Fast and Robust Variational Optical Flow for High-Resolution Images using SLIC Superpixels

Simon Donné, Jan Aelterman, Bart Goossens, and Wilfried Philips

iMinds-IPI-UGent *
{Simon.Donne,Jan.Aelterman,Bart.Goossens,philips}@telin.ugent.be

Abstract. We show how pixel-based methods can be applied to a sparse image representation resulting from a superpixel segmentation. On this sparse image representation we only estimate a single motion vector per superpixel, without working on the full-resolution image. This allows the accelerated processing of high-resolution content with existing methods. The use of superpixels in optical flow estimation was studied before, but existing methods typically estimate a dense optical flow field – one motion vector per pixel – using the full-resolution input, which can be slow. Our novel approach offers important speed-ups compared to dense pixel-based methods, without significant loss of accuracy.

Keywords: SLIC superpixels, segmentation, optical flow

1 Introduction

The input of high-resolution content to optical flow estimation algorithms is both a curse and a blessing. On the one hand, such images are able to capture details of the scene and its textures – invisible in lower resolutions – which may offer valuable cues for the optical flow estimation. On the flip side, higher resolution content means longer processing times and – specifically for optical flow estimation – a possible aggravation of the aperture problem. The aperture problem states that movement can only be tracked well near gradients, and then only in the direction of the gradient. As a result, large homogeneous regions typically pose problems for optical flow estimation. In high-resolution content, the size of homogeneous areas increases in terms of absolute pixel count: this requires for example an increase in search window or regularisation window size, further increasing processing times.

A typical solution to this problem is to work in a hierarchical way: first estimate the results on a lower-resolution version of the image, and then gradually refine the estimate on higher resolution versions up to the original input resolution.

We propose a novel framework for optical flow estimation, which allows the exploitation of high-resolution information in the lower resolutions. Contrary to the existing methods, which do content-independent downsampling of the input image, we use SLIC superpixels [1] to arrive at a sparse image representation.

* Part of the research leading to this work was performed within the iMinds HiViz project. Simon Donné is funded by BOF grant 01D21213, and Bart Goossens is a postdoctoral research fellow for FWO.

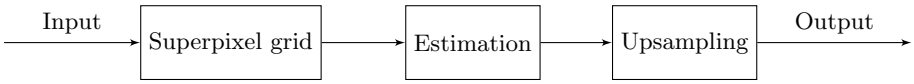

Fig. 1. Overview of the proposed approach: using SLIC, we arrive at a sparse representation of the input image. This sparse representation is processed using a pixel-based methods, and this result is upsampled and scaled to arrive at the final estimate.

This sparse representation is used as an input to existing dense pixel-based methods. An additional benefit of this novel approach is that superpixels are, to some extent, noise-robust. The output optical flow estimate of those existing methods is then be upsampled using the superpixel information, resulting in the output estimate. Our newly proposed method is shown schematically in figure 1. As the major interest lies in the speed-up of the dense pixel-based methods, we also implement each of the steps on the GPU, resulting in significant speed-ups.

After a more detailed overview of existing techniques that use (over)segmented images in section 2, we discuss several superpixel segmentation methods in section 3, amongst others both the original SLIC algorithm and its adaptation for parallel processing, gSLIC. Our proposed superpixel-based approach is explained in detail in section 4. To conclude this text we give experimental results, draw our conclusion and outline future work in respectively sections 5 to 7.

2 Existing Work Using (Over)Segmentation

The idea of content-adaptive downscaling from [10], can also be attained through supervoxels, e.g. in MRI [9]. We will use this idea – superpixels as a sparse image representation – and apply it as an initial step before pixel-based methods are applied.

The use of superpixels as a pre-processing step is familiar in object segmentation and recognition, where it is often an initial step [2, 12, 13]. Optical flow estimation methods that estimate a single motion vector per segment generally segment the scene into objects. This requires specially adapted methods which can work on segmented images instead of pixels [3, 14]. Novel to our approach is that we will estimate a single motion vector per superpixel without requiring such segment-based logic.

