Pop-up Modelling of Hazy Scenes

Lingyun Zhao(✉), Miles Hansard, and Andrea Cavallaro

Centre for Intelligent Sensing, Queen Mary University of London, London, UK
lingyun.zhao@qmul.ac.uk

Abstract. This paper describes the construction of a layered scene-model, based on a single hazy image of an outdoor scene. A depth map and radiance image are estimated by standard dehazing methods. The radiance image is then segmented into a small number of clusters, and a corresponding scene-plane is estimated for each. This provides the basic structure of a 2.5-D scene model, without the need for multiple views, or image correspondences. We show that problems of gap-filling and depth blending can be addressed systematically, with respect to the layered depth-structure. The final models, which resemble cardboard 'pop-ups', are visually convincing. An HTML5/WebGL implementation is described, and subjective depth-preferences are tested in a psychophysical experiment.

Keywords: Layered scene models · Dehazing · Visualization · WebGL

1 Introduction

Impressive image-based 3D environments can now be found in many popular computer games and virtual reality tours. However, the creation of such environments remains a complicated and time-consuming process, often requiring special equipment, a large number of photographs, manual interaction, or all three. This makes it difficult for naive users to create an interactive 3D model.

This paper describes a system that can semi-automatically create a '2.5-D' model, from a single outdoor image, taken in hazy conditions. In particular, we use a *layered scene model* [19], in which the depth of each layer is estimated from the local haze-content of the images. We cannot, without other information, hope to create a full 3D model from a single view. However, we show that interactive 2.5-D models can be visually compelling.

The target application scenario is that the hazy outdoor photos would be processed rapidly, so that users would be able to browse them using ordinary web-browsers (e.g. on mobile devices). Our approach is similar to the creation of a 'pop-up' illustration in a children's book. Just like the cardboard pop-ups, our resulting 3D model is plane-based, as shown in figure 1. The foreground objects are automatically pasted onto different planes, with appropriate positions and orientations. No gaps should appear between the planes, as the viewpoint is varied. The scope of this paper is limited to outdoor scenes, in hazy conditions.

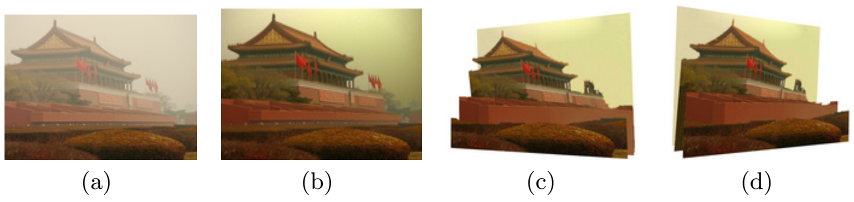


Fig. 1. Our system constructs a plane-based 3D environment from a single hazy image, without losing any information from the original image. (a) The original photograph; (b) dehazed version of the photograph. (c) and (d): two novel views from our 3D model.

However, the visualization pipeline could easily be adapted to any other source of depth-information, including motion [19], or disparity [21].

The challenges of generating a pop-up image are as follows. Firstly, the original image does not contain explicit depth information. In order to get the depth information, we usually need multiple views, or some other source of information. This can be difficult to obtain, especially if the image-quality is poor (e.g. due to haze), or if computational resources are limited (e.g. on mobile devices). Secondly, even if depth can be estimated, it is important to resolve visibility correctly, in order to produce a convincing model. This usually requires a surface model of some kind, such as a mesh, which can again be difficult to estimate.

This work makes use of the WebGL component of the new Html5 standard. The final pop-up models can be rendered very efficiently, because WebGL automatically uses GPU shaders for depth-testing and colour blending, whenever possible.

