Dynamic Super-Rays for Efficient Light Field Video Processing

Matthieu Hog, Neus Sabater, Christine Guillemot

Dynamic Super-Rays for Efficient Light Field Video Processing

Matthieu Hog[1,2], Neus Sabater[1], Christine Guillemot[2]
[1]Technicolor R&I [2]INRIA, Rennes, France

Abstract

One challenging problem with light field video editing is the dreadful volume of data to process. Image and video processing frequently rely on over-segmentation methods to reduce the computational burden of subsequent editing tasks. In this paper, we present the first approach for video light field over-segmentation called dynamic super-rays, which can be seen as temporally and angularly consistent superpixels. Our algorithm is memory efficient and fast. The results show timing close to editing time by leveraging GPU computational power.

1. Introduction

By imaging a scene from different viewpoints, a light field enables advanced processing such as depth estimation, refocusing, parallax shift, or super-resolution. The acquisition of such content is typically done with 3 classes of devices. Plenoptic cameras [21, 17] use a microlens array in front of a camera sensor to re-arrange by direction the light rays coming from the camera main lens. Camera arrays [34] are composed of a rig of cameras, often organized on a regular grid. Finally, camera gantries (e.g. Stanford's Lego gantry[1]) have a mechanical system to move a single camera along a plane, taking photos at regular intervals. Despite being aimed at different applications and having quite diverse spatial and angular resolutions, the content captured by these devices is often described using a representation called lumigraph [12], which describes all light rays as a set of views taken with a regular baseline on a single plane.

Currently, most of light field content used in research is static. This is perhaps due to the difficulties of capturing a dynamic light field for either of the aforementioned devices. Specifically, camera gantries are by design unable to record dynamic light fields, and the available plenoptic cameras are either limited to static light fields[2] or cannot easily produce a lumigraph[3]. Camera arrays require to build a synchronized camera acquisition system, which is a real technical challenge.

However, latest advances in light field video acquisition systems [8, 26] show that it is possible to capture quite voluminous light fields in real time. The problem of efficiently processing the captured 5D light field content is then twofold. First, the amount of data to handle, already a problem for static light fields, reaches a critical point for video light fields. As a consequence, computational efficiency is a core aspect in many light fields processing tasks, such as editing. Secondly, to enable editing via user interaction on a single key frame, algorithms must consistently propagate the edits angularly and temporally to the rest of the dynamic light field.

In this paper we present an over-segmentation scheme that exploits the light field redundancy in both the temporal and angular dimensions and leverages GPU computational power to enable easy and fast light field editing from a single reference view and frame. The proposed over-segmentation approach generalizes the concept of *super-rays*, introduced in [14] for static light fields, to *dynamic super-rays* for video light fields. Super-rays, seen as the light field analog of superpixels [1], are a grouping of rays captured in different views of the light field that are similar in appearance and coming from the same scene area.

Several constraints are taken into consideration in the design of the proposed method. First, the approach is parallelizable to take advantage of a GPU implementation. Second, it processes different frames sequentially and on-the-fly, so the memory footprint is reasonable, in contrast to methods operating on sliding windows. This is mandatory as the amount of data per frame is much greater than a conventional 2D video. Finally, super-rays are consistent across views *and* across frames in the temporal dimension.

Specifically, our contributions are:

- An end to end approach to create an angularly and temporally consistent light field over-segmentation that we call dynamic super-rays
- A new update term for creating and deleting superpixels and super-rays specially tailored for dynamic content

[1] http://lightfield.stanford.edu/lfs.html
[2] https://lytro.com
[3] http://raytrix.de type 2.0 plenoptic cameras with trifocal microlenses

- A new strategy for creating a temporal over-segmentation that is consistent with the scene movement

2. Related Work

We focus on the two areas of the literature that are related to the proposed method, namely image and video over-segmentation and light field editing.

Image and Video Over-Segmentation: From the seminal work of [23] that introduced the term superpixel, many approaches have been proposed for still image over-segmentation. Because superpixels are often used as a pre-processing step to speed up other algorithms, it is not surprising to see that superpixel methods yielding low run-time are more popular. Historical approaches like the multi-scale watershed segmentation [18], turbopixels [15] or quick shift [30] are outperformed by modern algorithms, in terms of performance over complexity. Among them, we can mention Superpixels Extracted via Energy-Driven Sampling (SEEDS) [28]. Starting from a regular lattice, the approach iteratively computes the cost of reassigning pixels blocks to neighborhoods superpixels. The cost function is designed to encourage superpixels with uniform colors. The reassignments are carried in a multi-scale fashion, starting from the coarsest level. Unfortunately, the process involves a lot of stochastic iterative operations, thus cannot take advantage of massively parallel computation. The method called Simple Linear Iterative Clustering (SLIC) [1] adapts the Loyd's algorithm for k-means clustering of image pixels in terms of spatial and color distance in the Lab space. In order to speed up the algorithm, the clusters are contained in a rather small local window corresponding to the size of the initial uniform seeding. Arguably, the strongest advantage of SLIC is its ability to leverage massively parallel computation. Because each assignment and update step can be carried out efficiently on the GPU, to the best of our knowledge, SLIC is the only approach providing real-time performances [22]. Many variations of this algorithm have been proposed [2, 16, 31], but all loosing this computational advantage.