The alternative use of superpixels in optical flow estimation is the estimation of one motion vector per pixel, a dense pixel-based field, while using a superpixel segmentation to guide this estimation [4, 7], for example by performing error aggregation over superpixels or by regularising pixels within the same superpixel. These methods still work on the full-resolution input image. In contrast, our new approach applies a pixel-based technique to the superpixel grid, drastically lowering the resolution that is being processed.

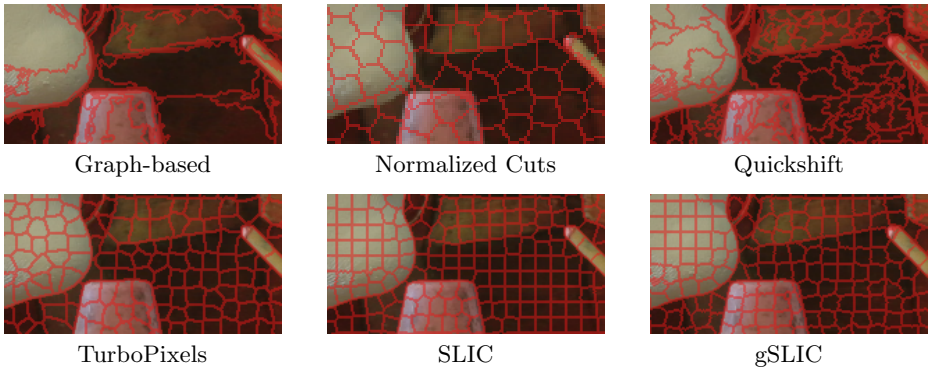

Fig. 2. Results of various superpixel segmentation methods. The input image was a detail of the Art image from [17].

Graph Based	Normalized Cuts	Quickshift	TurboPixels	SLIC	gSLIC
1.428 s	7.45 min	15.18 s	71.7 s	0.503 s	45 ms

Table 1. Execution times for the various methods on an image of 1390×1110 (Normalized Cuts was run on the half-resolution input). The used implementations were publically available (C++ or MATLAB). gSLIC was implemented using Quasar [8].

3 Existing Superpixel Methods

We take a *superpixel* to be a cluster of pixels that results from oversegmenting an image: groups of pixels that are similar in size (surface area) but not containing major edges. The term *oversegmentation* is used to indicate that they do not correspond with physical objects on a one-to-one basis: the image of a physical object is usually still divided into multiple superpixels. We discuss several algorithms for estimating superpixels, which are illustrated in figure 2.

Felzenszwalb's graph-based segmentation [6] is an efficient and popular segmentation algorithm. It uses a single scale parameter to influence the segment size. However, the actual size and number of segments varies greatly depending on the local content as seen in figure 2.

The normalized cuts-approach from [16] is very slow to execute (table 1). It was unable to process the full-resolution 1390×1110 image in an acceptable time, hence results were generated on the half-resolution input.

Quickshift [18] is based on an approximation to kernel-based mean-shift. By processing pixels as points in a five-dimensional space (three colour channels and two location coordinates), mean-shift in this five-dimensional space results in a segmentation of the original image as in figure 2.

Turbopixels [11] are based on geometric flows, providing boundary-conforming superpixels while keeping under-segmentation in check through a compactness constraint. It is a much faster method than Normalized Cuts, providing visually better results, but is not quite as fast as the other methods (see table 1).

SLIC (Simple Linear Iterative Clustering [2]) performs k-means clustering in the same five-dimensional color-location space as QuickShift, performing correspondence searches only locally. By controlling the seeds for the k-means clustering, it is very straightforward to impose a desired size on the superpixels, while a compactness parameter handles the trade-off between color and location, as in QuickShift. While the k-means clustering results in superpixels which are connected in the five-dimensional colour-spatial space, superpixels are assumed to be spatially connected. A post-processing step enforces spatial connectivity.