The method we propose can produce a visually pleasing 'pop-up' model, with interactive viewpoint control, based on a single hazy image. The novel contributions of this paper are as follows:

– We show that the haze-based depth information is suitable for use in 3D visualization.
– Texture-synthesis is used to fill the gaps that appear when the viewpoint is changed. Our method includes a novel preprocessing stage, which prevents 'mixed' image-regions (lying on depth boundaries) from being propagated.
– An appropriate image-blending model is introduced for depth-boundaries in the synthetic model.
– It is shown that the whole system is compatible with the WebGL rendering model, and can be displayed interactively in any Html5 browser.

This paper is organized as follows. In Section 2, we discuss state of the art in minimalistic 3D modelling. In Section 3, we introduce the pipeline to build up a plane-based 3D scene. In Section 4, we show the results of each step with five example images. Section 5 concludes the paper with discussion and ideas for future work.

2 Related Work

There are many different approaches in the field of image-based rendering and 3D modelling. Most systems require multiple images, or manually input information. Our system can automatically model a 3D outdoor scene — provided that a single hazy image is available.

2.1 Interactive 3D Modelling

The layered model was first proposed by Wang et al. [19]. They introduced a system for representing moving images with sets of overlapping layers. The layers are ordered by depth and they occlude each other in accord with the rules of compositing. They describe the methods for decomposing image sequences into layers using motion analysis.

Most research on full 3D reconstruction has focused on methods such as stereo vision or structure from motion. Popular urban modelling system introduced by Debevec et al. [2] and Cipolla et al. [3] require fewer images, but considerable user interaction. Other methods are able to perform user-guided modelling from a single image. Liebowitz et al. [16] and Criminisi et al. [13] demonstrate this approach, by recovering a metric reconstruction of an architectural scene, using geometric constraints. They can then compute 3D locations of user-specified points, via their distances from the ground plane. The user is also required to specify other constraints such as a square on the ground plane, a set of parallel 'up' lines and orthogonality relationships.

Most other approaches to build a metric reconstruction focus instead on producing perceptually pleasing approximations, but at the expense of quantitative accuracy. Zhang et al. [17] model free-form scenes by letting users place geometry constraints [14], such as normal directions, and then optimizing for the best 3D model to fit these constraints. Horry et al. [15] models a scene as an axis-aligned box, like a theatre stage, with floor, ceiling, backdrop and two side planes. An intuitive 'spidery mesh' interface allows the user to specify the coordinates of this box and its vanishing point. Foreground objects are manually labelled by the user and assigned to their own planes. This method produces impressive results but works only on certain scenes, since the front and back of the box are assumed to be parallel to the image plane. In our work, the positions and orientations of the planes are determined by the scene structure, rather than being limited to a pre-defined arrangement.

2.2 Automatic 3D Modelling

Hoiem et al. [4] introduced an automatic system to construct a rough 3D environment from a single image, by learning a statistical model of geometric scene-classes. Similar to our work, they limited their scope to dealing with outdoor scenes, but they also assumed that a scene is composed of a single ground plane, with piecewise planar objects sticking out of the ground at right angles and the sky. Under this assumption, they constructed a coarse, scaled 3D model from a single image by classifying each pixel as ground, vertical or sky and estimating the horizon position. Colour, texture, image location and geometric features are all useful cues for determining these labels. However, under their assumptions, crowded scenes, such as trees or people cannot be easily modelled. Additionally, their models cannot account for slanted surfaces, or scenes that contain multiple ground-parallel planes. Our system has no geometric constraints, but it does require a hazy photograph of the scene.

In recent work, Saxena et al. [6,7] present an algorithm for predicting depth from monocular image features. However, their depth maps are not accurate enough to produce visually-pleasing 3D models. Their method also relies on good quality of image features, which makes hazy images unsuitable. More recent work proposed by Saxena et al. [8] uses a Markov Random Field (MRF) to infer a set of plane parameters that capture both the 3D position and orientation of each surface-patch. They also incorporate object-information into the MRF to improve the results. However, the resulting models are often visually incomplete.

