Fast Automatic Single-View 3-d Reconstruction of Urban Scenes

Olga Barinova[1],Vadim Konushin[2], Anton Yakubenko[1],
KeeChang Lee[3], Hwasup Lim[3], and Anton Konushin[1]

[1] Moscow State University, Department of Computational Mathematics and Cybernetics, Graphics & Media Lab
{obarinova,toh,ktosh}@graphics.cs.msu.ru
[2] The Keldysh Institute of Applied Mathematic Russian Academy of Sciences
vadim@graphics.cs.msu.ru
[3] Samsung Advanced Institute of Technology
{lkc.lee,hwasup.lim}@samsung.com

Abstract. We consider the problem of estimating 3-d structure from a single still image of an outdoor urban scene. Our goal is to efficiently create 3-d models which are visually pleasant. We chose an appropriate 3-d model structure and formulate the task of 3-d reconstruction as model fitting problem. Our 3-d models are composed of a number of vertical walls and a ground plane, where ground-vertical boundary is a continuous polyline. We achieve computational efficiency by special preprocessing together with stepwise search of 3-d model parameters dividing the problem into two smaller sub-problems on chain graphs. The use of Conditional Random Field models for both problems allows to various cues. We infer orientation of vertical walls of 3-d model vanishing points.

1 Introduction

Inferring the 3-d structure of pictured scene from single image is a challenging problem for current computer vision systems. Whereas understanding geometry of the scene for a human is quite easy, this problem remains open for a computer due to intrinsic ambiguity introduced by perspective projection. However, for man-made environments such as urban scenes where certain structural regularities are present, introducing additional constraints on the scene geometry might help to disambiguate the problem.

Most methods for estimation of 3-d models from a single image make strong assumptions about the scene geometry. For example, Delage, Lee and Ng [3][4] considered indoor images and classified the image into ground and vertical walls to produce a simple visually pleasing 3-d model. In [1][2] Hoiem, Efros and Hebert proposed to use similar constraints to model outdoor scenes. They classified image regions into geometric classes and then used classification results to create coarse pop-up type scene model composed of ground, walls and sky. Criminisi, Reid and Zisserman [5] determined affine structure of the image using vanishing points with application in 3-d graphical modeling.

D. Forsyth, P. Torr, and A. Zisserman (Eds.): ECCV 2008, Part II, LNCS 5303, pp. 100–113, 2008.

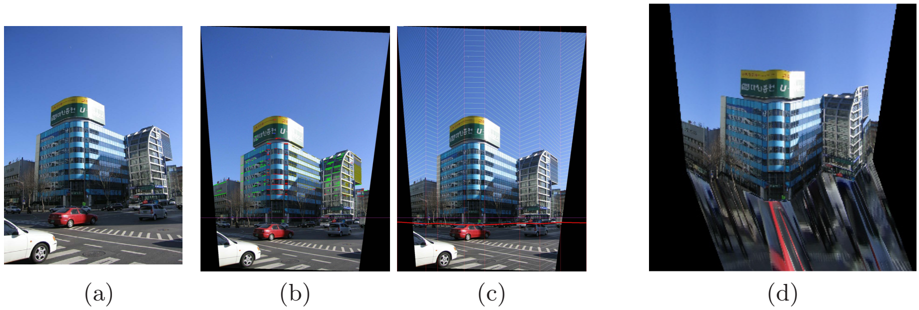

Fig. 1. Outline of the proposed algorithm (a) Source image (b) Preprocessed image: tilt correction, horizon detection and vanishing point estimation (c) Different positions of ground-vertical border along the vertical axis (d) Resulting model

In contrast, Saxena, Sun and Ng [7][8][9] made no explicit assumptions about the structure of the scene. They inferred both the 3-d orientation and location of the small planar regions in the image using a Markov Random Field by learning the relation between the image features and the location/orientation. MAP estimate in their model is efficiently performed by solving a linear program.

Our goal is to automatically create visually pleasant 3-d models. We choose a subset of the images (urban scenes) and fix an appropriate structure of 3-d model that allows to formulate the task of 3-d reconsruction as model fitting problem.

The structure of our 3-d models is as follows. We assume that the environment is composed of a flat ground with vertical walls where ground-vertical boundary is a continuous polyline (see Fig. 2). Each vertical wall of 3-d model corresponds to the building wall in the scene. Structural regularities of man-made environments allow tilt-correction of an image, horizon and vanishing point estimation, that provides information aboout scene geometry. Geometric structure of our 3-d models resembles the one used in [1][2] and [3][4], but in contrast to these works our 3-d model structure is rigidly fixed and can be inferred explicitly.