SLIC superpixel estimation is very amenable to parallelisation: an adaptation called gSLIC allows the clustering to be done efficiently on a GPU [15]. The authors of [15] mention a speed up of 10x to 20x, which our findings corroborate. Its results are shown in figure 2, and visually they appear very similar to the results from the original SLIC (not quite identical because of slight changes for better parallelisation).

We will use the gSLIC estimation of superpixels in our proposed workflow: this is a novel interpretation of the oversegmentation results as a dimensionality reduction resulting in a near-regular grid. gSLIC is an extremely fast segmentation that is of low implementational complexity, and its nature makes it easy to arrive at a superpixel grid of user-specified dimensions (by spawning the k-means seeds along the required grid).

3.1 gSLIC superpixels

We outline gSLIC's workings and the interpretation of the resulting superpixel grid as a sparse image representation in this section. The image is first divided into a regular grid of the desired size (as in figures 3 and 4[1]). The position of a superpixel on this grid is used as its label and denoted by (x_s, y_s).

When estimating superpixels using gSLIC, each pixel contains a label indicating which superpixel it belongs to; superpixels are described with a colour and a location, defined as the respective averages all pixels that belong to it.

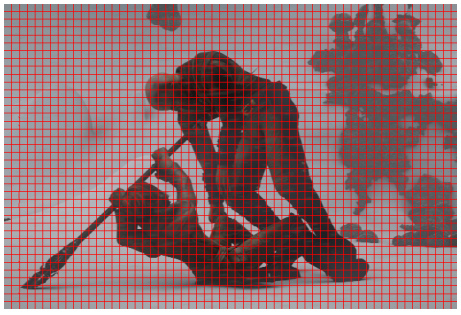

Fig. 3. Initialisation for gSLIC.

Fig. 4. Detail of the initialisation.

[1] Copyright Blender Foundation (www.sintel.org).

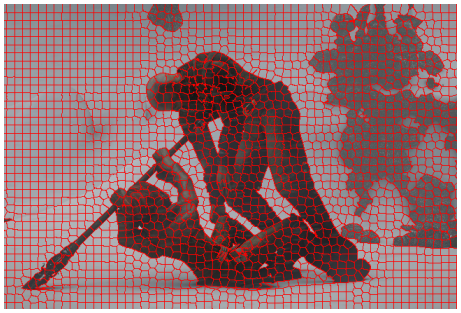

Fig. 5. Final clustering by gSLIC.

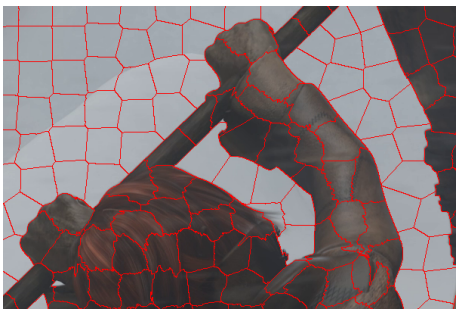

Fig. 6. Detail of the final clustering.

After the initialisation, k-means clustering is performed: in each iteration, each pixel selects the most fitting label (i.e. the closest cluster center in the 5-dimensional space of colour and location) after which the cluster centers are updated with the new description based on the new labels. A compactness value weights the color distance between a pixel and a prospective cluster against their Euclidean distance in the image plane - because Euclidean distances are used, colours are represented in the CIELAB colour space.

3.2 (g)SLIC in our proposed workflow

We intend to use SLIC superpixels for dimensionality reduction of the input image. We assume that the superpixels are very small in comparison to the content: this implies that the superpixels will be largely rectangular and uniform as the content is locally uniform/homogeneous. As the resulting segmentations for SLIC and gSLIC are extremely similar, we prefer gSLIC for its faster execution.