3 Model Construction

To create a layered model of the scene, we need information about its depth-structure. Formally, our system is based on the standard image-compositing 'over' operation $\mathbf{f} \odot \mathbf{b} = \mathbf{f} + (1 - \alpha_f)\mathbf{b}.$, where $\mathbf{f}$ and $\mathbf{b}$ are foreground and background layers, respectively. The opacity of each foreground pixel is determined by the alpha-map, α_f. We require a more general version of the operation $\mathbf{g} = \mathbf{f} \odot \mathbf{b}$, as used in the stereo-based model of Szeliski and Golland [21]:

$$\mathbf{g}(x,y) = \bigodot_{d=d_{\min}}^{d_{\max}} \mathbf{f}(x,y,d) \tag{1}$$

Here the operator $\bigodot_d$ composites each RGBA layer $\mathbf{f}$ over the current background layer, according to depth d. Note that $d_{\min}$ is the closest layer, which appears left-most in the sequence $\mathbf{f}(x,y,d_{\min}) \odot \cdots$, by analogy with the original over-operator.

Rather than using multiple views, range scanners, or complex scene-models, we use the depth information in hazy images. Meanwhile, we use the dehazed radiance image to provide the colour information for compositing. The complete pipeline is shown in figure 2, and described in the following sub-sections.

3.1 Dehazing

Atmospheric particles cause the *airlight* colour, which is effectively that of the haze, to mix with the scene radiance. This changes the observed colour distributions, and reduces the contrast of the images. Furthermore, these effects are distance dependent, which means that *dehazing* the images can produce information about scene-depth, as well as radiance. This can be seen from the Lambert-Beer law,

$$t_i = \exp(-\lambda r_i) \tag{2}$$

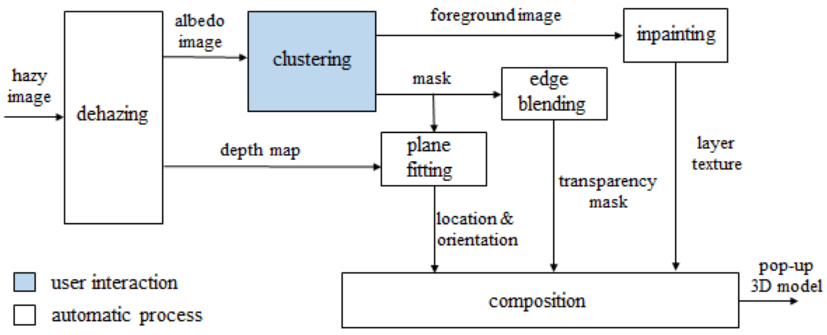
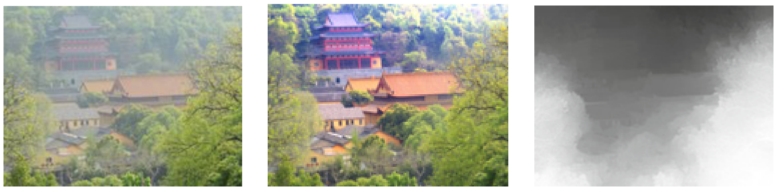


Fig. 2. Block diagram of the proposed system for creating a pop-up 3D model.



(a) Hazy image (b) Radiance image (c) Depth image

Fig. 3. Example of dehazing, using the dark-channel prior of He et al. [1].

where r_i is the distance of the scene along the i-th ray, λ is a weather-dependent constant, and t_i is the optical *transmission*.

We use the Dark Channel Prior (DCP) dehazing algorithm to estimate the transmission [1]. He et al. introduced the DCP, which is based on the assumption that every image patch will contain the projection of a dark or colourful (i.e. dark in one RGB channel) scene-point. Having estimated the global airlight colour, from the sky region, the transmission can be estimated from the dark-channel pixels.

3.2 Clustering and Plane-fitting

Rather than making a full 3D model, our target is to create a layered visualization. Hence it suffices to simply *cluster* the depth and radiance outputs of the dehazing process. This paper is not concerned with the details of the clustering process, so we use the semi-supervised GrabCut [18] method, to define a small number of clusters in the radiance image (usually around five).