Assumption about continuity of ground-vertical boundary being realistic for most of urban scenes makes resulting estimates more stable and reduces the dimension of the search space. Configuration of ground-vertical boundary determines geometry of 3-d model. Given an image of urban scene, our algorithm automatically infers parameters of polyline and produces 3-d model. The question of detecting ground/vertical "occlusion boundary" automatically has also been looked at in [6]. In contrast to this work we search for ground-vertical boundary in explicit form.

Another focus of this work is the efficiency of 3-d reconstruction algorithm, which becomes particularly important when processing images of reasonable resolution. Using probabilistic framework we suggest stepwise search of ground-vertical boundary parameters. Model fitting problem is divided into two smaller sub-problems on chain graphs. We construct Conditional Random Field models incorporating various types of information (appearance, geometric properties

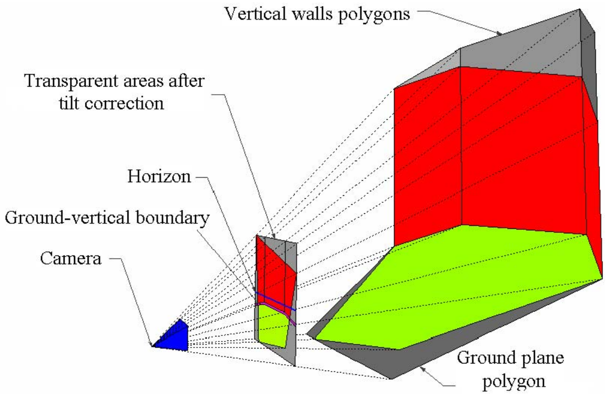

Fig. 2. 3-d model structure 3-d model is composed of a number of vertical walls and a ground plane

and context) for both problems. Inference in each model is highly efficient due to simplicity of graph structure. An outline of our algorithm is presented in Fig. 1.

The paper is structured as follows. In the next section we describe special preprocessing of an image which includes tilt-correction and vanishing points estimation. Section 3 explains how the high-level model for building-ground boundary is divided into two Conditional Random Field models. Sections 4 and 5 explain each model in details. Modeling procedure is described in Section 6. Experiments and conclusions are given in the final two sections.

2 Image Preprocessing

In this work image preprocessing is an essential part of the system. The outline of preprocessing is as follows. First, we apply Canny edge detection [10] to an image and then extract straight lines using a method described in [11]. Extracted lines are further used for tilt correction, horizon level estimation and vanishing points estimation. Preprocessing steps are demonstrated in Fig. 3.

2.1 Tilt Correction

In order to capture a building entirely, in one photo, a camera is usually pitched up. This leads to so called keystone effect (see Fig. 4). In keystoned images rectangles, like window frames, become trapezoids that are wider at the bottom (in case of pitching up). The vertical (with respect to the ground plane) lines become inclined. Non-zero camera roll angle additionally leads to the inclined horizon line.

The structure of our 3-d model requires building side borders to be vertical in the images. Given a tilted image, we need to compute a corresponding virtual

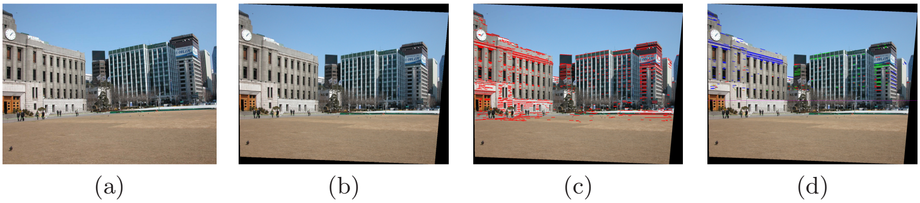

Fig. 3. Preprocessing scheme (a) Source image (b) Tilt corrected image (c) Extracted lines after filtering (d) Horizon estimation and vanishing point estimation. Lines, which refer to the same vanishing point, are shown with the same colors

view with zero pitch and roll angles. For this task we use automatic tilt correction algorithm similar to the one, described in [12]. It uses the extracted line segments that correspond to a vertical vanishing point and searches for the best rotational homography, which transforms them into vertical lines. Due to the abundant presence of vertical lines in man-made environments this algorithm is quite robust. An example of tilt correction result is demonstrated in Fig. 3(b). The same homography transform is then applied to all straight line segments.

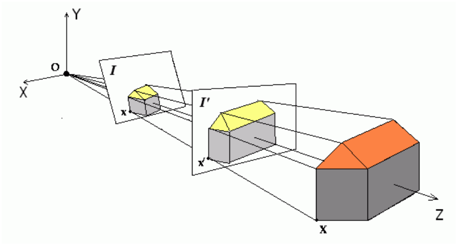

Fig. 4. Camera pitch and roll correction I' - a virtual view corresponding to a source image plane I

2.2 Horizon Level and Vanishing Points Estimation

Horizon level estimation. Due to perspective rules all vanishing points lie at the same straight line, which is called horizon. Our algorithm for horizon level estimation is based on the idea that horizon is transformed into level line after tilt correction and all vanishing points are assumed to have the same y-coordinate. We can take y-coordinate of any vanishing point for horizon level. In this work we choose the one with maximum support.

