

Depth SEEDS: Recovering Incomplete Depth Data using Superpixels

Michael Van den Bergh¹
¹ETH Zurich
Zurich, Switzerland

vamichae@vision.ee.ethz.ch

Daniel Carton²
²TU München
Munich, Germany

carton@lrsr.ei.tum.de

Luc Van Gool^{1,3}
³KU Leuven
Leuven, Belgium

vangool@esat.kuleuven.be

Abstract

Depth sensors have become increasingly popular in interactive computer vision applications. Currently, most of these applications are limited to indoor use. Popular IR-based depth sensors cannot provide depth data when exposed to sunlight. In these cases, one can still obtain depth information using a stereo camera set up or a special outdoor Time-of-Flight camera, at the cost of a reduced quality of the depth image. The resulting depth images are often incomplete and suffer from low resolution, noise and missing information. The aim of this paper is to recover the missing depth information based on an extension of SEEDS superpixels [11]. The superpixel segmentation algorithm is extended to take depth information into account where available. The approach takes advantage of the boundary-updating property of SEEDS. The result is a clean segmentation that recovers the missing depth information in a low-quality depth image. We test the approach outdoors on an interactive urban robot. The system is used to segment a person in front of the robot, and to detect body parts for interaction with the robot using pointing gestures.

1. Introduction

The use of depth sensors for interaction has known a jump in popularity after the recent introduction of consumer depth cameras (e.g., Kinect). However, the applications usually stay limited to indoor use as these infrared (IR)-based sensors cannot function in the presence of sunlight. They fail because the IR light from the sun overpowers the IR light or pulse emitted from the sensor. Yet many natural gesture-based interactions occur outdoors, e.g., pointing at landmarks, streets, objects and persons during interaction.

It is still possible to obtain a depth image outdoors using a special outdoors Time-of-Flight camera or a real-time stereo reconstruction. This comes at the cost of reduced image quality: the resulting depth images usually suffer from noise and missing depth information (holes), which limit the ability to segment persons and objects accurately.



Figure 1. The goal of this paper: taking a noisy, incomplete depth image (e.g., from real-time stereo) and recovering the missing depth detail using superpixels. The improved depth image can then be used for the segmentation of objects or persons, e.g., for human-computer interaction.

The aim of this paper is to clean up these incomplete depth images based on an extension of the SEEDS [11] superpixel algorithm, and at the same time to facilitate the segmentation of objects or persons based on depth. We present a new pixel-level updating algorithm for SEEDS that takes depth into account and handles unknown depth values efficiently. During the iterations of the algorithm, each superpixel not only maintains a statistic of its color, but also a mean depth based on the known depth values inside the superpixel. The approach takes advantage of the boundary-updating property of SEEDS, for which the optimal updating strategy can be selected based on what information is available for each pixel. The boundaries are updated in order to group together pixels based on color and depth similarity. Pixels without depth information are assigned based on color and based on the labeling of the surrounding pixels. The result is a clean superpixel segmentation, without noise and holes, that tightly follows object boundaries, as shown in Fig. 1. Segmenting objects and persons is then as

simple as connecting the superpixels with matching depth at their connecting boundaries.

We test this approach on a real-world application. We installed two cameras on the head of an interactive urban robot, from which we extract a real-time stereo reconstruction. Depth SEEDS are then used to segment a person in front of the robot and detect body parts. The user can then interact with the robot using pointing gestures.

2. Background

For this paper, we make use of depth images obtained from sensors which function outdoors. The two solutions to our knowledge are an outdoor Time-of-Flight camera (*e.g.*, Optex ZC-1000), or a real-time stereo system. Bleyer *et al.* [2] present a real-time stereo algorithm based on GPU-trees. This approach is fast, but the output is not ideal for segmentation as lots of oscillations appear in the disparity image. Geiger *et al.* [8] present a CPU-based real-time stereo algorithm (libELAS) that provides disparity images that are more useful to our application. The approach first looks for a selection of good feature points in both the left and right camera images, and creates a triangular mesh based on those feature points. The other pixels are expected to lie close to this triangular mesh, requiring only to look for a match in a local region, reducing the computation time. This also results in smooth surfaces which are beneficial for our object segmentation. The resulting depth maps are impressive, but nevertheless contain a lot of holes (parts of the depth image that are incomplete).

