itself. Besides that constraint, the volume of interest can be defined arbitrarily. For processing large datasets like video of an entire street, it makes sense to define several volumes of interest and process them independently. For video, a frame is chosen as reference, and the volume of interest is defined with respect to the camera's coordinate system for that frame. Reference frames are chosen at irregular intervals where the spacing is determined by overlap with the previous volume. Our video data also contains GPS measurements, so the camera path is geo-registered, and the vertical direction is known. For photo collections, the vertical direction can be found using a heuristic derived from photography practices. Most photographers will tilt the camera, but not allow it to roll. In other words, the x axis of the camera stays perpendicular to gravity. This heuristic can be used to compute the vertical direction as a homogeneous least squares problem as shown in [21]. The size and resolution of the volume are given as user parameters.

The next step is to compute the probabilistic occupancy grid over the volume of interest. Since the heightmap layers will be computed independently for each vertical column of the volume, the occupancy grid does not need to be fully stored. Only each column must be stored temporarily, which keeps the memory requirement low. We will first derive the occupancy likelihood for each voxel independently. Voxel occupancy is in fact not independent since it must obey the

layer constraint, and we will later show how to compute the layers for a column of voxels using dynamic programming. The variables used in our derivation are summarized as follows:

- O_p: a binary random variable representing the occupancy of voxel p.
- $Z_p = Z_1 \dots Z_k$: depth measurements along rays intersecting p from cameras $1 \dots$ k.
- z_{min}, z_{max}: depth range of the scene.
- σ: depth measurement uncertainty (standard deviation).
- S: depth of surface hypothesis.
- $L_x = l_1 \dots l_n$: configuration of layers at point x in the heightmap. l_i is the vertical position of layer i.

For simplicity we have assumed that all depth measurements have the same uncertainty σ although this is not a requirement.

We will now derive the likelihood for O_p. We will drop the subscript p until multiple voxels are considered for dynamic programming.

$$P(O|Z) \propto P(Z|O)P(O) \tag{1}$$

$$P(Z|O) = \prod_{i=1\dots k} P(Z_i|O) \tag{2}$$

Equation 2 states our assumption that the measurements are independent. We use the occupancy prior $P(O)$ to slightly bias the volume to be empty above the camera center and full below. This helps to prevent rooftops extending into empty space since the cameras don't observe them from the ground.

To determine $P(Z_i|O)$ we will follow [18] and introduce a helper variable S which is a candidate surface along the measurement ray. The depth measurement can then be formulated with respect to S.

$$P(Z_i|O) = \int_{z_{min}}^{z_{max}} P(Z_i|S,O)P(S|O)dS \tag{3}$$

$$P(Z_i|S,O) = P(Z_i|S) = \begin{cases} \mathcal{N}(S,\sigma)|_{Z_i} & \text{if inliner} \\ \mathcal{U}(z_{min}, z_{max})|_{Z_i} & \text{if outlier} \end{cases} \tag{4}$$

$$= \rho\mathcal{N}(S,\sigma)|_{Z_i} + (1-\rho)\mathcal{U}(z_{min}, z_{max})|_{Z_i} \tag{5}$$

The measurement model is a mixture of a normal distribution $\mathcal{N}$ and uniform distribution $\mathcal{U}$ to handle outliers. $\mathcal{N}|_Z$ is the disribution's density function evaluated at Z. ρ is the inlier ratio, which is a given parameter. $P(S|O)$ is the surface formation model defined as follows where $\epsilon \to 0$ and z_p is the depth of the voxel.

$$P(S|O) = \begin{cases} 1/(z_{max} - z_{min}) & \text{if } S < z_p - \epsilon \\ (1 - z_p/(z_{max} - z_{min})/\epsilon & \text{if } z_p - \epsilon \leq S \leq z_p \\ 0 & \text{if } S > z_p \end{cases} \tag{6}$$