We convert the range estimates $r_i = -\log(t_i)/\lambda$ in equation (2) to depth values d_i, (measured perpendicular to the optical axis). We then use the RANSAC algorithm [10] to fit a depth-plane to each cluster of (x, y, d) values. RANSAC is reliable even in the presence of a high proportion of outliers, which is relevant here, because the dehazing sometimes produces very bad depth-estimates (e.g. when an object has the same colour as the airlight). Note that the depths are only determined up to the overall scale-factor λ; this issue will be considered in section 4.

The clustering and fitting stage produces two outputs. Firstly, a label image, which maps each pixel to one of the layers. Secondly, a list of plane vectors, which encode the position and orientation of each layer. These outputs are stored as PNG images, and associated JSON data, which can be read by the web-browser.

3.3 Inpainting

Image inpainting is the problem of filling new colour and texture into missing parts of an image, so that the result is visually convincing. In our case, the missing parts appear at depth-boundaries, when the viewpoint changes [20]. We used Criminisi's algorithm [9] to fill the gap with patches from the surrounding region. These patches should continue the image texture into the hole, without introducing visual artifacts. Such techniques are collectively referred to as patch or examplar-based methods. Figure 4 shows the visualisation improvement after inpainting.

Two important modifications are made to the inpainting procedure. Firstly, patches should be sampled from the appropriate layer, so that the background texture grows *behind* the occluding layer. The inpainting algorithm starts computation from the edge to the centre of holes, and is therefore sensitive to the image-structure at the edge. However, we observe that this image structure is usually *not* representative of the occluded layer, because it will contain some mixture of the occluding layer (especially at complex boundaries, like foliage). Hence our second modification is to perform an erosion of each cluster, before proceeding with the inpainting process. This results in significantly more convincing textures, as shown in figure 5. The erosion radius was set to eight pixels, in all cases.

3.4 Edge Blending

The clustering process produces image-segments with hard edges. These look very unconvincing in the pop-up visualisation, especially at complex

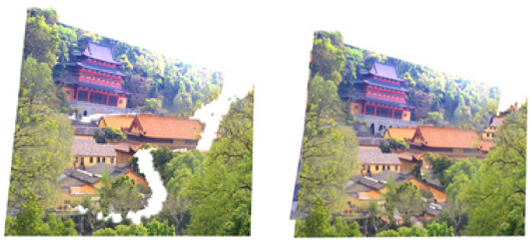

(a) Before inpainting (b) After inpainting

Fig. 4. Comparison between the layered model before and after inpainting.

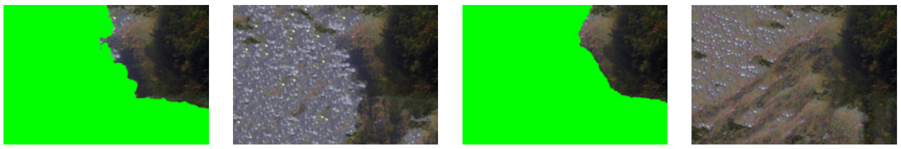

(a) Original cluster (b) Original inpainting (c) Eroded cluster (d) Eroded inpainting

Fig. 5. Comparison between direct inpainting result and inpainting after erosion. The original cluster is a part of a tree with some sky region at the edge. The green region is labelled as the hole. Direct inpainting tends to propagate the edge texture, which is not representative of the foliage. A more natural inpainting is obtained after erosion.

