Optimizing Superpixel clustering for real-time egocentric-vision applications

Pietro Morerio, Gabriel Claudiu Georgiu, Lucio Marcenaro and Carlo Regazzoni †

***Abstract*—In this work, we propose a strategy for optimizing a superpixel algorithm for video signals, in order to get closer to real time performances which are on the one hand needed for egocentric vision applications and on the other must be bearable by wearable technologies. Instead of applying the algorithm frame by frame, we propose a technique inspired to Bayesian filtering and to video coding which allows to re-initialize superpixels using the information from the previous frame. This results in faster convergence and demonstrates how performances improve with respect to the standard application of the algorithm from scratch at each frame.**

***Index Terms*—Superpixel, Video analysis, Egocentric vision, First-person vision, Optimization, Bayesian Filtering**

I. Introduction

THE concept of *Superpixels* has matured during the last ten years and by now has been widely explored within the computer vision community. Superpixel algorithms are meant to group *similar* pixels into meaningful regions, or clusters, in order to create a higher-level structure in an image. They capture similarities mainly by jointly considering colour and spatial proximity and thus try to provide a semantic clustering of an image.

Although the majority of image processing algorithms operate at pixel level, processing higher-level representations (see Figure 1) can turn out to be more efficient. For example, one can reduce the hundreds of thousands of pixels to hundreds or thousands of superpixels while still maintaining very accurate boundaries of objects or other key features in an image, such as colour statistics. In fact, it is often even more convenient to get rid of noisy (and often redundant) pixel-level information.

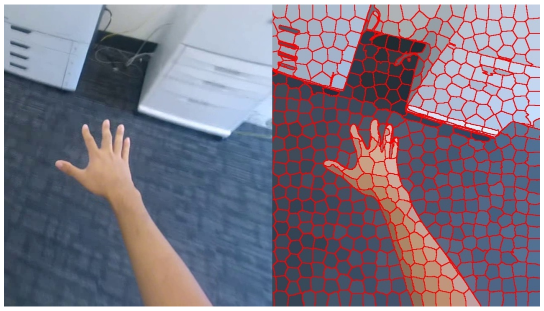


Fig. 1. Superpixel methods provide a higher level representation of an image. Such a representation is particularly suitable for segmentation purposes and in fact often used to this end.

†*Department of Naval, Electrical, Electronic and Telecommunication Engineering* - University of Genoa, *Cognitive Information and Signal Processing for Telecommunications group* - www.isip40.it.

For these reasons, Superpixels methods have been exploited for many purposes, ranging from segmentation to feature computation and are becoming a really popular preprocessing tool in many computer vision applications. Just to mention some: foreground-background segmentation [1], object localization [2], tracking (and extended tracking) [3] [4] [5]. By providing a mid level representation of an image, Superpixel methods are also used in visual saliency detection [6], in order to extract relevant information from images. Eventually, Superpixels can be extended to Supervoxels, which are widely employed in biomedical applications [7].

The majority of superpixel methods are segmentation-oriented and in fact the line between over-segmentation and superpixel-segmentation is not neat at all. [8] proposes that superpixel segmentation is a particular oversegmentation which preserves a sufficient amount of the salient features of its underlying pixel-level representation. This is why superpixels can be exploited for different purposes other than segmentation; however, again, such a sufficient amount is extremely arbitrary, yet depending on the application. We suggest that superpixel algorithms are simply a class of methods for extracting arbitrarily higher level features from images.

There are several approaches to generating Superpixels. Each method can be considered to perform better only depending on the kind of problem it is applied to. For instance, graph-based methods such as [9] seem to provide better adherence to boundaries. On the other hand, one may want to construct a graph out of a Superpixel grid in which case enforcing Superpixels connections is an issue [10]. Also, some methods give more regular cluster's contours [11], while some older approaches construct irregular shapes with inhomogeneous sizes [12]. State-of-the-art algorithms can be categorized in 2 main groups, as either *graph-based* or *gradient ascent* methods. *Graph-based* approaches consider each pixel as a node in a network (or graph), connected to his neighbours through edges. Weights of edges are related to pixel's similarity in a given colour space and Superpixels are generated by gradually cutting such overconnected graphs, by minimizing some cost function defined over the whole image [13], or by finding minimum spanning trees [14]. Gradient-descent-based algorithms start instead from a given rough clustering of the image. Clusters are then refined iteratively until some global convergence criterion is met [15], or after a fixed number of iterations [16].