We do not enforce label connectivity nor do we fuse small clusters because our superpixels are, by and large, uniform. An example superpixel clustering as we propose to use for the input image is shown in figure 7. Note that the superpixels form a largely regular grid, except near the content edges, which they adhere to nicely.

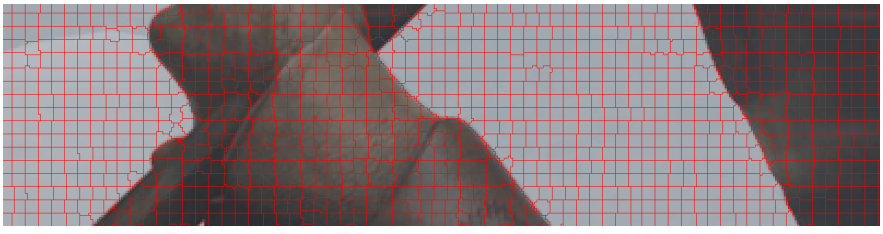

Fig. 7. Example of a superpixel grid such as we propose to use.

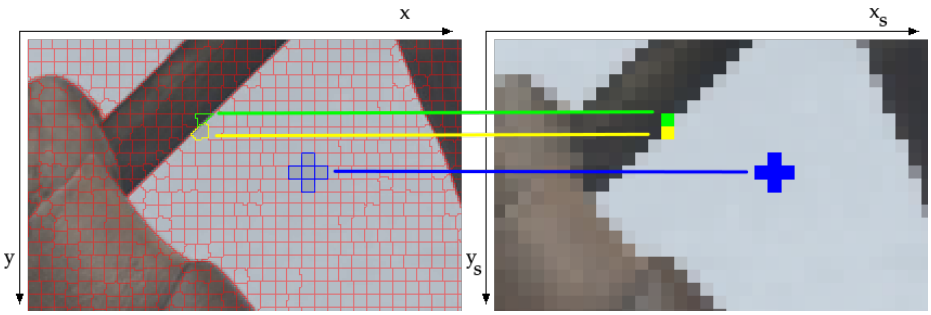

Fig. 8. The full-resolution input image on the left is subdivided into superpixels (red). Each of these superpixels corresponds to a location on the superpixel grid (as indicated by the coloured connections). The superpixel grid is visualized as an image by assigning the colour description of a superpixel to its location in the superpixel grid, resulting in the superpixel grid image shown on the right.

4 Variational Optical Flow on the superpixel grid

We now outline, in detail, our novel workflow for optical flow estimation on high-resolution images: the superpixel clustering followed by dense pixel-based processing of the superpixel grid, and finally the upsampling using the superpixel labeling information.

(gSLIC) superpixel clustering and creation of the superpixel grid:

Each of the superpixels has a two-dimensional coordinate corresponding to its location on the superpixel grid, as well as a colour. After superpixel estimation, each pixel $\boldsymbol{p}$ on the sampling grid of the original input images I_k has a label $(x_{k,s}(\boldsymbol{p}), y_{k,s}(\boldsymbol{p}))$ corresponding to the superpixel it belongs to. This is illustrated in figure 8: the input image on the left is shown with its estimated superpixels.

The labels of the superpixels (i.e. their initial locations) can be used to create an image of the superpixel grid as in figure 8: each location in the superpixel grid image has the colour from the description of the corresponding superpixel. The superpixel grid is only a fraction of the size of the original image: if we impose the superpixels to be roughly squares with an area of 2^2, it only has half the width and half the height.

Processing the superpixel grid with the pixel-based method

In the case of optical flow estimation, we create a superpixel grid for both input images. We process these two superpixel grids with an existing, pixel-based method. For the purpose of this paper, we have chosen a total-variation based optical flow estimation [5]. The output is an estimate of the optical flow between both superpixel grids, as shown in the top right of figure 9.

The superpixel grid.

The superpixel grid optical flow estimate.

Detail of the original input image.

The upsampled optical flow estimate.