There are a number of typical approaches for dealing with this noise and holes. In [8], they are filled up based on a minimum hole size. Any hole smaller than a threshold is filled up with the depth values from surrounding pixels. Another typical solution to deal with noise is smoothing based on an MRF [9]. The problem with these approaches is that they clean up the noise without knowledge of the underlying image. Considering that the color image of the same scene is available, recovering the depth data can be done by incorporating this data.

In the field of object segmentation, many approaches make use of superpixel-based low-level partitioning of the image [5, 7, 4, 6, 10, 3]. This allows for groups of pixels (superpixels) to be classified and thus to achieve an accurate segmentation of the whole image. This partitioning is usually based on clustering pixels based on colors or local texture features. Achanta *et al.* [1] present a fast approach for color-based clustering of the image into superpixels, based on *simple linear iterative clustering* (SLIC). Van den Bergh *et al.* [12] extend SLIC to take into account depth and optical flow data. However, this method does not deal with missing depth values and incomplete depth images. In fact, it is unable to deal with incomplete depth images because in SLIC, pixels are assigned to superpixels regardless of strict

spatial constraints. This results in a lot of stray labels which need to be cleaned up after the iterations of the algorithm (which can be seen as a form of hole-filling itself). Furthermore, SLIC is not real-time, and it is not state-of-the-art in terms of accuracy and boundary recall.

Recently, Van den Bergh *et al.* [11] presented a new superpixel algorithm (SEEDS), which is significantly faster than SLIC and achieves state-of-the-art accuracy. Furthermore, the boundary-updating property of SEEDS lends itself well to the use with incomplete depth images. It allows us to adapt the updating strategy based on the available depth information.

The remainder of this paper is structured as follows: in Section 3 we introduce Depth SEEDS, our adaptation of the SEEDS algorithm. Then, we present a person segmentation application, which uses the Depth SEEDS in order to segment a person and detect body parts for interaction. In the experiments section (Section 4) we evaluate the recovery of depth information and present a real-world experiment where the algorithm is run on a robot for interaction with humans.

3. Depth SEEDS

Recently, SEEDS [11] was proposed as an efficient superpixel segmentation algorithm. We have chosen to use SEEDS in this paper for its real-time properties and because it offers state-of-the-art performance. Furthermore, SEEDS has an interesting boundary updating property, which lends itself very well to our purpose.



Figure 2. The SEEDS algorithm works in three steps (from left to right): first, the superpixels are initialized as a grid. Then, the superpixel boundaries are moved with coarse block-level updates. Finally, the boundaries are refined using pixel-level updates.

SEEDS first initializes the superpixels as a rectangular grid, and then iteratively moves the boundaries of the superpixels in order to maximize an energy function and thus to refine the segmentation. First, the boundaries are updated by moving large blocks of pixels at once (which will be called block-level updates), followed by a number of iterations where the boundaries are updated by moving single pixels (which will be called pixel-level updates). This is illustrated in Fig. 2. Here, we introduce a slight variation to the original SEEDS, because rather than alternating block and pixel-level updates, we start with a fixed number of block-level updates, followed by only pixel-level updates.

3.1. Block-level updates

As we only do block-level updates at the beginning of the algorithm, the order of iterations differs slightly from [11]. The block-level updates are only performed at the beginning as a coarse initialization before performing pixel-level updates. The algorithm iterates through all the blocks in the image, and if such a block lies on a superpixel boundary, an update is proposed, as shown in Fig. 3. The update is evaluated based on a statistic of both candidate superpixels.