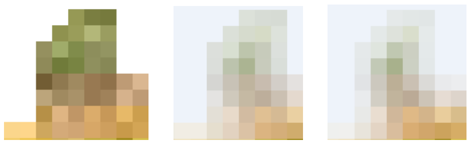
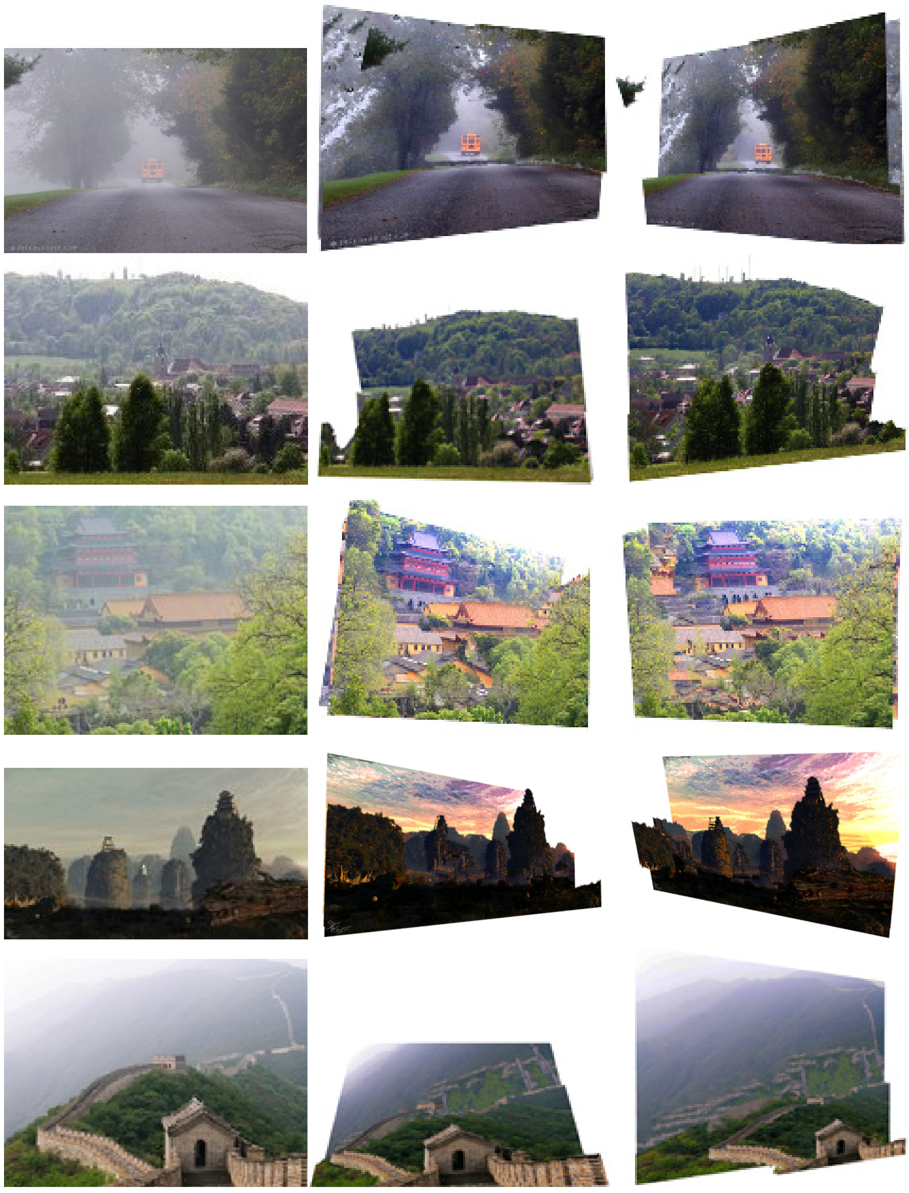

(a) Original (b) Linear (c) Cubic

Fig. 6. Comparison between different edge blending methods. (a) Original cluster-edge. White pixels indicate the background region. (b) Linearly-blended image. (c) Cubic blend produces a smoother transition.