Fig. 9. Illustration of the proposed method: the superpixel grids (top left) for both input images are processed by a dense pixel-based method, resulting in an optical flow estimate for the superpixel grids (top right). After upsampling (bottom right), we arrive at an optical flow estimate for the original input image (bottom left).

Upsampling the flow estimate using the superpixel labelling

Finally, the flow estimate on the superpixel grid is upsampled: each pixel in the high-resolution input image has a label which can be used to look up the estimated flow for the superpixel grid. In the case of, e.g., optical flow, the estimate also needs to be scaled (horizontal offsets must be multiplied by the average width of all superpixels, and ditto for vertical offsets).

Note that the sampling grid underlying the superpixel grid is only nearly regular, while the existing pixel-based methods assume a completely regular grid. Because the superpixel grid is very close to regular, and because this irregularity evens out over relatively small neighbourhoods, we expect this to have only a minor influence. Indeed, as the results in section 5 indicate, the loss of accuracy is small. Furthermore, as we have noted earlier, the superpixels are to some extent noise-robust, and this results in a net gain of accuracy for even small amounts of noise.

Figure 9 shows an example result of the upsampling: the top left shows the superpixel grid, in the top right the superpixel grid flow estimate is shown and in the bottom the original input resolution and the resulting flow estimate are shown. We see that the resulting high-resolution output follows the image content well: a more quantitative comparison will be done in the next section.

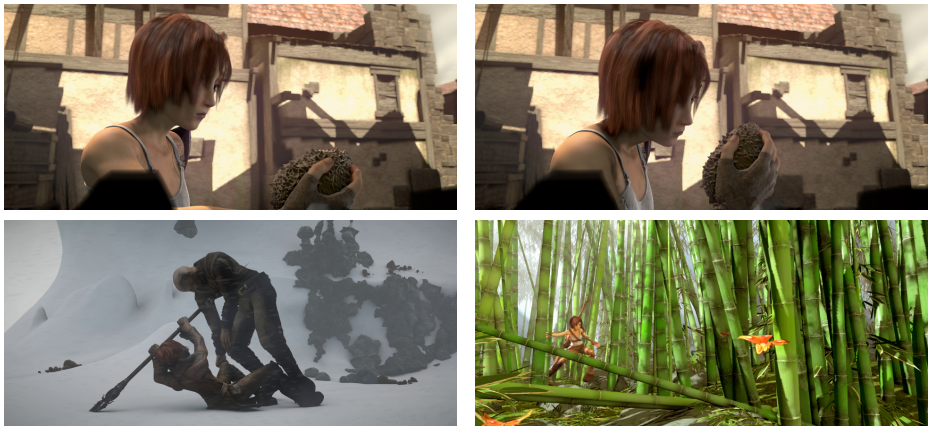

Fig. 10. Test images for evaluating the proposed method.

5 Experiments and results

We perform both the full-resolution pixel-based approach (i.e. the existing approach on the full-resolution input image) and the superpixel-grid approach on the test images from figure 10 in full 4K-resolution. The images are perturbed by various levels of noise as shown in figure 11. The noise is AWGN with a standard deviation of 0.0 to 0.1 when expressing pixel values in the range $[0, 1]$ (with saturation on the boundaries).

We find that SLIC superpixels (and superpixels in general) are noise-robust: figure 11 shows how the clustering of the input image with an increasing amount of noise changes only slightly - even up to a high noise level (while we have not found it necessary, adjusting the compactness of the superpixels can help as well).