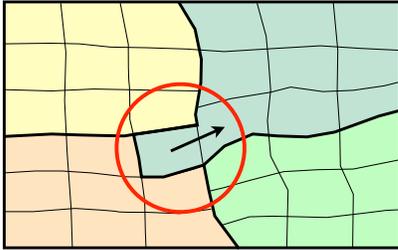


Figure 3. Block-level updates of the superpixel boundaries.

In our case, this statistic is a color histogram of each candidate superpixel. Note that such histograms can be updated very fast (with one increment and one decrement) as explained in [11]. The block of pixels is then moved to the superpixel with the histogram the block most likely belongs to.

3.2. Pixel-level updates

After we have obtained a coarse initialization of the superpixels using block-level updates, the algorithm switches to pixel-level updates in order to refine the superpixel boundaries, as shown in Fig. 4. In this step, the algorithm will make use of color information and depth information.

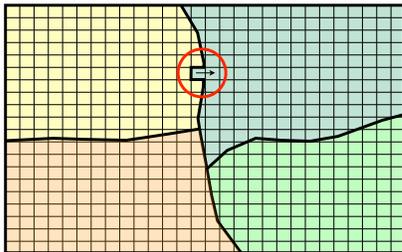


Figure 4. Pixel-level updates of the superpixel boundaries.

Instead of using the histogram as a statistic, a simpler mean-based statistic is used. For each superpixel, the mean color and mean depth are maintained. This can still be implemented efficiently: the algorithm keeps total L , a , b and depth counters (assuming Lab color space), which can be increased/decreased with each pixel-level update. The mean color or depth is then obtained by dividing these counters by the number of elements (pixels) inside the superpixel.

In [11], it is pointed out that a mean-based metric converges to the same result as a histogram-based metric, albeit slower (with more iterations). However, by using the histogram-based metric in the block-level updates, we already come to a very good initialization with few iterations. Thus, using the mean-based metric only in the refinement step will yield an equally good result without too many additional iterations. The use of a mean-based color metric simplifies the incorporation of depth into the metric, considering depth is a one-dimensional, linear metric.

The algorithm iterates through all the pixels in the image, and if a pixel is found to lie on a superpixel boundary, an update is proposed, as shown in Fig. 4. The pixel is then assigned to a superpixel based on a distance metric between the pixel color and depth, and the mean color and depth of the candidate superpixels. This is similar to the approach used in [12], however, we extend the metric in order to deal with missing depth values.

Indeed, if depth information is present in the pixel being evaluated, the following Euclidian distance metric is used:

$$D = \sqrt{(L - L_s)^2 + (a - a_s)^2 + (b - b_s)^2 + (d - d_s)^2}, \quad (1)$$

where L , a and b are the luminance, a and b color values of the pixel, L_s , a_s and b_s the mean luminance, a and b color values of the candidate superpixel, d the depth of the pixel, and d_s the mean depth of the candidate superpixel.

However, if no depth information is present in the pixel being evaluated, the algorithm automatically selects an alternative metric, for which no depth information is required:

$$D = \sqrt{(L - L_s)^2 + (a - a_s)^2 + (b - b_s)^2}. \quad (2)$$

Furthermore, if the pixel has no depth value, it does not count towards the computation of the mean depth of the superpixel which it belongs to.

The pixel-level updating uses the same boundary term as in [11]. This boundary term prefers smooth boundaries, and thus ensures that the decision for assigning each pixel also depends on the labeling of the surrounding pixels.

Once a clean superpixel segmentation is obtained, objects can simply be segmented using a modified connected components analysis. One merely needs to link together the connected superpixels (based on a depth threshold) in order to obtain a segmentation of the different objects or persons in the scene. This will be demonstrated in the next section.