The problem of over-segmentation for multiview is a rather unexplored topic, most of the proposed approaches focus on computing a consistent segmentation rather than an over-segmentation. These approaches usually generate superpixels separately on each view and group them into meaningful object segments afterwards [6, 19].

On the contrary, video over-segmentation is a more researched topic. We can distinguish several video over-segmentation categories. The first one assumes the entire video sequence to be processed as a whole. For instance, in [1], the temporal dimension is treated in the same way as the spatial dimensions in order to create a single volume, where volumetric superpixels are computed. Whereas in [10], superpixels are computed separately on each frame, then temporal correspondences between superpixels are established using an affinity metric in order to extract object segments. The second and more popular category aims at computing the segmentation consistently, from a frame to another. Several approaches have been proposed. In [7] a pre-computed dense optical flow is used along with a Gaussian process to update the labeling from a frame to another. The superpixels deletion or creation is done by inference using the same Gaussian process. In [29], the framework proposed in [28] is extended to videos. When a new frame arrives, the same moves are performed to adapt for the new frame geometry, using the previous frame assignments. Creation and deletion of superpixels are done by looking at color histogram distances. In [24], the most related work to ours, dynamic SLIC superpixels are computed in a sliding window of 20 frames. A dense flow is used to propagate the assignment from a frame to another and several SLIC iterations are run. The centroid color is shared between corresponding superpixels on several frames of a sliding window. The superpixel update criteria is solely based on the superpixel size evolution. Unfortunately, none of the aforementioned approaches are readily applicable for video light field over-segmentation for different reasons. Loading all, or a large amount of the frames of the video sequence as in [24, 1] is prohibitive in the case of a light field. Performing a late merge of frames superpixels as in [10] does not provide any temporal consistency. Methods taking the assumption that the temporal dimension is densely sampled, as in the temporal propagation step of [7], will most likely fail in our case, where objects with wide motion are involved. Finally, all the approaches relying on a stack of granular operations as in [29], are not suitable for a GPU implementation, necessary to handle the large volume of data in editing time.

Light field editing: Because segmentation is the first step of many editing algorithms, it is natural to see most light field editing papers being centered around this topic. A level set method is proposed in [5] to extract objects which are assumed to be organized in layers in the scene. In [32], the authors present a more flexible albeit more computationally expensive method to compute arbitrary object segments. The approach uses user scribbles on a reference view to learn a joint color and depth object model that is used to infer a label for each ray of the light field. These label assignments are further regularized inside and between views, taking into account depth occlusions. In [20], a dense angular sampling is assumed and the authors build a graph with 4D anisotropic connectivity in order to use graph cuts to assign a label to each ray. The color and depth model is estimated using a SVM technique. In these two approaches, because the regularization does not scale well with the size of the input light field, the running times are rather high. To solve this issue, in [13] rays coming from the same scene

point are represented in a single node on a ray-based graph structure, before using a depth and color Gaussian mixture model in a graph cut scheme. This reduces significantly the size of the graph and scales well with the number of input images. However, the quality and the run-time depends on the quality of the dense depth estimation on all views, which can be quite expensive to compute. Furthermore, as for conventional 2D images, the approach does not scale with the spatial resolution of the light field.

To overcome this problem, recent work focused on providing a light field over-segmentation. In [35], a depth estimation is used to propagate an initial SLIC over-segmentation on a reference view to all the views of the light field. The initial segmentation is then iteratively refined by optimizing an energy function based on segmentation smoothness inside and between the views along with a color, position and disparity uniformity prior. In [14], a method to compute angularly consistent superpixels, named super-rays is proposed. The approach does not require a dense depth estimation and focuses on speed and parallelism but still provides satisfactory segmentations. This last algorithm has shown to be a good trade-off between speed and accuracy for still light fields, so we consider it as a starting point for our over-segmentation method for light field videos. We summarize it in Sec. 3.

3. Static super-rays summary

In this section we briefly describe and give the main notations of the super-rays algorithm for static light fields. Further details can be found in [14].

Notations: Let r be a ray of the light field LF, and $(\mathbf{s}, \mathbf{x})$ its coordinates using the two plane parametrization [12], where $\mathbf{s} = (s, t)$ and $\mathbf{x} = (x, y)$ are the angular and spatial coordinates respectively. Besides, each ray has an associated $CIELab$ color value Lab_r. Let $\mathbf{x}' := \mathcal{P}^d_{\mathbf{s}'}(\mathbf{x})$ be the spatial pixel position in view $\mathbf{s}'$ imaging the same scene point, at a distance d, as $\mathbf{x}$ in view $\mathbf{s}$. This is, $r = (\mathbf{s}, \mathbf{x})$ and $r' = (\mathbf{s}', \mathbf{x}')$ are corresponding rays imaging the same scene point in different views.