This model states that the surface must be in front of the occupied voxel, but not behind it. We will also need the measurement likelihood given that the voxel is empty, which we will denote by $\neg O$. The derivation is the same, replacing O with $\neg O$, except the surface formation model is

$$P(S|\neg O) = 1/(z_{max} - z_{min}). \tag{7}$$

We will now define our n-layer model and show how to recover it with dynamic programming. We will derive the likelihood of L_x which is the layer configuration at pixel x in the heightmap. This pixel contains a vertical column of voxels, which we will denote as O_i where i is the height of the voxel ranging from 0 to m.

$$P(L|Z) \propto P(Z|L)P(L) \tag{8}$$

$$P(Z|L) = \prod_{i=0}^{l_1-1} P(Z|O_i) \prod_{i=l_1}^{l_2-1} P(Z|\neg O_i) \dots \prod_{i=l_n}^{m} P(Z|\neg O_i). \tag{9}$$

$$P(L) = \prod_{i=0}^{l_1-1} P(O_i) \prod_{i=l_1}^{l_2-1} P(\neg O_i) \dots \prod_{i=l_n}^{m} P(\neg O_i). \tag{10}$$

Note that the measurement likelihoods alternate between the full condition $P(Z|O_i)$ and the empty condition $P(Z|\neg O_i)$ as dictated by the layer constraint. Also note that the number of layers is assumed to be odd, giving the final product the empty condition. This is true for outdoor urban scenes. For indoor scenes, an even number of layers could be used.

We will now define our cost function C by taking the negative log-likelihood of $P(L|Z)$, which will simplify the dynamic programming solution.

$$C = -ln\ P(Z|L)P(L) = -\sum_{i=0}^{l_1-1} (ln\ P(Z|O_i) + ln\ P(O_i)) \tag{11}$$

$$-\sum_{i=l_1}^{l_2-1} (ln\ P(Z|\neg O_i) + ln\ P(\neg O_i)) \dots \tag{12}$$

To simplify the sums over the layers we will define the following:

$$I_a^b = -\sum_{i=a}^{b} (ln\ P(Z|O_i) + ln\ P(O_i)) \tag{13}$$

$$\bar{I}_a^b = -\sum_{i=a}^{b} (ln\ P(Z|\neg O_i) + ln\ P(\neg O_i)). \tag{14}$$

The sums I_0^b (resp. $\bar{I}$) for all b can be precomputed making it easy to compute $I_a^b = I_0^b - I_0^{a-1}$ (resp. $\bar{I}$).

We can now write our cost function recursively in terms of C_k which is the cost only up to layer k.

$$C_k(l) = \begin{cases} I^l_{l'} + C_{k-1}(l') & \text{if } odd(k) \\ \bar{I}^l_{l'} + C_{k-1}(l') & \text{if } even(k) \end{cases} \tag{15}$$

$$l' = \arg\min_{l' \leq l} C_{k-1}(l') \tag{16}$$

$$C_0(l) = 0 \tag{17}$$

The original cost function is then $C = C_n(m)$ where n is the number of layers and m is the number of voxels in the vertical column.

The layer configuration that minimizes C can be computed with dynamic programming. In order for this to be true, the problem must exhibit *optimal substructure* and *overlapping subproblems* [22]. The problem has optimal substructure because of the independence between non-adjacent layers, i.e. an optimal configuration of layers $1 \ldots i-1$ will still be optimal regardless of the position of layer i. (As in C_k, we consider only the voxels below the layer.) The overlapping subproblems occur since computing the optimal position of any layer greater than i requires computing the optimial configuration of layers $1 \ldots i$. Therefore, the optimal configuration can be solved with dynamic programming. The recursive formulation in Equation 19 lends easily to the table method, and the solution can extracted by backtracking.

