Eurographics Symposium on Rendering 2011
Ravi Ramamoorthi and Erik Reinhard
(Guest Editors)

Volume 30 (2011), Number 4

Silhouette-Aware Warping for Image-Based Rendering - Supplementary material

Gaurav Chaurasia, Olga Sorkine, George Drettakis

We present results of automatic segmentation algorithms for our datasets (see Sec. 1) and implementation details of the 3D point processing step of our approach (see Sec. 2).

1. Silhouette Extraction

A comparison between different segmentation algorithms and manual silhouettes annotation is shown in Fig. 1. All results have been generated using code provided by the authors. The first row shows one of the input images of our datasets. Second row shows silhouettes annotated manually which have been used to generate the image-based rendering results presented in the paper. The third row shows the final result of occlusion boundary extraction from single image by Hoiem et al. [HSEH07]. The fourth and fifth rows show the soft and binary edge maps using a combination of Arbelaez [Arb06] and Maire et al. [MAFM08] called *gPb-ucm* algortihm. The last row shows hierarchical segmentation [AMFM11] using results from the fifth row.

We observed that [HSEH07] works well for some cases, but suffers from inaccurate localization, false positives and missed edges. We tried erasing false matches and scribbling missed edges manually, but this took more time than completely manual annotation of silhouettes. We also converted edge-maps polylines using contour tracing [TC89] and polygon approximation [DP72] to check if user interaction on polygonal curves is easier. However, contour tracing becomes ambiguous for many intersecting contours and the result has too many doubled line segments and noise (see Fig. 2(c)) to fix in less than 40 seconds per image required for manual annotation. The double line appears because chaining algorithm tries to close the contour by walking back and forth.

We tried using depth cues from reconstruction to eliminate false positives. For this, we oversegmented the images into thousands of contiguous regions, called 'superpixels' (Fig.3(b)) using [FH04], and used available depth to assign a depth to every superpixel. The boundary between two superpixel is considered a silhouette if they are at substantially different depths (Fig.3(c)). However, we observed these 'depth images' have very noisy silhouettes becuase not all superpixels have robust depth information.

We believe that a combination of depth and multi-view information with the best elements of previous segmentation algorithms could allow the development of an more robust algorithm. While a completely automated approach is very hard, we are confident that the overall manual interaction time would be greatly reduced with such an approach.

2. 3D Point Selection

The goal of this step is to decimate the reconstructed point set down to a sparse set distributed uniformly over the image, fill in regions with few or no reconstructed points and remove erroneous points near silhouettes. The following steps are used to The 3D point selection described in Sec. 4.2 of the paper involves the following steps:

Decimation. We splat the point set with a large splat size and depth test enabled. We count the number of pixels that each point covers after the depth test. We select a subset of desired size that covers the maximum number of pixels (see Fig. 4(a),(b)). The splat size is not critical as long as it is not too small (e.g., 21×21).

Hole filling. The point set retained after decimation is splatted on the image. If a window of $n \times n$ pixels does not contain any splatted points, we mark this window as a hole. We add a 3D point which projects inside this window at the depth of nearest neighbor point. We choose n as the same as the splat size used earlier (see Fig. 4(c)).

Silhouette points. Reconstructed points near silhouettes can be erroneous. To ensure consistency, we remove all points within a 20 pixel distance of silhouettes and replace them by new 3D points that project to the same position with depth equal to other points' depth on the same side of the silhouette. This ensures that silhouettes clearly separate points with different depths.

Manual outlier removal. This optional step is useful when there are many points at incorrect depths. Our interface shows points with color coded depths, which makes it easy for the user to identify such outliers (see Fig. 4(d)). They can be removed by a simple 'select-and-delete' operation at any stage of the process.