Given a light field, *super-rays* are all the perceptually similar rays corresponding to the same scene area. That is, the mapping $A\colon LF \subset \mathbb{Z}^4 \rightarrow \mathbb{Z}$, such that each ray r is assigned a super-ray label c is computed. Let SR_c be the set of rays r such that $A(r) = c$.

The super-ray computation is inspired by SLIC [1] and has the same main steps. One major difference is that each super-ray SR_c is characterized by a centroid ray r_c with angular coordinates corresponding to the reference view $\mathbf{s}_c$ and a depth d_c associated to it.

Initialization: The spatial positions $\mathbf{x}_c$ of the centroid rays are initialized on a regular grid of step S in the reference view. The corresponding $CIELab$ color values on such positions are the initial color values of the centroid rays Lab_{r_c}, and the depth d_c of each centroid ray r_c is estimated via block-matching.

Assignment step: At each iteration, each ray $r = (\mathbf{s}, \mathbf{x})$ of the light field is assigned a super-ray label $A(r)$. First, the depth estimated in the previous step is used to compute $r'_c = (\mathbf{s}', \mathcal{P}^{d_c}_{\mathbf{s}'}(\mathbf{x}_c))$, the corresponding rays of r_c. Then, each ray in a neighborhood $N_S(r'_c)$ of size S around r'_c is assigned to the super-ray SR_c if it minimizes the color and spatial distances:

$$A(r) = \arg\min_c \left(\Delta_{Lab}(r, r_c) + \lambda \, \Delta_{xy}(r, r'_c) \right), \qquad (1)$$

where $\Delta_{Lab}(r, r_c) = ||Lab_r - Lab_{r_c}||^2$, $\Delta_{xy}(r, r'_c) = ||\mathbf{x} - \mathcal{P}^{d_c}_{\mathbf{s}'}(\mathbf{x}_c)||^2$ and λ is the parameter weighting the color and spatial distances.

Update step: Following the assignment step, the spatial coordinates of the ray centroid and its corresponding Lab values are updated. Indeed, the new color value of r_c is the average of the color values of all rays in SR_c and the new spatial coordinates are the average coordinates of all light rays $r = (\mathbf{s}, \mathbf{x})$ in SR_c projected on the reference view using the depth d_c:

$$\mathbf{x}_c = \frac{1}{|SR_c|} \sum_{r \in SR_c} \mathcal{P}^{d_c}_{\mathbf{s}_c}(\mathbf{x}). \qquad (2)$$

Note that the angular coordinates $\mathbf{s}_c$ of the centroid rays do not change along the iterations, while the spatial coordinates are updated.

Iterations: As in SLIC, the two previous steps are repeated until the centroids are stable. This happens within 10 to 15 iterations. A cleanup step is optionally run to reassign labels to disconnected rays.

4. Dynamic super-rays

The proposed approach is inspired from techniques proposed to generate temporally consistent superpixels [24, 7], that can be decomposed into three main steps: (i) initialize the current frame segmentation by temporally propagating the segmentation of previous frames, (ii) adapt the segmentation to changes in geometry and (iii) create and delete segments to take into account occlusions and objects entering or leaving the scene.

Our algorithm is summarized in Alg. 1 and illustrated in Fig. 1.

4.1. Sparse temporal propagation

Computing a dense and accurate optical flow for light fields can be a quite tedious task, especially when memory and time requirements are taken into account. Moreover, because super-rays embed a depth information per segment, the problem we aim to resolve is a scene flow estimation problem. That is, we aim to find the displacements of 3D