4. Person Segmentation on an Interactive Urban Robot

In order to demonstrate the practical application of recovering missing depth information using Depth SEEDS, we demonstrate the system on a real-world application. In

this application, a stereo pair of cameras is installed on an interactive urban robot, as shown in Fig. 5. The robot is taken outside in order to communicate with humans, asking them to indicate directions and landmarks, as illustrated in Fig.6. This type of interaction usually involves pointing gestures. In order to recognize these gestures, we want to detect the head and arms of the interacting persons.

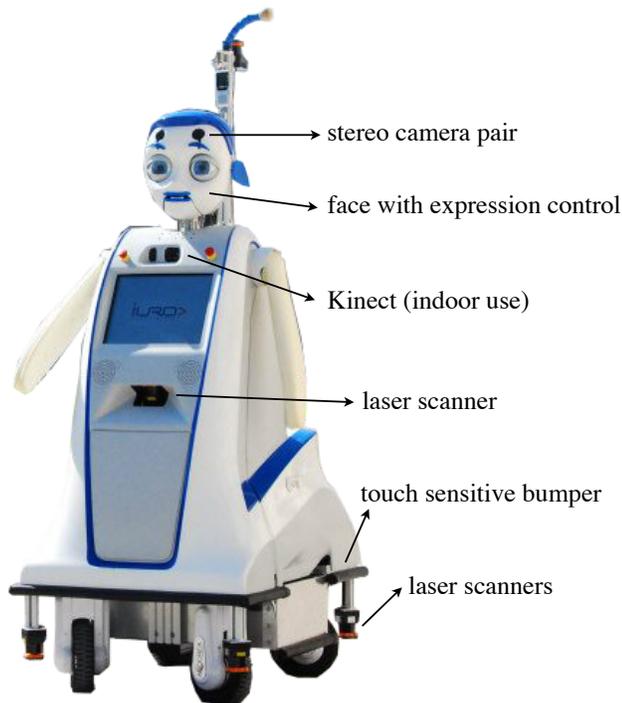


Figure 5. Overview of the sensor devices on the urban robot. A stereo camera pair is installed in the head (above the eye brows) for outdoor segmentation.

To this end, the nearest ‘body’ is segmented based on a modified connected component analysis. Depth SEEDS superpixels are connected together based on the difference in depth between connecting superpixels. This allows for an efficient segmentation of the different objects or persons present in the image.

For our purpose, we select the object closest to the camera. A basic verification is performed to check if it is human, *i.e.*, it is not taller than a reasonable person, and there is a narrower section at the top which represents the head, which is confirmed using a face detector. Subsequently, a simple body part detection is performed by computing the median horizontal position of the body and the head, and then labeling the extremities of the segmentation, as illustrated in Fig. 7.

More precisely, the labeling works as follows: First, the closest connected depth component is selected. Then, the median horizontal position of this component is calculated. In order to estimate an initial labeling of the head, the algo-



Figure 6. Top: detail of the robot head with the stereo camera pair above the eye brows; Bottom: example of the robot interacting with people.

rithm evaluates all the pixels and on this median position, starting from the top of the image downwards. Once this line crosses the segmentation, the head is labeled, until the shoulders are detected as an increase in width of the segmentation.

Once the head is labeled, a new, refined median position is computed based on only the head pixels. Then, the head labeling is repeated based on this new median. If hands are present to the left or the right of the head, they are temporarily marked as stray labels. At this point, the head labeling is confirmed with a face detection.

After the head is labeled, we continue down the median to detect arms. Based on a threshold distance from the median, pointing arms can be labeled. If these arm labels are connected to stray labels, those stray labels are marked as arm as well.

In our case, where we only want to detect pointing gestures, this simple approach has proven more reliable than skeleton fitting or more complex segmentation methods. Based on the arm labeling, we can extract the 3D pointing direction of the arm of the person. This interaction system is demonstrated in the experiments section.