Many parts of the heightmap will not need all n layers. The extra layers will be free to fit the noise in the measurements. To avoid this, we incorporate the Bayesian Information Criterion (BIC).

$$C_{BIC} = -ln\ P(Z|L)P(L) + \frac{1}{2} n\ ln\ |Z_x| \tag{18}$$

$|Z_x|$ is the number of measurements interacting with the heightmap pixel x. The first part of the equation is exactly C and the second part adds a penalty of $ln\ |Z_x|$ for every layer in the model. We can add this penalty into our recursive formulation by adding $ln\ |Z_x|$ at each layer unless the layer position is the same as the preceding layer.

$$C^{BIC}_k(l) = \begin{cases} I^l_{l'} + C_{k-1}(l') + T(l \neq l')\frac{1}{2} ln\ |Z_x| & \text{if } odd(k) \\ \bar{I}^l_{l'} + C_{k-1}(l') + T(l \neq l')\frac{1}{2} ln\ |Z_x| & \text{if } even(k) \end{cases} \tag{19}$$

Thus model selection is performed by prefering layers to *collapse* unless there is sufficient evidence to support them. The table required to solve the problem is of size $m \times n$, and the sum variables are of size m. Therefore the algorithm takes $O(mn)$ time and space per heightmap pixel, and the whole heightmap takes $O(whmn)$ time and $O(wh + mn)$ space.

4 Results

We have tested our n-layer heightmap method on street-level video datasets and photo collections downloaded from the web. For the video datasets, the camera

Fig. 3. Original photos and depthmaps computed from Internet photo collections

poses and depthmaps were computed with the real-time system of Pollefeys et al.[3]. To compute the camera poses for the photo collections, we used the method of Li et al.[9]. The output of their approach also gives a clustering of the images which can be used to select compatible views for stereo. We computed a depthmap for each photograph by selecting the 20 views in the same cluster with the most matched and triangulated SIFT points in common. Stereo is performed on the GPU using a simple NCC planesweep. Results are shown in Figure 3.

From these inputs we used our n-layer heightmap system to obtain a 3D reconstruction in the form of a texture-mapped 3D polygonal mesh. Texture mapping the mesh is a non-trivial problem, however, we did not focus on this in our method. We have used a simple method to reconstruct the appearance at each point on the surface. Each point is projected into all cameras, a 3-channel intensity histogram is constructed. The histogram votes are weighted by a guassian function of the difference between the measured depth and the heightmap model's depth, which helps to remove the influence of occluders. The final color is the per-channel median and is easily obtained from the histograms.

Figure 4 shows the improvement gained by using multiple layers in the heightmap. Overhanging structures are recovered while the clean and compact nature of the reconstruction is preserved. Figures 5 show the results of the reconstructions from video. Figures 6 show the results of the reconstructions from photo collections.

Our system can process video at 13.33 Hz. Computing a 3-layer 100x100 heightmap with 100 height levels from 48 depthmaps takes only 69 ms to on the GPU. The other steps are not as fast as we did not focus as much on optimizing them. Converting the heightmap into a mesh takes 609 ms, and generating texture maps takes 1.57 seconds. The total time for processing a heightmap is 2.25 seconds. However, heightmaps only need to be computed about every 30 frames of video. (All frames are used for depthmaps.) Therefore our system can process video at 13.33 frames per second. Reconstructing photo collections is more

1 Layer 3 Layer 1 Layer 3 Layer

Fig. 4. 1-layer and 3-layer reconstructions

Fig. 5. 3D reconstructions from video

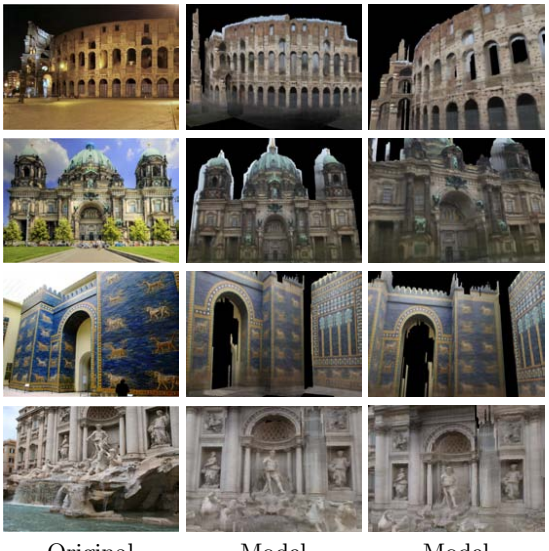

Fig. 6. 3D reconstructions from internet photo collections

challenging. Each scene takes 20-30 minutes, and most of that time is spent computing NCC stereo.