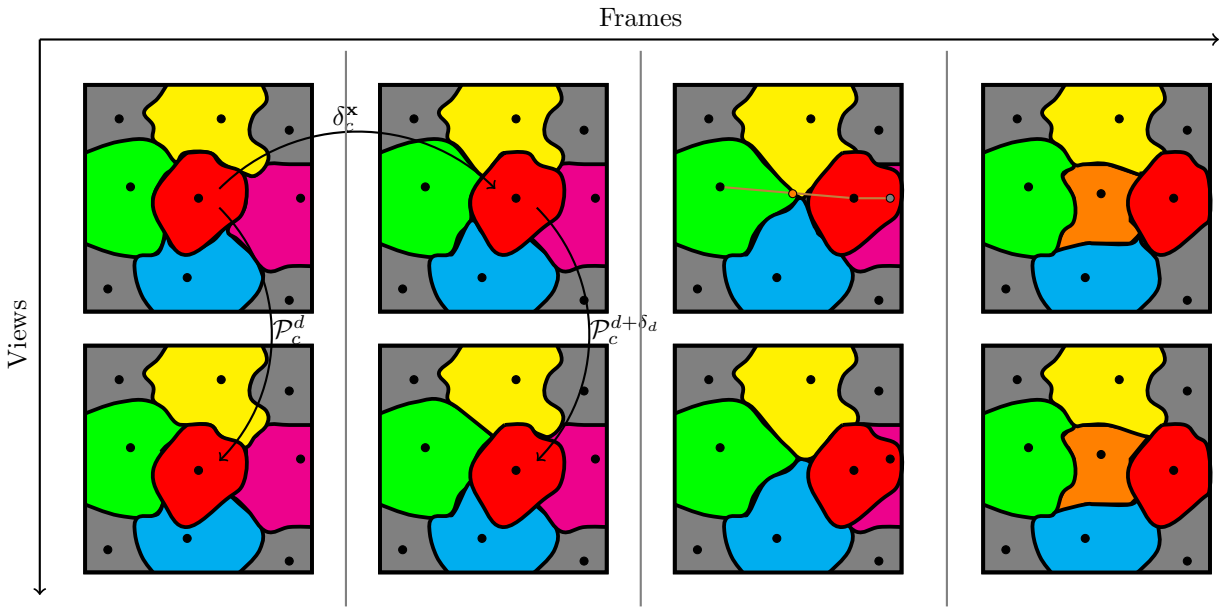

Figure 1. Illustration of our algorithm in a simple case. The red foreground super-ray is tracked over the consecutive frames of a 2×1 light field. Other super-rays do not move since the background is static. The depth d is used to enforce angular consistency from a view to another, while the scene flow $(\delta^{\mathbf{x}}, \delta^d)$ guarantees temporal consistency. On the third frame, the moving red super-ray becomes too close of the pink super-ray and too far from the green one, triggering the creation of the orange super-ray and the deletion of the pink one on the next frame.

Algorithm 1: Dynamic super-ray algorithm

Data: Input light field frame LF^f
Result: Super-ray assignments A^f
if f == *first frame* **then**
  Compute A^f as in [14]
else
  Move centroids with $(\delta_c^{\mathbf{x}}, \delta_c^d)$ (Sec . 4.1)
  Delete and create centroids (Sec. 4.2)
  for *5 iterations* **do**
    Do the assignment step Eq. 1
    Do the update step (Sec 4.3)

points in the scene rather than pixels shifts in the image plane. Fortunately, in the case of super-rays, the scene flow estimation needs to be estimated only for centroids and not for all rays of the light field. Formally, for two consecutive light field frames f and $f+1$, one could estimate the scene flow $(\delta_c^{\mathbf{x}}, \delta_c^d)$ at the centroid ray $r_c^f = (\mathbf{s}_c, \mathbf{x}_c)$ as

$$(\delta_c^{\mathbf{x}}, \delta_c^d) = \underset{\delta^{\mathbf{x}}, \delta^d}{\arg\min} \sum_{\mathbf{s}'} \Delta^B_{RGB}(r_c^{f+1}, r_c'^{f+1}) \qquad (3)$$

where

$$r_c^{f+1} = (\mathbf{s}_c, \mathbf{x}_c + \delta^{\mathbf{x}}),$$
$$r_c'^{f+1} = \left(\mathbf{s}', \mathcal{P}_{\mathbf{s}'}^{d_c+\delta^d}(\mathbf{x}_c + \delta^{\mathbf{x}})\right);$$

and Δ^B_{RGB} the color distance between two patches of size B centered at r_c^{f+1} and $r_c'^{f+1}$.

However, this 3-dimensional cost function being quite expensive to minimize, we have split the problem into optical flow and depth estimation, like other methods for light field scene flow estimation in the literature [4, 27]. Besides, it is not clear to which extent solving jointly the displacement for $\mathbf{x}$ and d would be beneficial.

Now, in state of the art optical flow estimation methods [3] Deep Flow [33] stands out for its performance in terms of quality and run-time. Deep flow first searches for sparse matches using Deep Match [25] between downsampled versions of the input frames, and then the matches are densified by regularizing a selected set of sparse matches. Deep Match and has many properties which are interesting for our problem. It is robust and efficient since it can be implemented on GPU[4] and the matches are searched in a limited local window. Thus, we solve the sparse flow estimation using deep matches. In contrast to deep flow, we do not seek to obtain a dense and precise optical flow, but rather a robust and fast sparse flow for each centroid.

[4] http://lear.inrialpes.fr/src/deepmatching/

We compute the set of deep matches [25] from two downsampled frames f and $f+1$. Then, the estimated flow $\delta^{\mathbf{x}}_m = \mathbf{x}^{f+1}_m - \mathbf{x}^f_m$, using the deep matches in the full resolution coordinate system, is used to compute the flow of each centroid $\delta^{\mathbf{x}}_c$ using a simple and fast bilinear interpolation. Precisely, $\delta^{\mathbf{x}}_c$ is the distance-weighted average flow $\delta^{\mathbf{x}}_m$ of its 4 nearest matches.

Using the notation above, the depth is updated using the same strategy as in [14]:

$$\delta^d_c = \arg\min_{\delta \in D} \left\{ \min_{o \in \Omega} \sum_{\mathbf{s}'} o(\mathbf{s}')\ \Delta^B_{RGB}(r^{f+1}_c, r'^{f+1}_c) \right\}, \tag{4}$$

where D is the small range of potential depth movements and Ω is a family of spatio-angular patches.