depth-boundaries (e.g. due to foliage). To make the transition more natural, we add an alpha channel to each foreground layer. We then compute the distance-transform of each segment, with respect to any adjacent and *more distant* segments. This gives a scalar map $\delta(\mathbf{x})$, which encodes the image-distance from pixel $\mathbf{x}$ to the nearest occluded point. It is then straightforward to set a transition in the alpha channel $\alpha(\mathbf{x})$, based on the following rule:

$$\alpha(\mathbf{x}) = \begin{cases} \beta\big(\delta(\mathbf{x})/\epsilon\big) & \text{if } \delta(\mathbf{x}) \leq \epsilon \\ 1 & \text{otherwise} \end{cases} \tag{3}$$

where ϵ is the 'border-width' of the depth transition (set to 5 pixels in all experiments here), and $\beta(\cdot)$ is a blending function. One possibility for the latter is simply $\beta(t) = t$, which results in a linear transition. The cubic Hermite function $\beta(t) = t^2(3 - 2t)$ is more suitable, because it has zero slope at either end of the interval $t \in [0, 1]$, which results in a smoother transition. This is illustrated in figure 6. The outside edge pixels in the cubic edge blending result are softer, compared to the other results. We therefore apply cubic edge blending in our experiment.



(a) (b) (c)

Fig. 7. Examples of pop-up scenes, generated with our method. Column (a) shows the original hazy input images. Columns (b) and (c) show example novel views, taken from our pop-up models.

4 Results

We use the Three JavaScript library [22], which is built on WebGL, to display the final models. Each layer is rendered as a quadrilateral, with the corresponding RGBA texture obtained as described in section 3. The compositing procedure (1) involves *order-dependent transparency*, which can be problematic in OpenGL/WebGL. However, in practice, we find that the layers do not have complex depth relationships, and so few artefacts are observed.

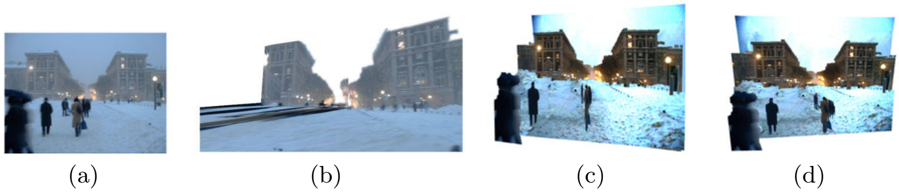


Fig. 8. Original hazy image (a), and 3D model (b), taken from [4]. Note the artefacts corresponding to the pedestrians in (b). Images (c) and (d) show other novel views generated from (a), using our system.

We can use either perspective or orthographic cameras to project the model. The former (used here) looks more natural, but the latter has the advantage of exactly re-creating the input image, from the appropriate viewpoint. We add interactive controls for the viewpoint and depth-scaling. The latter compensates for the unknown constant λ in equation (2), by effectively letting the user set a visually pleasing depth-range for the scene.

Figure 7 shows the qualitative results of our algorithm on several real-world images. Note that the lower region, in most outdoor photographs, is nearer to the observer than the upper region. We tested our algorithm on an image from the Hoiem et al. [4] database. The failure in their result due to overlap of vertical regions in the image is solved in our work. (Figure 8). Since we do not have the knowledge about the ground truth, the location of each plane is largely based on the accuracy of the dehazing algorithm. Some inherited problems from dehazing will also affect our result. Figure 9 shows an example of a typical failure.

4.1 Evaluation

We test viewers' subjective preferences of depth structure, in a psychophysical experiment. The evaluation system is made in a web browser, with user-interactive buttons to increase or decrease the overall depth-range of the scene (i.e. the distance between the front and back layers). Meanwhile, the entire scene undergoes a slow left-right change of viewpoint, which is not under user-control. We asked the participants to interactively adjust the depth scale, in order to

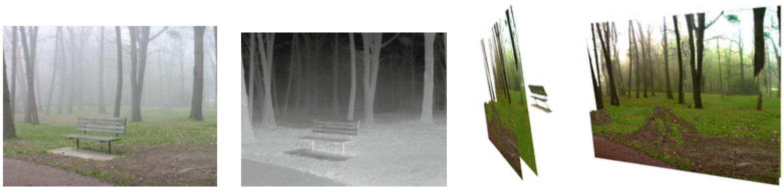

(a) Original image (b) Depth map (c) Side (d) Front view

Fig. 9. Example of depth failure. The colour of the bench is similar to that of the haze, leading to an over-estimate of its depth (c). Nonetheless, the front view of the final model is visually consistent (d).