This vanishing point is estimated in a way similar to [5] using locally optimized RANSAC [14]. A pair of lines is chosen at random and their intersection is taken for vanishing point. Then all the lines that if proceeded lie close to this point are considered as inliers. After that vanishing point is recalculated as a closest point to all the inliers using least squares fit. Then set of inliers is changed according to estimated coordinates of vanishing point. This process is repeated several times, and at the end we choose a vanishing point with maximum support.

Lines filtering. Due to presence of various non-building objects in urban scenes a considerable amount of detected straight lines are not horizontal and do not refer to vanishing points. These lines are harmful for vanishing point estimation and should better be pruned out. In this work those lines are filtered out using machine learning like in [13].

Extracted line segments are classified by boosted trees trained on a set of manually labeled images into two categories: horizontal lines and clutter. We use color cues, geometric cues and gradient cues for classification. Geometric cues include coordinates of segment end points, its direction and length, and the fraction of segments that are almost parallel with it. Color cues include means and variances of color components and intensity calculated over the area around each line segment. We also use mean value of gradient along the line segment which characterizes edge strength. Fig. 3(c) shows lines left after filtering.

Vanishing points estimation. In this work vanishing point estimation is performed in two steps. First we perform lines grouping and then estimate vanishing point for each group of lines.

Lines grouping procedure is based on the fact that all lines which refer to the same vanishing point cross horizon level close to each other. Consider distribution of x-coordinates of crossings between horizon and line segments. Groups of parallel lines form the modes of this distribution. We seek for the modes of distribution using mean shift clustering [15].

Since y-coordinates of vanishing points are already known, we need to estimate only their x-coordinates. X-coordinate of one vanishing point for each group of lines is estimated with an algorithm quite similar to the one for horizon estimation. The difference is that we choose single line instead of a pair each time and take it's intersection with horizon level as a vanishing point. Lines marked as outliers by RANSAC are filtered out and are not used further for estimation orientation of the walls. Vanishing points and horizon level estimation are shown in Fig. 3(d).

3 Stagewise Inference of 3-d Model Parameters

In this work we assume that ground-vertical boundary is specified by a continuous polyline. Due to perspective constraints, each polyline segment passes through a vanishing point. Let's fix the left end of polyline (p_x^1, p_y^1), then a single polyline segment connects (p_x^1, p_y^1) with corresponding vanishing point (v_x^1, h). Let's denote the crossing of the first segment with the border between first and second walls by (p_x^2, p_y^2). Then single line segment passes through (p_x^2, p_y^2) and (v_x^2, h) etc. This implies that if x-coordinates of all polyline fractures and corresponding vanishing points are fixed then only a single polyline passes through (p_x^1, p_y^1) (see Fig. 5). Polyline is completely determined by a set af $2n + 1$ parameters $\{p_x^1, ..., p_x^n; v_x^1, ..., v_x^n; h; p_y^1\}$.

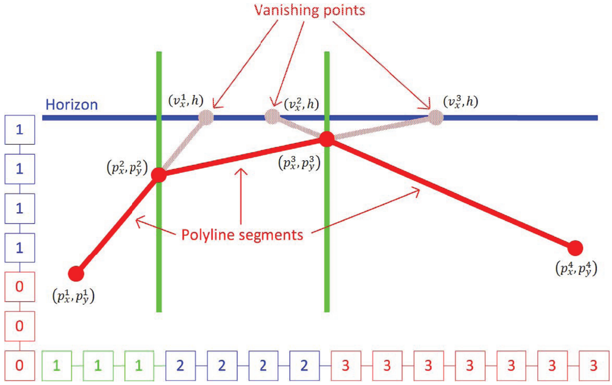

Fig. 5. Ground-vertical boundary Labeled chains represent results of polyline fractures estimation and polyline vertical positioning

Conditional probability formula allows dividing search for model parameters into two subproblems:

$$P(p_x^1,...,p_x^n;v_x^1,...,v_x^n;h;p_y^1|I) =$$

$$P(p_y^1|p_x^1,...,p_x^n;v_x^1,...,v_x^n;h;I)P(p_x^1,...,p_x^n;v_x^1,...,v_x^n;h|I) \tag{1}$$

The first subproblem includes estimation of x-coordinates of polyline's fractures $p_x^2,...,p_x^n$ and vanishing points $v_x^1,...v_x^n$ corresponding to each line segment. As far as vanishing points coordinates have been calculated at the stage of preprocessing, we just need to come up with correspondence between vanishing points and model walls. In this work for each wall that lies in between p_x^i and p_x^{i+1} we choose corresponding vanishing point that most of horizontal lines between p_x^i and p_x^{i+1} refer to. So the first problem evolves into estimation of $n-1$ x-coordinates of the polyline's fractures.