4.2. Centroid creation and deletion

Because of object movements in the scene, the super-ray topology can change in time. For instance, parts of the super-rays can be occluded or disoccluded, or completely appear or disappear due to objects entering or leaving the scene. For this reason, creating and deleting super-rays might be necessary. While the superpixel size or color consistency has been used to determine the creation or deletion in other research works, we propose to leverage the depth information associated to the super-ray to detect occlusions and disocclusions.

In particular, a new super-ray is candidate to be created at the midpoint of two super-rays when their centroid distance exceeds a given threshold $S*\tau$. Conversely, a super-ray will be a candidate to be deleted if two super-rays are too close from each other, i.e. their centroid distance is lower than a threshold S/τ. In particular, the occluded super-ray (with the smallest disparity or biggest depth) is the candidate for deletion. For the sake of efficiency, and to avoid duplicates, we search the candidate centroids to be deleted or created in a 4-nearest neighborhood, computed as illustrated in Fig. 2. Specifically, the approximate neighborhood of a centroid c is defined as $\mathcal{N}(c) = \{c^{left}, c^{right}, c^{up}, c^{down}\}$ where

$$c^{left} = \arg\min_{\hat{c}} \Big\{ |y_{\hat{c}} - y_c| \text{ s.t. } x_{\hat{c}} < x_c,\ |y_{\hat{c}} - y_c| < S \Big\}, \tag{5}$$

and similarly for the other neighbor centroids.

Now, in order to maintain the number of super-rays constant, we create the same number of super-rays we delete. If the number of candidates for deletion is smaller (resp. bigger) than the number of candidates for creation, only the centroids with the biggest (resp. smallest) centroid distance are created (resp. deleted).

Finally, because objects can move inside or outside of the reference view, the super-rays near the image borders are treated as follows. New super-rays are created in the reference view between the image borders and the closest centroids. For instance, if a centroid c does not have a neighbor c^{right}, a new centroid will be $(\frac{x_c+M}{2}, y_c)$, M being the reference view width. Super-rays that leaves the reference view image plane are automatically deleted.

Note that the centroid neighborhood can be used for further processing, as it is a convenient way of representing the super-rays structure.

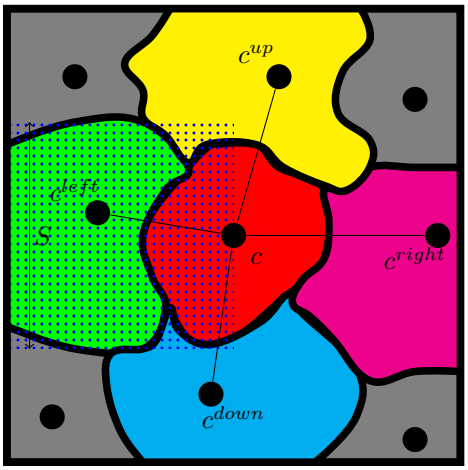

Figure 2. Super-ray neighborhood. Each super-ray is represented by a solid color and its centroid by a black dot. The search area for the left neighbor c^{left} of the red super-ray c is represented by the blue dots, and the final neighborhood connections of c by the black lines.

4.3. New frame over-segmentation

In the new frame, after defining the set of centroids in the reference view, all the rays of the light field are assigned to a centroid. Similarly to super-rays, the assignment is done using Eq. 1 in an iterative process with color and position centroid updates. While the centroid color is updated with the same color average strategy as super-rays, the centroid position update changes for dynamic super-rays. So Eq. 2 becomes

$$\mathbf{x}^{f+1}_c = \left(\frac{p}{|SR^{f+1}_c|} \sum_{r \in SR^{f+1}_c} \mathcal{P}^{d_c}_{\mathbf{s}_c}(\mathbf{x}^f_r) \right) + (1-p)(\mathbf{x}^f_c + \delta^{\mathbf{x}}_c) \tag{6}$$

where p is a parameter controlling how much the super-rays are allowed to move from their theoretical position. When $p = 1$, this step corresponds to the same SLIC iteration as in Eq. 2, and when $p = 0$, the super-ray centroids are not updated at all, providing the best temporal consistency. Newly created centroids (as described in Sec. 4.2) are updated using $p = 1$, allowing them to adapt to scene changes.

In [24], 5 SLIC iterations are run, where the centroids are allowed to move freely. As a consequence, superpixels of static objects tend to move since they are affected by the creation, deletion and movements of nearby superpixels. On the contrary, our dynamic super-rays movement is congruous with the objects movement in the scene, providing a more consistent temporal over-segmentation.

5. Experiments

Currently, two datasets for video light fields captured with camera arrays are available. The Fraunhofer dataset [8], with sequences of 3×3, 3×5 and 4×4 views, a camera resolution of 1920×1080 pixels and sequences between 150 and 400 frames. On the other hand, the Technicolor dataset [26] with sequences of 4×4 pseudo-rectified views, a camera resolution of 2048×1088 pixels and sequences between 300 and 390 frames. The cameras baseline is quite important for the second dataset.