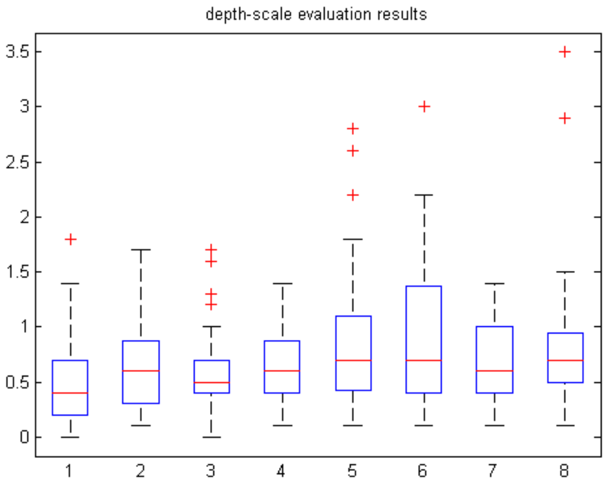


Fig. 10. Depth preferences by scene. The red horizontal lines indicate the median depth scale, for each of the eight scenes. The box edges are the 25th and 75th percentiles. The red crosses are outliers. The vertical axis is the scale factor, which is 1 for the default configuration.

achieve the most pleasing visual effect. A demonstration video for our experimental results is available online at: http://youtu.be/qKoZ5A-MvA4.

We asked 13 observers to evaluate 8 scenes, in random order, with three repetitions of each. The evaluation results are shown in figures 10 and 11. We draw the following general conclusions. Firstly, most people enjoy the experience of the virtual pop-up scene with depth information, because they do not set the depth-scale to zero. Secondly, for some specific images, such as that shown in figure 9, people prefer the trivial planar image to the pop-up. This is due to dehazing artefacts, such as missing foreground objects. Thirdly, based on the

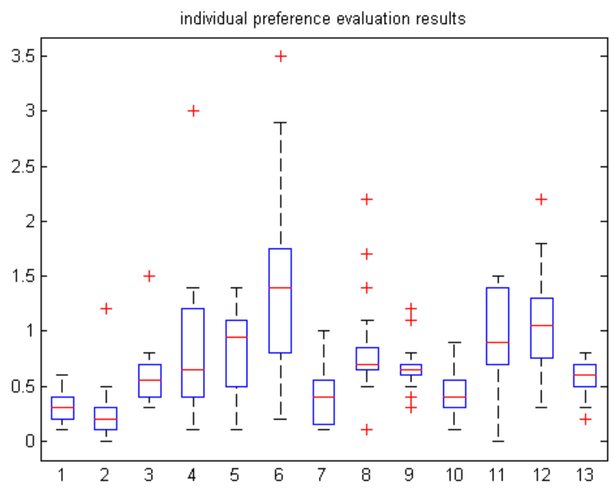


Fig. 11. Depth preferences by participant. The red horizontal lines indicate the median depth scale, for each of the 13 participants. The vertical axis is the scale factor, which is 1 for default scenes, as in figure 10.

experimental results, some planes may intersect each other at extreme depth-scales, which may limit the amount of depth that users prefer.

We also analyse viewers' individual preferences, by showing the evaluation results for each person, in figure 11. With reference to the median values, we conclude that the depth-preference varies from person to person, and that most participants prefer a scene with significant depth. Participant number 6 has much higher (and more variable) depth preferences than the others. Participants numbered 1, 2, 3, 9 and 13 are female; the other participants are male. We note that the female depth preferences appear to be more consistent across different scenes.

5 Conclusions

We have proposed a new way to model a 3D scene, based on a single hazy image. The result can be viewed interactively in an ordinary web-browser. We have shown that the problems of hole-filling and edge-blending can be addressed systematically, in relation to the layered-depth structure.