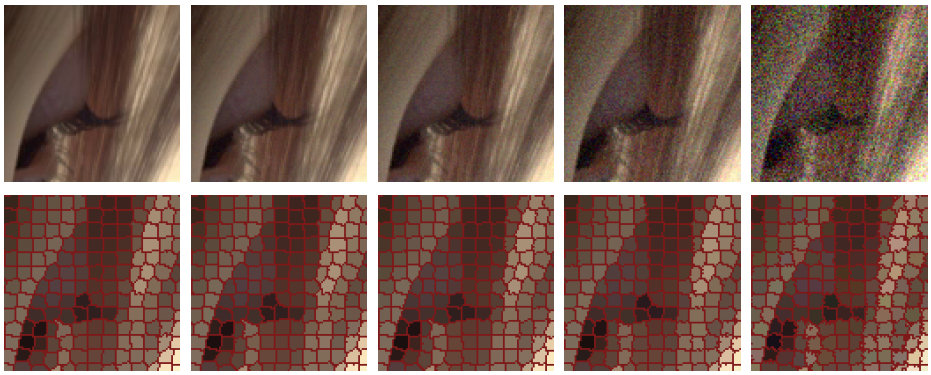

Fig. 11. Illustration of the noise levels and their impact on the estimated superpixels.

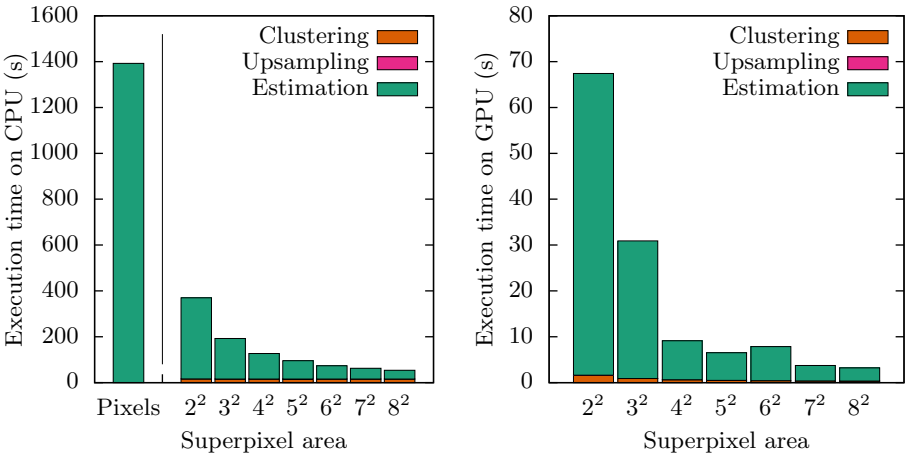

Fig. 12. Timing data for the proposed method in function of superpixel size. The used CPU was an Intel Core i7-980X and the GPU was an nVidia GeForce GTX 770.

Speed of the proposed method

Using segments on a near-regular grid as a sparse image representation, as we propose in this work, has the distinct advantages of reducing the resolution of the input to the pixel-based method and hence its execution time. Figure 12 displays the total execution time of both the existing dense pixel-based method on the original input image, as well as our proposed approach for various superpixel sizes. The total execution times are divided into the three steps of the approach: the superpixel clustering, processing the superpixel grids with the pixel-based methods and finally the upsampling to the original input resolution. We see that the time spent clustering is small, and the time required for upsampling is so small as to be invisible on the graph: its impact on the processing time is negligible. As the superpixel size increases, the processing time lowers: as the superpixels grow larger, the dimensions of the superpixel grid image lower – the dense pixel-based optical flow estimation still constitutes the largest part of the processing time.

Thanks to the Quasar platform [8] we have both a CPU and GPU implementation of the approach. Using superpixels about 3 by 3 pixels, we achieve a speed-up factor of more than 40 compared to the existing pixel-based method on the CPU (it was not able to be run on the GPU because of memory restrictions on the used model of GPU).