As hyper-parameters, fixed for all the datasets, we use a down-sampling factor of 2 and a flow window of 30 pixels for the computation of the deep matches. The δ^d search range is limited to $1/10$ of the depth search range, given for each dataset. The depth block size is fixed to 11×11 pixels. The compactness parameter λ is fixed to 0.5 and τ and p are fixed to 1.9 and 0.4 respectively. We generated 1500 super-rays for the Technicolor dataset and 2000 for the Fraunhofer dataset. These number of super-rays offer a good trade-off between segmentation accuracy and super-ray tolerance to occlusions (as discussed in [14]) for each of the datasets. Our dynamic super-rays are computed in the whole sequences without fragmenting them.

Fig. 3 shows the output of our algorithm for a small area of the dataset *Birthday* [26]. For the sake of visualization, results are only shown for two views, the reference view $\mathbf{s}_c = (1, 1)$ and another view $\mathbf{s} = (1, 0)$, and three non-consecutive frames $f = 95, 100, 105$. For each view, we show the input image with the optical flow only on the reference view (top left), the color-coded assignment (top right), the super-ray average color (bottom left) and finally the super-ray contours (bottom right).

Please note that it is hard to evaluate our over-segmentation results on paper due to the reduced number of frames or views we can illustrate compared with the full light-field videos. We strongly encourage the reader to visualize all our results on our web-page[5]. The resulting videos show concatenated views with usual visualization methods, namely, the super-ray average color, the color-coded super-ray labels, and the super-ray contours (as in Fig. 3). We also visualize the value of the flow for each centroid, as well as the coarse depth of each super-ray, by assigning the super-ray centroid depth to all the rays having the same label. Finally, we show the centroids neighborhood structure in which the deleted or created centroids are differently colored.

We compare our method with the algorithm in [24] which is the state of the art for temporal consistent superpixels on videos. Note that in this experiment we focus on the temporal aspect since it has already been shown [14] that computing superpixels on each of the views separately does not guarantee angular consistency. So, here we show our over-segmentation results for the reference view only. Fig. 4 shows this comparison on five frames, $f = 260, 262, 263, 264, 265$. In particular, on the top row, we show the neighborhood structure described in Sec. 4.2. Each centroid appears as a blue dot, and horizontal and vertical neighborhoods are illustrated with cyan and magenta edges respectively. Centroids of deleted super-rays are represented in red, while new super-rays are represented in yellow. The second and third rows correspond to our results and the results of [24] respectively.

We observe that the update step in [24] allows the superpixels on the static background to move freely. On the contrary, our super-rays are not moving so the scene movement is consistent with the super-rays movement. We believe this is a major benefit if dynamic super-rays are to be used in further editing tasks. We invite the reader to view the video in the supplementary material where temporal consistency is more visible[5].

Besides the qualitative comparison with [24] we have also observed considerable differences in terms of computational complexity. Depending on the datasets, the algorithm in [24] takes several hours and up to one day (using the original implementation) to run for all the frames of a single view video. In our case, the biggest advantage is the GPU friendliness. Indeed, the SLIC-based iterations, the deep flow computation and the super-ray creation and deletion, are highly parallelizable. On the same machine (equipped with a *Nvidia GTX 1080* GPU hosted by an *Intel Xeon E5-2630* CPU) our current *Python/PyOpencl* implementation gives an average running time for each iteration of 0.157s and 0.059 to 0.083s (depending on the input size), respectively on [26] and [8]. Further improvements are to be expected by a more optimized implementation.

Dynamic super-rays with the neighborhood structure presented in Sec. 4.3 offer a useful representation of the scene captured by the light field videos. Temporal super-rays can be seen as a powerful tool for efficient light-field video editing in which the edits in one reference view of the light-field can be easily propagated to other frames and

[5] Since the supplementary material space is too limited to fit all our results, it only contains the two full sequences shown in this paper. To respect the double blind policy, all our results are hosted on an anonymous Youtube channel https://www.youtube.com/channel/UCHFkXPUSiV3UFxlABRmQkNA/videos and can be downloaded https://drive.google.com/open?id=1L-txOlsgmP6AXkKsqbdwVK5GBZMl2PYD. All our results will be available on our web-site upon publication.

Figure 3. Video super-rays for 3 frames and 2 views of the dataset *Birthday* [26].