The second problem involves vertical positioning of the polyline with fixed $p_x^2,...,p_x^{n-1};v_x^1,...v_x^n;h$ parameters. If any point of polyline is known to lie below (above) the horizon level then polyline is guaranteed to lie entirely below (above) the horizon level. This finding involves a constraint $p_y^1 < h$ that reduces a search space for p_y^1 to a segment below horizon level.

4 A Conditional Random Field Model for Fractures of Ground-Vertical Boundary

Partition of picture into vertical walls is represented by a labeled chain graph (see Fig. 5). Each node of the graph corresponds to a vertical band in the image. Label l of a graph node means that corresponding vertical band in a image belongs to model wall with index l. Total number of labels K is chosen beforehand. As long as we allow different walls to share the same vanishing point, adjacent walls can actually lie on the same plane. Thus parameter K can be viewed as maximum possible number of vertical walls in 3-d model.

We introduce additional constrains that gurantee resulting model walls to be connected. For each pair of adjacent graph nodes the label of the right node can not be less than the label of the left one, or $p(l_{i+1} < l_i) = 0$, label of right node can not differ from the label of the left node by more than 1, or $p(l_{i+1} - l_i > 1) = 0$, label of the left end of the chain equals 1, $p(l_1 \neq 1) = 0$, label of the right end of the chain equals K, $p(l_w \neq n) = 0$, where w is the total number of graph nodes and K is maximum number of vertical walls.

The entire set of such graph labelings can be described in feature space with the parameter set θ. Desired output consists of the parameter set θ, plus a labeling $\mathbf{l}$.

The use of a Conditional Random Field allows incorporation of appearance, geometric and context cues in a single unified model. The intuition behind the model used in this work is the following. Horizontal lines which lie on the same building wall all meet at a same vanishing point, while adjacent building walls are not parallel and refer to different vanishing points. Due to lighting conditions color of each building wall in the image has low variance while different building walls have different colors. The borders between adjacent building walls are characterized with local discontinuities in color and orientation of lines.

CRF model for vertical walls takes the following form:

$$P(\mathbf{l}|I, \theta) = \prod_{i=1}^{w} p(l_i|\theta) \times \prod_{i=1}^{w-1} p(l_i, l_{i+1}), \tag{2}$$

where w stands for a total number of graph nodes, unary potential $p(l_i, \theta)$ captures distribution of features inside every vertical wall, pairwise potential $p(l_i, l_{i+1})$ captures the "edginess" between two adjacent graph nodes; θ is a vector of internal parameters of CRF model.

We use Expectation Maximization algorithm (EM) for CRF likelihood maximization. The first step (the "E" step) that includes searching for the best labeling $\mathbf{l}$ given fixed cluster parameters θ can be effectively solved using dynamic programming. The second step (the "M" step) is to fix the labeling $\mathbf{l}$ and find the best clusters parameters.

We adopt a nonparametric approach [20] for calculating unary potential. Suppose we are given a kernel function $K(\mathbf{f}_i; \mathbf{f}_j)$ which measures the affinity between features $\mathbf{f}_i$ of graph node i and features $\mathbf{f}_j$ of graph node j. Value of $K(\mathbf{f}_i; \mathbf{f}_j)$ shows how much we believe the two points originated from the same process when all we know is their coordinates. Suppose that we have an approximation of an optimal labeling of the chain. Then a nonparametric density estimator for probability that graph node i belongs to a label l given set of N points drawn from the distribution of features inside l-th wall is calculated as:

$$P(l_i = l|\mathbf{f}_{l1}, .., \mathbf{f}_{lN}) = \frac{1}{N} \sum_{j=1}^{N} K(\mathbf{f}_i, \mathbf{f}_{lj}), \tag{3}$$