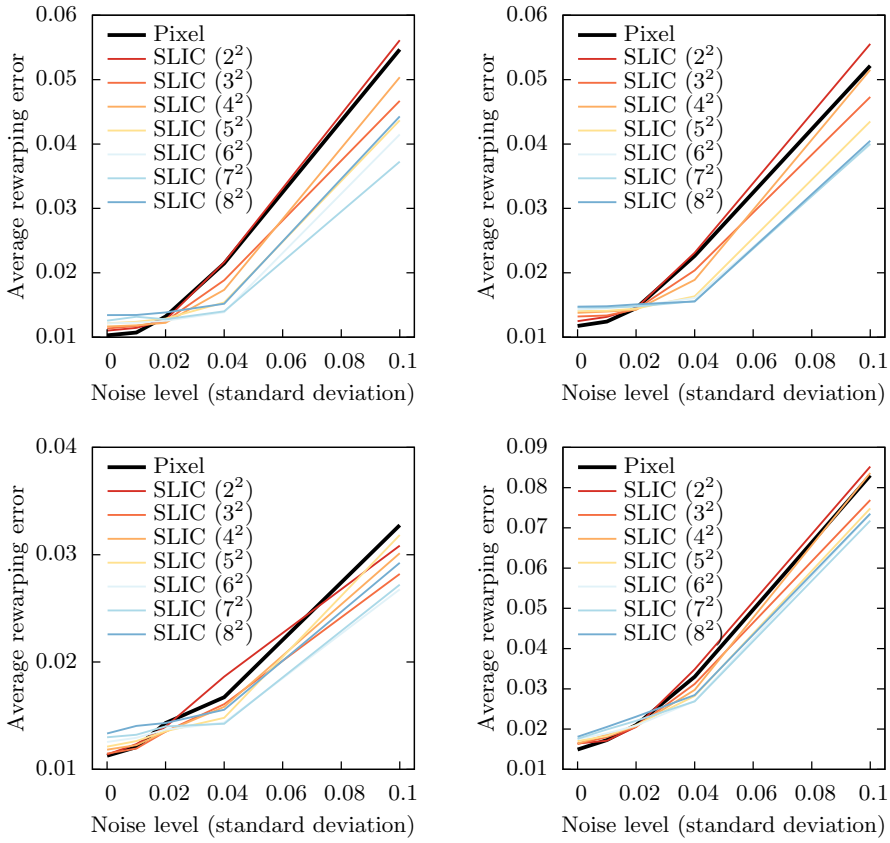

Fig. 13. Noise impact for the test images shown in figure 10, displayed in the same order as the test images themselves.

Accuracy of the proposed method

In order to quantify the accuracy of the proposed methods, the (possibly noisy) inputs are used to estimate an optical flow field between the two views. This optical flow estimate is then used to warp the original (*noise-free*) input images onto each other: the resulting average error is used to evaluate the quality of the estimate. In this way, the actual noise on the input images does not affect the error directly (only through its impact on the optical flow estimate).

The results are shown in figure 13. We see that the dense pixel-based method has a slight quality advantage over the proposed method based on superpixels, but only in the absence of noise. However, as soon as the input images are perturbed by a small amount of noise, our proposed approach benefits from the robustness of the superpixels, outperforming the pixel-based method. As the superpixel size increases, so does the initial loss of accuracy; this is offset by the increased noise robustness: the error increases less as the noise level is raised.

6 Conclusion

In this paper, we have demonstrated the use of superpixels as a way of reducing the resolution of the input and upsampling the output of an algorithm. We thus showed that the dense pixel-based method can be easily and successfully applied to a superpixel grid, and need not be troubled by the fact that it is now working on a superpixel grid. In this way, we bridge a gap between high-resolution content and (possibly slow or computationally expensive) state of the art techniques.

By limiting ourselves to superpixel grid application of an existing pixelgrid technique, we forfeit the full potential of superpixel-based methods. On the other hand, this approach opens up the full range of existing pixel-based methods. When using superpixels in order to reduce the resolution of the input, the underlying grid is not entirely uniform. However, we illustrate that – given that the superpixels are sufficiently small so that the grid is *close* to regular – this has limited impact on the resulting estimate.