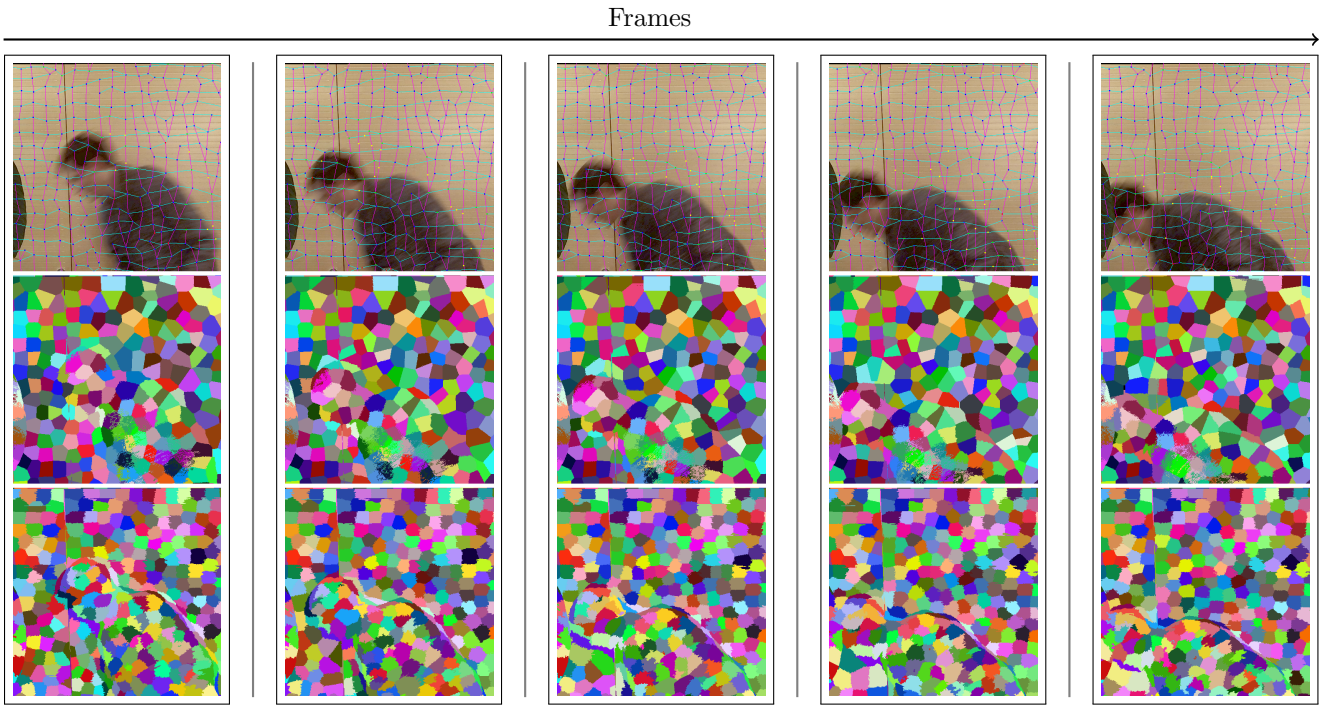

Figure 4. Over-segmentation comparison with [24] on the reference view over 5 frames. Our dynamic super-rays (second row) are consistent with the scene movement while the superpixels in [24] (third row) move in static regions.

views. For example, in [14] it is presented how to use super-rays to generate intermediate views to correct the angular aliasing caused by the poor angular sampling of sparse light fields. Similarly, dynamic super-rays can be used for temporal image interpolation without *flickering* caused by an inconsistent interpolation. Other examples of temporal super-rays applications include light-field video compression (e.g. adapting the approach in [9]), or light field color transfer (e.g. using the algorithm in [11]).

Limitations: We have observed that our approach has some limitations, in particular, when the depth or the flow estimation becomes erroneous, the super-ray consistency is not guaranteed from one view to another. Such failure case is visible in the dataset *Newspeaker* [8], where a very uniform green background challenges both the depth and flow estimations. When the depth is inconsistent, the centroids are wrongly projected, leading to large areas with no nearby centroid for the rays to be assigned to.

The other failure case involves small moving objects, because of our sparse flow computation strategy, the optical flow for small object can be wrongly evaluated to the flow value of its surrounding. This is visible on the dataset *Train* [26], where centroids struggle to follow the train wagons.

In conclusion, even if depth and flow estimation are mature research topics we have observed that challenging datasets may still produce inaccurate estimates. In particular, the images in the two datasets suffer heavily from motion blur, noise and over and under exposition. Furthermore, the dataset in [8] has some large texture-less areas.

However, loosing consistency in flat areas is not critical. Indeed, if the zone to edit is totally uniform, the editing become trivial (e.g. using a simple color threshold), dismissing the needs for super-rays in the first place.

6. Conclusion

We presented an approach to generate angularly and temporally consistent light field video over-segmentations. Our algorithm design allows a GPU implementation, allowing computational performances that are required to cope with the high volume of data. To the best of our knowledge, this is the first approach to deal with the problem of video light field editing.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012. 1, 2, 3

[2] R. Achanta and S. Süsstrunk. Superpixels and Polygons using Simple Non-Iterative Clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017)*, 2017. 2

[3] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for opti-

cal flow. *International Journal of Computer Vision*, 92(1):1–31, 2011. 4

[4] T. Basha, S. Avidan, A. Hornung, and W. Matusik. Structure and motion from scene registration. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1426–1433, June 2012. 4

[5] J. Berent and P. L. Dragotti. Unsupervised extraction of coherent regions for image based rendering. In *BMVC*, pages 1–10, 2007. 2

[6] N. D. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla. Automatic object segmentation from calibrated images. In *Visual Media Production (CVMP), 2011 Conference for*, pages 126–137. IEEE, 2011. 2

[7] J. Chang, D. Wei, and J. W. Fisher, III. A video representation using temporal superpixels. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013. 2, 3

[8] L. Dabala, M. Ziegler, P. Didyk, F. Zilly, J. Keinert, K. Myszkowski, H.-P. Seidel, P. Rokita, and T. Ritschel. Efficient Multi-image Correspondences for On-line Light Field Video Processing. *Computer Graphics Forum*, 2016. 1, 6, 8