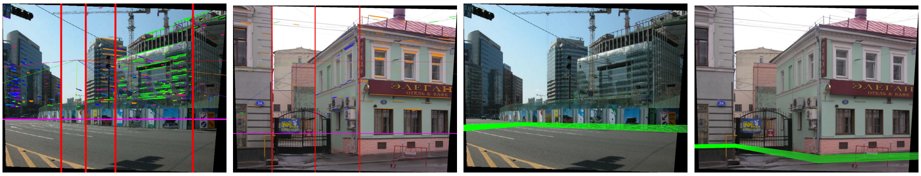

Fig. 6. Training examples (a),(b) borders of 3-d model walls are manually marked for training the first model (c),(d) optimal positions of polyline along the vertical axis are manually marked for training second model, fixed polyline fractures are inferred from the first model

where $\mathbf{f}_{l1}, .., \mathbf{f}_{lN}$ are features of reference graph node that belong to l-th wall. Vector $\theta = \{\mathbf{f}_{11}, .., \mathbf{f}_{1N}, ..., \mathbf{f}_{K1}, .., \mathbf{f}_{KN}\}$ contains features of graph nodes that belong to each label after each E-step of EM. Reference graph nodes in this work are chosen equally spaced inside each wall.

4.1 CRF Model Training

In this work we used a method based on piecewise training [17]. Piecewise training involves dividing the CRF model into pieces, each trained independently. So we need to find $K(\mathbf{f}_i; \mathbf{f}_j)$ and $p(l_i, l_{i+1})$ and use hand-labeled images of urban scenes for this purpose. The ground truth for learning included pictures where borders of vertical planes were marked manually (see Fig. 6(a)(b))).

Both $K(\mathbf{f}_i; \mathbf{f}_j)$ and $p(l_i, l_{i+1})$ are searched in exponential form:

$$K(\mathbf{f}_i; \mathbf{f}_j) = [1 + \exp(\sum_{m=1}^{M} \lambda^m |\mathbf{f}_i^{(m)} - \mathbf{f}_j^{(m)}|)]^{-1} \tag{4}$$

$$p(l_i, l_{i+1}) = [1 + \exp(-\sum_{m=1}^{M} \mu^m |\mathbf{f}_i^{(m)} - \mathbf{f}_{i+1}^{(m)}|)]^{-1}, \tag{5}$$

where $\mathbf{f}_i^{(m)}$ is the m-th component of feature vector, $\lambda = (\lambda^1, ..., \lambda^m)$ and $\mu = (\mu^1, ..., \mu^m)$ - coefficients of exponential models.

We use logistic regression for searching λ over a training set. Training set for learning $K(\mathbf{f}_i; \mathbf{f}_j)$ consists of pairs of features labeled with -1s if both corresponding graph nodes belong to the same wall and $+1$s if graph nodes belong to different walls. CRF pairwise term is learned from only positive examples, because of high class imbalance. Training set for learning $p(l_i, l_{i+1})$ consists of pairs of features which correspond to a pair of graph nodes around a marked edge. We use the first principal component obtained by PCA as coefficients in exponential model.

Due to requirements of computational efficiency of the algorithm, we use quite simple features $\mathbf{f}$ for estimation of ground-vertical boundary fractures. Consider

ith graph node, that corresponds to a vertical band of pixels between a_i and b_i. All features $\mathbf{f}_i$ are calculated over R rectangular regions limited by horizon level at the bottom with left borders $a_i - \delta_r, r = 1, ..., R$ and right borders $b_i + \delta_r, r = 1, ..., R$ respectively.

Feature vector includes simple color statistics and frequencies of lines referring to every vanishing point. To simplify calculation of these features we exploit integral images [22].

5 Vertical Positioning of Polyline

We formulate problem of polyline vertical positioning on a chain graph (see Fig. 4). Each graph node correspond to a fixed value of p_y^1 and in consequence a single boundary position along the vertical axis. The goal is to assign 1s to graph nodes that refer to vertical walls of 3-d model and 0s to graph nodes that refer to ground plane.

We again introduce Conditional Random Field, which incorporates different characteristics of ground-vertical boundary. The model takes into accout local color discontinuities around the boundary, similarity of ground color on different pictures, differences in color between each building and the ground below it, distinctive angles between of polyline's segments corresponding to visually-pleasant 3-d models.

Our CRF model for this problem has the following form:

$$P(\mathbf{l}|I) = \prod_{i=1}^{h} p(l_i) \times \prod_{i=1}^{h-1} p(l_i, l_{i+1}) \tag{6}$$

where h is a number of graph nodes, unary potential $p(l_i)$ captures color distribution inside region between polyline and horizon level and a region below polyline, $p(l_i, l_{i+1})$ captures local discontinuities around polyline position and angles between segments of polyline.

We again use piecewise training of CRF. In our experiments training set consists of pictures, where users manually marked ground truth positions of polyline which lead to visually pleasant models (see Fig. 6 (c)(d)). We exploit calibrated probabilistic outputs from boosting [19] for both unary and pairwise potentials. For example, unary potential takes a form:

$$p(l_i) = [1 + \exp(A_1 \cdot F(\tilde{\mathbf{f}}_i + B_1)]^{-1}, \tag{7}$$

where $\tilde{\mathbf{f}}_i$ is a feature vector of ith graph node, $F(.)$ is boosting output, A_1 and B_1 are Platt scaling parameters [19]. To avoid issues with imbalanced classes we use random undersampling [21].