On a final note, this way of processing high-resolution data can be easily expanded to higher-dimensional data. Specifically, high-resolution video data can be processed in the same way. In that case we can either subsample along the temporal dimension or not, depending on the temporal resolution and the temporal content of the input.

7 Future work

While the proposed workflow allows the continued use of existing methods on high-dimensional data using an abstraction layer which hands the algorithms lower-resolution data, the step will eventually have to be made towards superpixel-based approaches: methods which explicitly take the non-regular grid into account. This will allow those algorithms to process increasingly higher-dimensional data without any additional effort aside from pre-processing by a superpixel method while keeping a theoretically optimal framework and still limiting the amount of data handed to the pixel-based method.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Ssstrunk, S.: SLIC Superpixels. Tech. rep., EPFL, EPFL (2010)
2. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Ssstrunk, S.: SLIC Superpixels Compared to State-of-the-art Superpixel Methods. IEEE Transactions on Pattern Analysis and Machine Intelligence (2012)
3. Van den Bergh, M., Van Gool, L.: Real-time stereo and flow-based video segmentation with superpixels. In: Applications of Computer Vision (WACV), 2012 IEEE Workshop on. pp. 89–96 (Jan 2012)
4. Chang, H.S., Wang, Y.C.: Superpixel-based large displacement optical flow. In: Image Processing (ICIP), 2013 20th IEEE International Conference on (Sept 2013)
5. Drulea, M., Nedevschi, S.: Motion estimation using the correlation transform. Image Processing, IEEE Transactions on 22(8), 3260–3270 (2013)

6. Felzenszwalb, P., Huttenlocher, D.: Efficient graph-based image segmentation. International Journal of Computer Vision 59(2) (2004)
7. Gkamas, T., Nikou, C.: Guiding optical flow estimation using superpixels. In: Digital Signal Processing (DSP), 2011 17th International Conference on. pp. 1–6 (July 2011)
8. Goossens, B., Vylder, J.D., Philips, W.: Quasar - A new heterogeneous programming framework for image and video processing algorithms on CPU and GPU. In: 2014 IEEE International Conference on Image Processing, ICIP 2014, Paris, France, October 27-30, 2014. pp. 2183–2185. IEEE (2014), http://quasar.ugent.be
9. Heinrich, M.P., Jenkinson, M., Papież, B.W., Glesson, F.V., Brady, S.M., Schnabel, J.A.: Edge- and detail-preserving sparse image representations for deformable registration of chest mri and ct volumes. In: Proceedings of the 23rd International Conference on Information Processing in Medical Imaging. pp. 463–474. IPMI'13 (2013)
10. Kopf, J., Shamir, A., Peers, P.: Content-adaptive image downscaling. ACM Trans. Graph. 32(6) (Nov 2013)
11. Levinshtein, A., Stere, A., Kutulakos, K.N., Fleet, D.J., Dickinson, S.J., Siddiqi, K.: Turbopixels: Fast superpixels using geometric flows. IEEE Trans. Pattern Anal. Mach. Intell. 31(12) (Dec 2009)
12. Mori, G.: Guiding model search using segmentation. In: Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on. vol. 2 (Oct 2005)
13. Mori, G., Ren, X., Efros, A., Malik, J.: Recovering human body configurations: combining segmentation and recognition. In: Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on. vol. 2 (June 2004)
14. Nishigaki, M.: Color segmentation-based optical flow computation and motion segmentation (2008)
15. Ren, C.Y., Reid, I.: gslic: a real-time implementation of slic superpixel segmentation. Tech. rep., University of Oxford, Department of Engineering Science (2011)
16. Ren, X., Malik, J.: Learning a classification model for segmentation. In: Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on (Oct 2003)
17. Scharstein, D., Pal, C.: Learning conditional random fields for stereo. In: Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on. pp. 1–8 (June 2007)
18. Vedaldi, A., Soatto, S.: Quick shift and kernel methods for mode seeking. In: Computer Vision – ECCV 2008. Lecture Notes in Computer Science, vol. 5305 (2008)