5 Conclusion

We proposed a novel n-layer heightmap depthmap fusion providing a natural way to enforce vertical facades while providing advantageous structure separation. The main advantage of the proposed approach is the generality of the modeled geometry. The regularization along the vertical direction allows the heightmap fusion to effectively suppress depth estimation noise. Our fusion is computationally efficient providing real-time computation. We demonstrated the proposed method on several challenging datasets downloaded from the Internet.

References

1. Cornelis, N., Leibe, B., Cornelis, K., Gool, L.V.: 3d urban scene modeling integrating recognition and reconstruction. IJCV (2008)
2. Früh, C., Jain, S., Zakohr, A.: Data processing algorithms for generating textured 3d building facade meshes from laser scans and camera images. IJCV (2005)
3. Pollefeys, M., et al.: Detailed real-time urban 3d reconstruction from video. Int. Journal of Computer Vision (IJCV) (2008)
4. Zebedin, L., Bauer, J., Karner, K., Bischof, H.: Fusion of feature- and area-based information for urban buildings modeling from aerial imagery. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part IV. LNCS, vol. 5305, pp. 873–886. Springer, Heidelberg (2008)
5. Xiao, J., Quan, L.: Image-based street-side city modeling. In: Siggraph Asia (2009)
6. Gallup, D., Frahm, J.M., Pollefeys, M.: Piecewise planar and non-planar stereo for urban scene reconstruction. In: CVPR (2010)
7. Zach, C., Pock, T., Bischof, H.: A globally optimal algorithm for robust tv-l1 range image integration. In: ICCV (2007)
8. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: Exploring photo collections in 3d. In: SIGGRAPH, pp. 835–846 (2006)
9. Li, X., Wu, C., Zach, C., Lazebnik, S., Frahm, J.M.: Modeling and recognition of landmark image collections using iconic scene graphs. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 427–440. Springer, Heidelberg (2008)
10. Agarwal, S., Snavely, N., Simon, I., Seitz, S.M., Szeliski, R.: Building Rome in a day. In: ICCV (2009)
11. Goesele, M., Snavely, N., Curless, B., Hoppe, H., Seitz, S.M.: Multi-view stereo for community photo collections. In: ICCV (2007)
12. Furukawa, Y., Curless, B., Seitz, S.M., Szeliski, R.: Towards internet-scale multi-view stereo. In: CVPR (2010)
13. Cornelis, N., Cornelis, K., Van Gool, L.: Fast compact city modeling for navigation pre-visualization. In: Computer Vision and Pattern Recognition (CVPR) (2006)
14. Furukawa, Y., Curless, B., Seitz, S.M., Szeliski, R.: Manhatten-world stereo. In: Proceedings IEEE CVPR (2009)
15. Sinha, S.N., Steedly, D., Szeliski, R.: Piecewise planar stereo for image-based rendering. In: Proceedings IEEE ICCV (2009)
16. Margaritis, D., Thrun, S.: Learning to locate an object in 3d space from a sequence of camera images. In: ICML (1998)
17. Pathak, K., Birk, A., Poppinga, J., Schwertfeger, S.: 3d forward sensor modeling and application to occupancy grid based sensor fusion. In: IROS (2007)
18. Guan, L., Franco, J.S., Pollefeys, M.: 3d object reconstruction with heterogeneous sensor data. In: 3DPVT (2008)
19. Furukawa, Y., Curless, B., Seitz, S.M., Szeliski, R.: Towards internet-scale multi-view stereo. In: CVPR (2010)
20. Merrell, P., Akbarzadeh, A., Wang, L., Mordohai, P., Frahm, J.M., Yang, R., Nister, D., Pollefeys, M.: Real-Time Visibility-Based Fusion of Depth Maps. In: Proceedings of International Conf. on Computer Vision (2007)
21. Szeliski, R.: Image alignment and stitching: A tutorial. Microsoft Research Technical Report (2005)
22. Cormen, T.H., Leisorson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms. The MIT Press, Cambridge (2001)