Features $\tilde{\mathbf{f}}_i$ for unary potential include mean values of color and intensity around polyline. Features $\tilde{\tilde{\mathbf{f}}}_i$ for pairwise potential include mean and minimum angles between polyline segments and horizon, mean and minimum angles between adjacent polyline segments, mean, maximum and minimum y-cordinate of polyline, differences in color between regions below and above the polyline.

Again all color features are calculated over different rectangular regions around polyline segments and integral images are exploited for speed-up of calculation.

6 Modeling

Given the continuous polyline, horizon level in pixels and camera focal length in pixels it is possible to construct the 3-d scene. Our modeling scheme resembles the one, proposed in [23]. Scene is modeled as a horizontal polygon for the ground plane and a set of connected vertical trapeziform polygons for the rest of the scene. One vertical polygon is reconstructed for each straight segment of the polyline. The texturing is performed by straightforward projective mapping of tilt-corrected images onto reconstructed geometry. The idea of the algorithm is illustrated in Fig. 2.

Each vertex of the polyline in the image and bottom corners of the image should be projected onto the horizontal ground plane of the 3-d scene in order to determine the ground plane polygon. After that vertical planes with infinite heights can be also reconstructed. Top line in the image should be projected onto the reconstructed vertical planes to determine height of each vertical polygon. Since image is tilt corrected we can consider virtual views with zero pitch and roll angles, which mean that camera view vector is parallel to the ground plane and image center Y coordinate is equal to horizon level in the image. Z coordinate of the camera center affects only the scale of the resulting model and if it is specified exactly, we get a metric reconstruction as a result. Focal length in pixels can be calculated from EXIF data of JPEG file.

7 Experiments

Our image database is composed of 200 photos taken in a number of districts of Moscow and Seoul. Our test database also includes some photos taken in Moscow, Seoul and some images from Make3D and Automatic Photo Pop-up image bases [24][25]. We aim at outdoor urban scenes, so many of images of landscapes, indoor scenes, etc. from those databases are not included in our test set. These images are resized so that each image resolution is approximately 1.5 megapixel.

We have tested our algorithm, Make3D and Automatic Photo Pop-up on our test image database. Some of the test images and screenshots of the resulting models are shown in Fig. 7.

We did not use any heuristic measures for the quality of the models. The only criteria was the subjective opinion on the visual quality of the 3-d model. Resulting models were demonstrated to 4 users. Users were asked to answer whether the model had fine or unsatisfactory visual quality. The results of this visual comparison are shown in Table 1.

Due to our building-specific model, results of the proposed algorithm are visually more plausible, than the results of the compared algorithms. Hoiem el. al. and Make3D reported about 33% and 65% of visually pleasing models