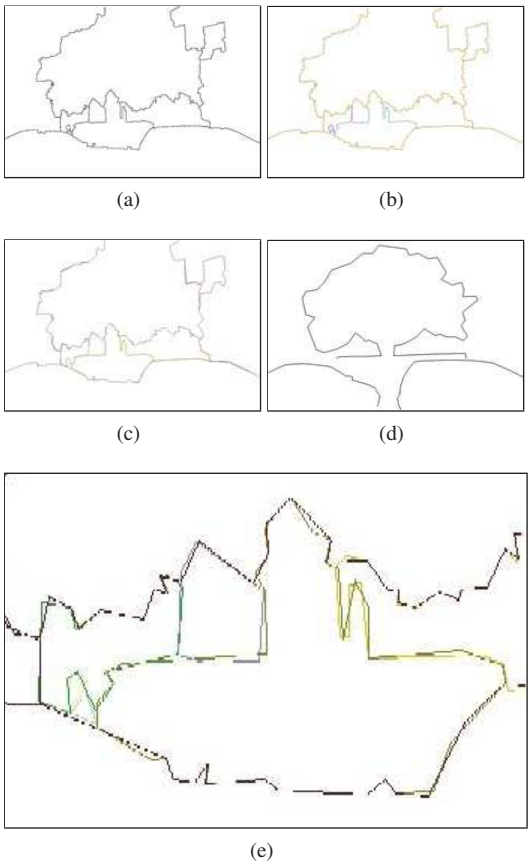

Figure 2: *(a) Binary edge map using [HSEH07]. (b) Contour tracing with each contour shown in a different color. (c) Polygon approximation of the contours. Note the jagged contours and double line segment for each contour. (d) Manual silhouettes. (e) Zoomed in view of line segments in (c). We observed that because of double contours and intersections, user input required to convert (c) into (d) was more than the annotating silhouettes manually.*

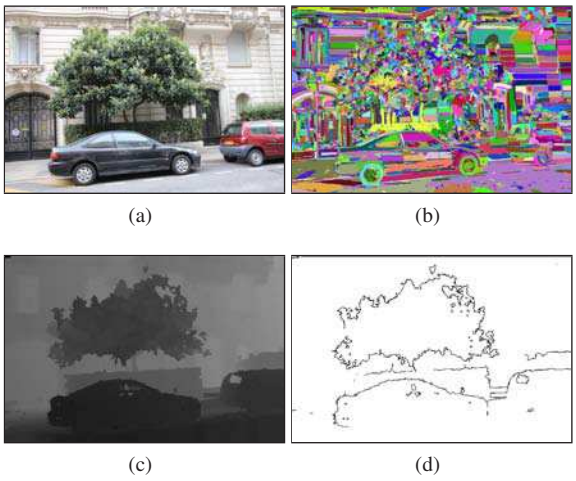

Figure 3: *Depth cues for segmentation (a) Input image. (b) Image decomposed into superpixels using [FH04]. (c) Depth map with each superpixel assigned the depth of reconstructed points present inside itself. (d) Edge detection on the depth map. Note the noisy edge map, which will require precise user intervention.*

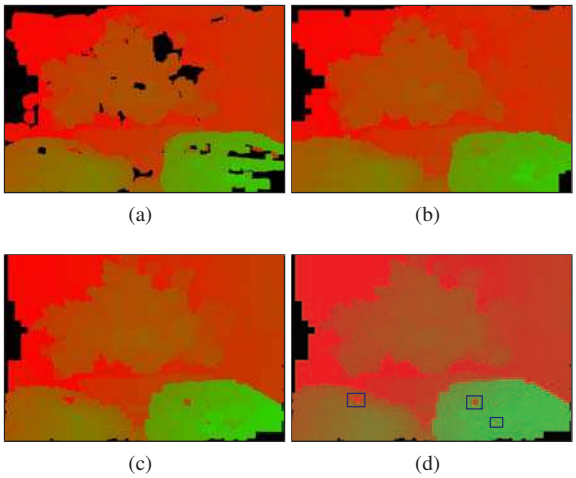

Figure 4: *(a) Splat of all input points with depth ranging from green (near) to red (far) and splat size* 21×21*, (b) Same splat after retaining 5k points, (c) Same splat after hole filling. (d) Outlier points with wrong depth shown in blue box.*

References

[AMFM11] ARBELAEZ P., MAIRE M., FOWLKES C., MALIK J.: Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell. to appear* (2011).

[Arb06] ARBELAEZ P.: Boundary extraction in natural images using ultrametric contour maps. In *Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop* (2006), CVPRW '06, pp. 182–.

[DP72] DOUGLAS D., PEUCKER T.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer 10*, 2 (1972).

[FH04] FELZENSZWALB P. F., HUTTENLOCHER D. P.: Efficient graph-based image segmentation. *Int. J. Comput. Vision 59* (September 2004), 167–181.

[HSEH07] HOIEM D., STEIN A. N., EFROS A. A., HEBERT M.: Recovering occlusion boundaries from a single image. *ICCV* (2007), 1–8.

[MAFM08] MAIRE M., ARBELAEZ P., FOWLKES C., MALIK J.: Using contours to detect and localize junctions in natural images. *In Proc. CVPR* (2008), 1–8.

[TC89] TEH C.-H., CHIN R.: On the detection of dominant points on digital curves. *IEEE Trans. Pattern Anal. Mach. Intell. 11*, 8 (Aug. 1989).

Figure 1: *Automatic silhouette extraction results. Top row: input image. The second row: manually annotated silhouettes. Third row: final output of [HSEH07]. Fourth row:* gPb-ucm *algorithm [Arb06]+ [MAFM08]. Fifth row: binary edge map obtained by thresholding the* gPb-ucm *result. Sixth row: Hierarchical segmentation [AMFM11]. Note that the result of automatic segmentation are similar to manual annotation only for first and last dataset, and even then the localization is not very good. See Sec. 1 for more information about each method.*