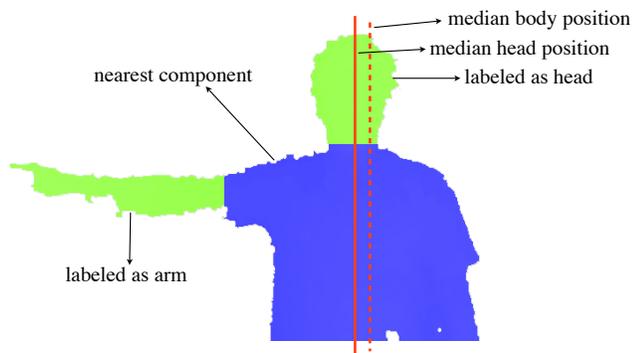


Figure 7. Overview of the head and arm labeling algorithm.

5. Experiments

We first evaluate the ability of our algorithm to recover missing depth values. Then, we demonstrate the functionality of the algorithm on the real-world scenario of an interactive urban robot.

5.1. Evaluation of the Depth Recovery

We evaluate Depth SEEDS by measuring the average depth error after recovering depth values in an image. To this end, we take relatively clean depth images from a Kinect sensor, and introduce synthesized noise at different scales. Two types of noise are introduced. First, where a gap in depth is detected, a black boundary is introduced of thickness W . Secondly, random square holes are introduced throughout the image. These holes have a width and height W , similar to the thickness of the introduced borders. This is illustrated in Fig. 8 and 9.

Depth SEEDS is run on the noisy version of the depth image in order to recover the missing depth data, and the result is then compared to the original clean image. For evaluation, we compute the average depth error (in mm). This experiment is repeated for different values of W , varying the size of the introduced holes. The results of this experiment are shown in Fig. 10. This plot shows that the synthesized noise introduces a large depth error in the depth image, and it shows that Depth SEEDS is able to recover most of this error very efficiently.



Figure 8. Experiment with noise of size $W = 5$ pixels. Left: clean input image. Middle: image with synthesized noise. Right: recovered depth image using Depth SEEDS.



Figure 9. Experiment with noise of size $W = 10$ pixels. Left: clean input image. Middle: image with synthesized noise. Right: recovered depth image using Depth SEEDS.

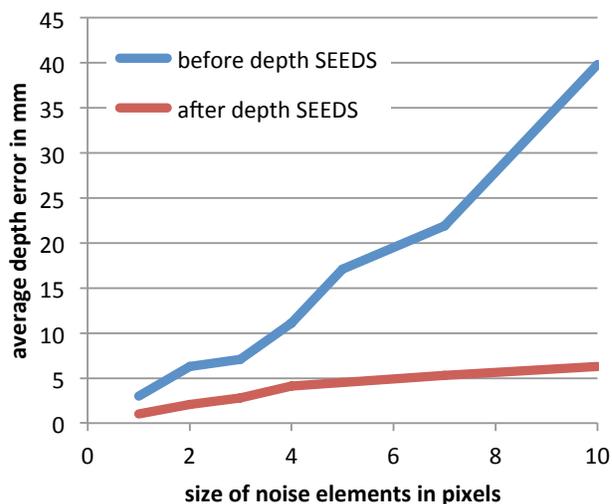


Figure 10. Average depth error after cleaning up noise using Depth SEEDS shows a significant reduction in noise/error in the depth image.

5.2. Real-world Experiment

As explained in Section 4, a real-world experiment was performed with a stereo camera set up mounted on an interactive urban robot. The result of this experiment is shown in a video in supplementary material, and also in Fig. 11 and 12. Usually, using real-time stereo in real-world scenarios proves to be problematic. However, Depth SEEDS is able to recover clean segmentations with lots of detail. This segmentation can then be used to efficiently and robustly label the head and arms of the interacting person.

6. Conclusion

In this paper we have presented an adaptation of the SEEDS algorithm, which not only takes depth information into account on top of color information, but also deals with unknown depth values in an elegant way. The resulting Depth SEEDS algorithm is able to generate clean segmentations based on noisy and incomplete depth images, *e.g.*, from real-time stereo, and to recover the missing depth data fairly accurately.