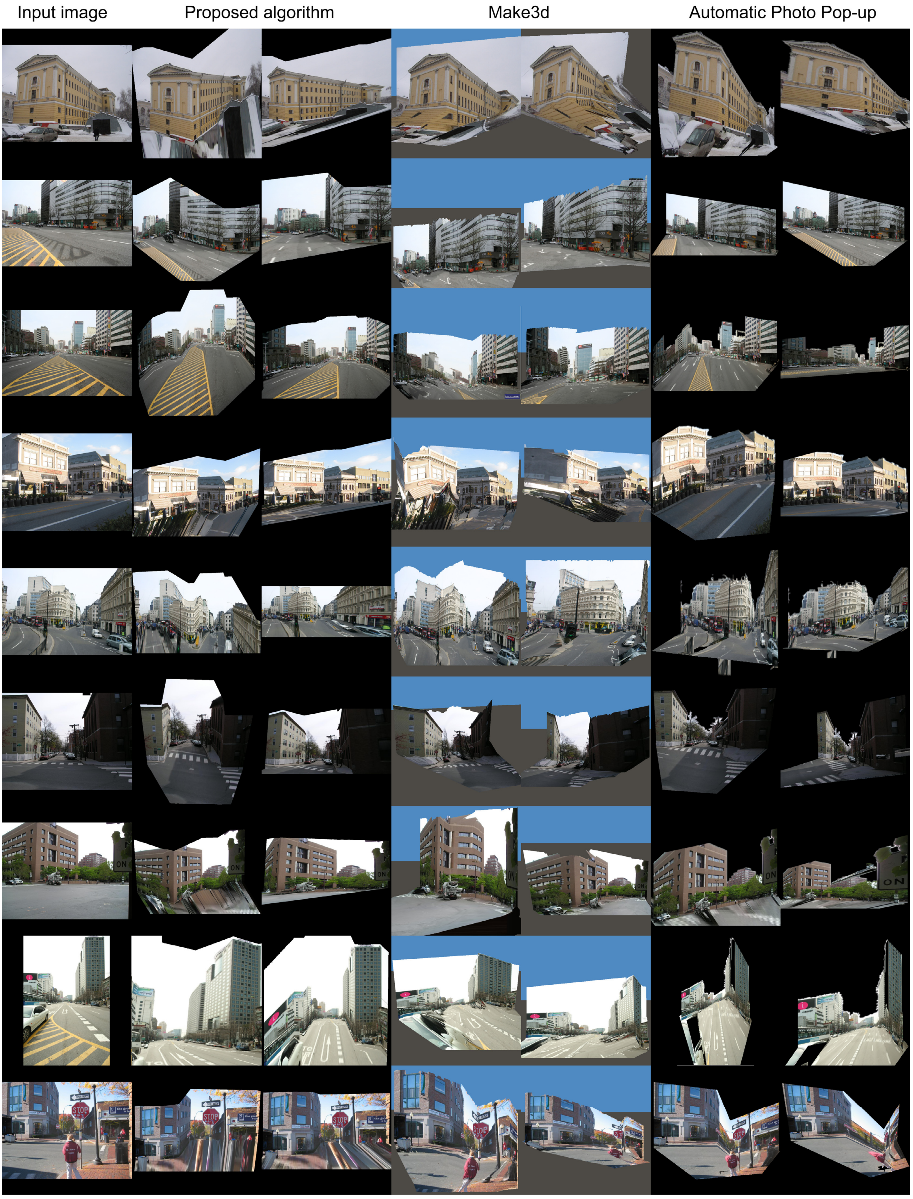

Fig. 7. Comparison of the proposed algorithm with Automatic Photo Pop-up and Make3D

correspondingly, but due to the specific structure of the images used for testing the compared algorithms performed worse.

Another advantage of the proposed algorithm is its low computational time and consumed memory. In our experiments with 1.5 megapixel images, using PC

Table 1. Comparison of our algorithm with Automatic Photo Pop-up and Make3D algorthms on our Seoul test database

	Our algorithm	Automatic Photo Pop-up	Make3D
Fine results (%)	43%	36%	25%
Best among 3 algorithms (%)	52%	27%	21%

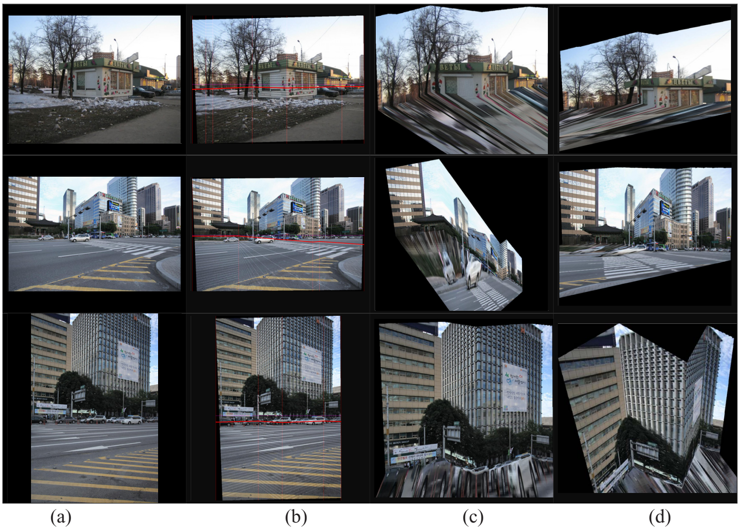

Fig. 8. Bad results (a)-original image, (b)-resulting ground-vertical boundary, (c),(d)-screenshots of 3-d model. The first example demonstrates error in polyline vertical positioning, second example shows errors in estimation of polyline fractures, third example shows the case when ou assumption about continuity of ground-vertical boundary does not suit to scene geometry.

Pentium 2.8 GHz with 1 GB RAM, average modeling time was about 8 seconds, while required memory didnt exceed 30 MBs. Hoiem et. al. reported about 1.5 minutes processing time for 800x600 image on 2.13 GHz Athlon using unoptimized MATLAB code. It was not mentioned about memory consumption, but in our experiments Auto Pop Up algorithm required up to 120 MBs. Algorithm proposed in [7][8][9] is slower and consumes more memory, than Automatic Photo Pop-up.

In some cases the proposed algorithm does not produce good looking resulting 3-d models. The most often reasons are incorrectly estimated fractures of the polyline or errors in its estimated position along the vertical axis. Another source of error is in the assumption on the scene layout and the constraint of the continious polyline of the ground-vertical boundary. Examples of such errors are demonstrated in Fig. 8.