[9] G. Fracastoro, F. Verdoja, M. Grangetto, and E. Magli. Superpixel-driven graph transform for image compression. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 2631–2635. IEEE, 2015. 8

[10] F. Galasso, R. Cipolla, and B. Schiele. Video segmentation with superpixels. In *Proceedings of the 11th Asian Conference on Computer Vision - Volume Part I*, ACCV'12, pages 760–774, Berlin, Heidelberg, 2013. Springer-Verlag. 2

[11] R. Giraud, V.-T. Ta, and N. Papadakis. Superpixel-based color transfer. In *IEEE International Conference on Image Processing (ICIP)*, 2017. 8

[12] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 43–54. ACM, 1996. 1, 3

[13] M. Hog, N. Sabater, and C. Guillemot. Light Field Segmentation Using a Ray-Based Graph Structure. In *European Conference on Computer Vision - ECCV*, page 16, Amsterdam, Netherlands, Oct. 2016. 2

[14] M. Hog, N. Sabater, and C. Guillemot. Super-rays for efficient light field processing. *IEEE Journal of Selected Topics in Signal Processing*, PP(99):1–1, 2017. 1, 3, 4, 5, 6, 8

[15] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi. Turbopixels: Fast superpixels using geometric flows. *IEEE transactions on pattern analysis and machine intelligence*, 31(12):2290–2297, 2009. 2

[16] Z. Li and J. Chen. Superpixel segmentation using linear spectral clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1356–1363, 2015. 2

[17] A. Lumsdaine and T. Georgiev. The focused plenoptic camera. In *ICCP*, pages 1–8. IEEE, 2009. 1

[18] F. Meyer and P. Maragos. Multiscale morphological segmentations based on watershed, flooding, and eikonal pde. In *International Conference on Scale-Space Theories in Computer Vision*, pages 351–362. Springer, 1999. 2

[19] B. Mičušík and J. Košecká. Multi-view superpixel stereo in urban environments. *International journal of computer vision*, 89(1):106–119, 2010. 2

[20] H. Mihara, T. Funatomi, K. Tanaka, H. Kubo, Y. Mukaigawa, and H. Nagahara. 4d light field segmentation with spatial and angular consistencies. In *Computational Photography (ICCP), 2016 IEEE International Conference on*, pages 1–8. IEEE, 2016. 2

[21] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan. Light field photography with a hand-held plenoptic camera. *Computer Science Technical Report*, 2(11):1–11, 2005. 1

[22] C. Y. Ren, V. A. Prisacariu, and I. D. Reid. gSLICr: SLIC superpixels at over 250Hz. *ArXiv e-prints*, Sept. 2015. 2

[23] X. Ren and J. Malik. Learning a classification model for segmentation. In *International Conference on Computer Vision*, page 10. IEEE, 2003. 2

[24] M. Reso, J. Jachalsky, B. Rosenhahn, and J. Ostermann. Temporally consistent superpixels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 385–392, 2013. 2, 3, 6, 8

[25] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Deepmatching: Hierarchical deformable dense matching. *International Journal of Computer Vision*, 120(3):300–323, 2016. 4, 5

[26] N. Sabater, G. Boisson, B. Vandame, P. Kerbiriou, F. Babon, M. Hog, R. Gendrot, T. Langlois, O. Bureller, A. Schubert, and V. Allié. Dataset and pipeline for multi-view light-field video. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1743–1753, July 2017. 1, 6, 7, 8

[27] P. P. Srinivasan, M. W. Tao, R. Ng, and R. Ramamoorthi. Oriented light-field windows for scene flow. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3496–3504, 2015. 4

[28] 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*, pages 13–26. Springer, 2012. 2

[29] M. Van den Bergh, G. Roig, X. Boix, S. Manen, and L. Van Gool. Online video seeds for temporal window objectness. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 377–384, 2013. 2

[30] A. Vedaldi and S. Soatto. Quick shift and kernel methods for mode seeking. In *European Conference on Computer Vision*, pages 705–718. Springer, 2008. 2

[31] P. Wang, G. Zeng, R. Gan, J. Wang, and H. Zha. Structure-sensitive superpixels via geodesic distance. *International journal of computer vision*, 103(1):1–21, 2013. 2

[32] S. Wanner, C. Straehle, and B. Goldluecke. Globally consistent multi-label assignment on the ray space of 4d light fields. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1011–1018, June 2013. 2

[33] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. DeepFlow: Large displacement optical flow with deep matching. In *IEEE Intenational Conference on Computer Vision (ICCV)*, Sydney, Australia, Dec. 2013. 4

[34] B. Wilburn, N. Joshi, V. Vaish, M. Levoy, and M. Horowitz. High-speed videography using a dense camera array. In *CVPR*, volume 2, pages II–294. IEEE, 2004. 1

[35] H. Zhu, Q. Zhang, and Q. Wang. 4d light field superpixel and segmentation. In *IEEE CVPR*, 2017. 3