The algorithm of automatic single-view 3D reconstruction still needs to be improved. We plan to replace GrabCut with an automatic method to segment the radiance image, e.g. based on Mean Shift. This will additionally make use of the depth map, from the dehazing process, to perform RGBD clustering. It would also be interesting to study how the perceived quality of the final model depends on the number of clusters. Finally, future experiments will allow the participants to control the relative plane-depths, rather than just the overall depth scaling. It will be interesting to analyze the consistency of these judgements, and how they relate to 3D scene perception.

References

1. He, K., Sun, J., Tang, X.: Single Image Haze Removal Using Dark Channel Prior. IEEE Trans. PAMI **33**(12), 2341–2353 (2011)
2. Debevec, P.E., Taylor, C.J., Malik, J.: Modeling and rendering architecture from photagraphs: a hybrid geometry and image-based approach. In: Proc. ACM SIGGRAPH, pp. 11–20 (1996)
3. Cipolla, R., Robertson, D., Boyer, E.: Photobuilder - 3D models of architectural scenes form uncalibrated images. In: Proc. IEEE Conference on Multimedia Computing and Systems, pp. 25–31 (1999)
4. Hoiem, D., Efros, A., Hebert, M.: Automatic photo pop-up. In: Proc. ACM SIGGRAPH, pp. 577–584 (2005)
5. Hoiem, D., Efros, A., Hebert, M.: Putting objects in perspective. In: Proc. CVPR, pp. 3–15 (2006)
6. Saxena, A., Chuang, S.H., Ng, A.Y.: Learning depth from single monocular images. In: Proc. NIPS, vol. 18, pp. 1161–1168 (2005)
7. Saxena, A., Chuang, S.H., Ng, A.Y.: 3D depth reconstruction from a single still image. In: Proc. IJCAI, pp. 53–69 (2007)
8. Saxena, A., Sun, M., Ng, A.Y.: Learning 3D scene structure from a single still image. In: Proc. ICCV, pp. 1–8 (2007)
9. Criminisi, A.: Objecte removal by exemplar-based inpainting. In: Proc. CVPR (2003)
10. Fischler, M., Bolles, R.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Communications of the ACM **24**, 381–395 (1981)
11. Schnabel, R., Wahl, R., Klein, R.: Efficient RANSAC for Point-cloud Shape Detection. Computer Graphics Forum **26**(2), 214–226 (2007)
12. Yingyang, M., Förstner, W.: Plane Detection in Point Cloud Data. Technical report, Institute of Geodesy and Geoinformation, University of Bonn (2010)
13. Criminisi, A., Reid, I., Zisserman, A.: Single View Metrology. Int. Journal of Computer Vision **40**(2), 123–148 (2000)
14. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press (2004)
15. Horry, Y., Anjyo, K.I., Arai, K.: Tour into the picture: using a spidery mesh interface to make animation from a single image. In: Proc. ACM SIGGRAPH, pp. 225–232 (1997)
16. Liebowitz, D., Criminisi, A., Zisserman, A.: Creating architectural models from images. In: Proc. Eurographics, vol. 18, pp. 39–50 (1999)
17. Zhang, L., Dugas-Phocion, G., Samson, J., Seitz, S.: Single view modeling of freeform scenes. In: Proc. CVPR, pp. 990–997 (2001)
18. Rother, C., Kolmogorov, V., Blake, A.: Grab-Cut: Interactive foreground extraction using iterated graph cuts. ACM Trans. Computer Graphics **23**(3), 309–314 (2004)

19. Wang, J.Y.A., Adelson, E.H.: Representing Moving Images with Layers. IEEE Trans. Image Processing **3**(5), 625–638 (1994)
20. Swirski, L., Richardt, C., Dodgson, N.A.: Layered Photo Pop-Up. SIGGRAPH Posters (2011)
21. Szeliski, R., Golland, P.: Stereo matching with transparency and matting. In: Proc. ICCV, pp. 517–524 (1998)
22. Three JavaScript Library. http://threejs.org/