Our algorithm, trained on images taken in a small number of districts, showed promising results. We believe that having a larger and more diverse dataset of training images would improve the algorithm.

8 Conclusions

We presented an algorithm for inferring rough 3-d structure from a single image of urban scene. Unlike Saxena et al we impose constraints on the geometry assuming that scene is composed of ground and a number of vertical walls, which is the case for urban scenes. In contrast to Hoiem et al we search for ground-vertical boundary, which completely defines the model in explict form.

Employing greedy search of 3-d model parameters we divide the problem into two smaller sub-problems. For each subproblem we incorporate appearance, geometric properties and context into CRF model trained with supervised learning and then solve for geometry via MAP inference. We compared our method with those proposed by Saxena et al and Hoiem et al on our test set composed of pictures of Seoul, Moscow and pictures downloaded from internet. This comparison have shown that our method provides 3-d model with comparable or superior visual quality at significantly less computational expense.

Acknowledgements. Special thanks to Alexander Sobolev, Anna Kuzmishkina and Olga Senyukova for valuable comments and help during all stages of this research. Many thanks to Victor Gaganov for priceless advises and help in improving the quality of the paper text.

References

1. Hoiem, D., Efros, A., Hebert, M.: Automatic photo pop-up. In: Proc. of ACM SIGGRAPH (2005)
2. Hoiem, D., Efros, A., Hebert, M.: Geometric context from a single image. In: Proc. of ICCV (2005)
3. Delage, E., Lee, H., Ng, A.: Automatic single-image 3d reconstructions of indoor manhattan world scenes. In: Proc. of ISRR (2005)
4. Delage, E., Lee, H., Ng, A.: A dynamic bayesian network model for autonomous 3d reconstruction from a single indoor image. In: Proc. of CVPR (2006)
5. Criminisi, A., Reid, I., Zisserman, A.: Single view metrology. IJCV 40 (2000)
6. Hoiem, D., Sten, A.N., Efros, A.A., Hebert, M.: Recovering Occlusion Boundaries from a Single Image. In: Proc. of ICCV (2007)
7. Saxena, A., Sun, M., Ng, A.: Learning 3-D Scene Structure from a Single Still Image. In: Proc. of ICCV workshop on 3D representation for Recognition (2007)
8. Saxena, A., Chung, S., Ng, A.: Depth Reconstruction from a Single Still Image. In: IJCV (2007)
9. Saxena, A., Chung, S., Ng, A.: Learning Depth from Single Monocular Images. In: Proc. of NIPS, vol. 18 (2005)
10. Canny, J.: A Computational Approach To Edge Detection. PAMI 8 (1986)
11. Kosecka, J., Zhang, W.: Video Compass. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2353, pp. 476–491. Springer, Heidelberg (2002)
12. Vezhnevets, V., Konushin, A., Ignatenko, A.: Interactive image-based urban modeling. In: Proc. of PIA, pp. 63–68 (2007)

13. Barinova, O., Kuzmishkina, A., Vezhnevets, A., Vezhnevets, V.: Learning class specific edges for vanishing point estimation. In: Proc. of Graphicon, pp. 162–165 (2007)
14. Chum, O., Matas, J., Kittler, J.: Locally Optimized RANSAC. In: DAGM Symposium on Pattern Recognition, vol. 25, pp. 236–243 (2003)
15. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Trans. on Pattern Analysis and Machine Intelligence 24, 5 (2002)
16. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proc. of ICML (2001)
17. Sutton, C., McCallum, A.: Piecewise training of undirected models. In: Proc. of UAI, vol. 21 (2005)
18. Shotton, J., Winn, J., Rother, C., Criminisi, A.: TextonBoost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context. In: IJCV (2007)
19. Niculescu-Mizil, A., Caruana, R.: Obtaining Calibrated Probabilities from Boosting. In: Proc. of UAI, pp. 413–420 (2005)
20. Wasserman, L.: All of Nonparametric Statistics. Springer, Heidelberg (2006)
21. Hulse, J., Khoshgoftaar, T., Napolitano, A.: Experimental perspectives on learning from imbalanced data. In: Proc. of ICML, pp. 935–942 (2007)
22. Viola, P., Jones, M.: Robust Real-time Object Detection. In: IJCV (2001)
23. Kang, H., Pyo, S., Anjyo, K., Shin, S.: Tour into the picture using a vanishing line and its extension to panoramic images. In: Proc. Eurographics, pp. 132–141 (2001)
24. `http://www.cs.uiuc.edu/homes/dhoiem/projects/data.html`
25. `http://make3d.stanford.edu/`