We have evaluated the accuracy of the recovered depth values, and we have also demonstrated the functionality of



Figure 11. Extract from the experiment video, showing our method is able to recover large portions of missing depth data, and also to recovered significant detail in the finger of the person. Top left: input stereo disparity image. Top right: input color image. Bottom left: segmented output of Depth SEEDS. Bottom right: segmented person with head and left arm labeled.

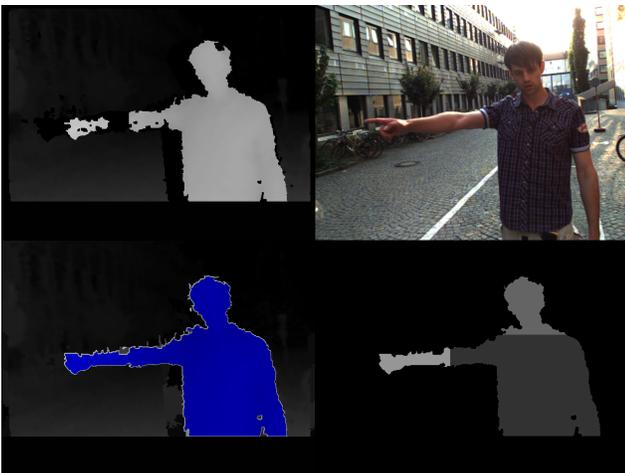


Figure 12. Extract from the experiment video: the arm of the person is cut off in the disparity image, but still recovered in the final segmentation. Top left: input stereo disparity image. Top right: input color image. Bottom left: segmented output of Depth SEEDS. Bottom right: segmented person with head and left arm labeled.

the algorithm in a real-world application. The resulting system is able to achieve real-time pointing interaction with humans in an outdoor scenario, something that is not possible with typical consumer depth cameras.

Acknowledgments.

This work has been in part supported by the European Commission projects RADHAR (FP7 ICT 248873) and IURO (FP7 ICT 248314).

References

- [1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. Slic superpixels. 2010. EPFL Technical Report 149300.
- [2] M. Bleyer and M. Gelautz. Simple but effective tree structures for dynamic programming-based stereo matching. In *International Conference on Computer Vision Theory and Applications*, pages 415–422, 2008.
- [3] X. Boix, J. M. Gonfaus, J. Van de Weijer, A. D. Bagdanov, J. Serrat, and J. Gonzalez. Harmony potentials: Fusing global and local scale for semantic image segmentation. *International Journal of Computer Vision*, 91(1):83–102, 2011.
- [4] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(24):603–619, 2002.
- [5] P. F. Felzenszwalb and D. P. D.P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 2(59):167–181, 2004.
- [6] C. Fowlkes, D. Martin, and J. Malik. Learning affinity functions for image segmentation: Combining patch-based and gradient-based approaches. In *International Conference of Computer Vision and Pattern Recognition*, pages 54–61, 2003.
- [7] B. Fulkerson, A. Vedaldi, and S. Soatto. Class segmentation and object localization with superpixel neighborhoods. In *International Conference on Computer Vision*, pages 670–677, 2009.
- [8] A. Geiger, M. Roser, and R. Urtasun. Efficient large-scale stereo matching. In *Asian Conference on Computer Vision*, pages 25–38, 2010.
- [9] A. Griesser, S. D. Roeck, A. Neubeck, and L. Van Gool. Gpu-based foreground-background segmentation using an extended colinearity criterion. *Vision, Modeling, and Visualization Conference*, pages 319–326, 2005.
- [10] D. Martina, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):530–549, 2004.
- [11] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool. SEEDS: Superpixels extracted via energy-driven sampling. In *European Conference on Computer Vision*, volume 7, pages 13–26, 2012.
- [12] M. Van den Bergh and L. Van Gool. Real-time stereo and flow-based video segmentation with superpixels. In *IEEE Workshop on Applications of Computer Vision*, pages 89–96, 2012.