Despite the prolific literature on the topic, the application of superpixels to video analysis is still at his early stages and have been employed in first-person-vision only in [17] for hand segmentation and tracking purposes. More in details, this work exploits the by now consolidated *Simple Linear Iterative Clustering* (SLIC) algorithm proposed in [16], by simply applying the method from scratch frame by frame. This turns out to be quite slow, still not allowing real-time performances, required by egocentric-vision applications. [18] actually provides a real time implementation of SLIC, but it needs a graphic card to exploit GPU and the NVIDIA CUDA framework. Such implementation is hardly portable to a wearable device.

Other attempts to provide fast superpixel algorithms are [19] and [20]. However, while the first focuses again on single images and not on videos, the second sacrifices accuracy for the sake of performance.

To the best of our knowledge, the only effective attempt to bridge the gap between Superpixels and video is addressed by [8], where a characterization of Temporal Superpixels (TSPs) is provided. Here a complex generative model is proposed in an attempt of treating the video signal not as a simple sequence of images, but with a special stress on time. The method differs from the Supervoxels approach [21], which works well for actual volumetric data (e.g. medical imaging), in that time is not simply treated as an extra dimension. It is probably the first probabilistic model to represent superpixels. However, the method, although very effective, is far from providing real-time performances, requiring tens of seconds for performing Bayesian inference over a single frame.

In this work we suggest a novel way of performing a smart re-initialization of superpixels in consecutive frames, in order to optimize frame's elaboration time. The primary goal is here to get closer to real-time video elaboration. The proposed approaches (section II) take some inspiration from Bayesian filtering and from video coding. In particular, results (section III) are shown for SLIC [16], but they can be easily extended to other methods such as [10] and with modifications to graph-based approaches as well.

II. Proposed Method

In this section we first briefly review SLIC (section II-A), in order to better explain the proposed optimization methods (sections II-Ba nd II-C).

A. SLIC

SLIC considers a 5-dimensional feature vector for each pixel, composed of its (x, y) position, and its three *Lab* colour channel values. The algorithm is initialized with a fixed number of cluster's centres, equally spaced and arranged in a regular grid of step S (refer to Figure 2). k-means clustering is then performed by searching for similar pixel in overlapping $2S \times 2S$ regions (here is the key to speeding up with respect to standard k-means). Once each pixel has been associated to the nearest cluster, centres' positions are adjusted to be the mean feature vector of all the pixels belonging to the cluster. A residual error E between the new and the previous cluster centre locations is then computed. These steps are repeated iteratively until convergence. However, the authors claim that 10 iterations suffice for most images.

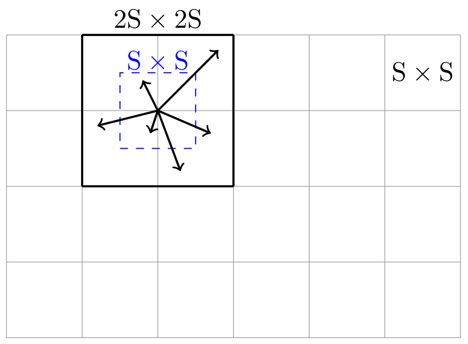


Fig. 2. Unlike standard k-means algorithms, SLIC searches a limited $2S \times 2S$ region only. The expected superpixels' size is $S \times S$, as the initialization grid cells. S is derived from the image dimensions and the number of desired superpixels

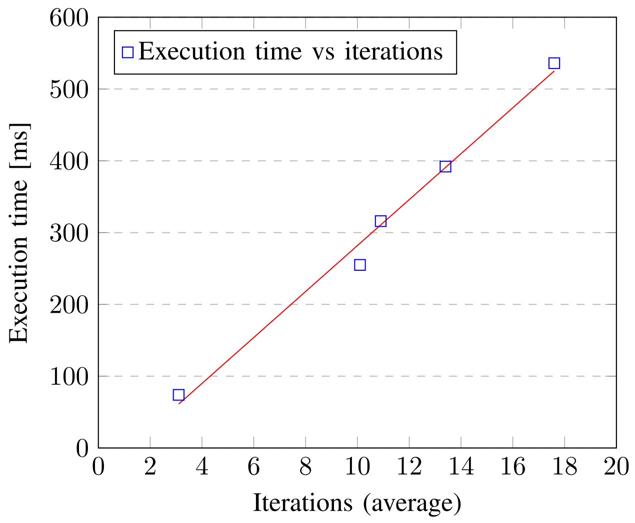


Fig. 3. Execution time [ms] against average number of iterations required for convergence: the relation is approximately linear. Data are provided in table I.

In order to measure the amount to which SLIC's performances can be improved, we follow the original algorithm, by fixing a threshold $0 \leq E_{max} \ll 1$. We then measure execution time (or, equivalently the number of iterations, which is directly proportional as shown in Figure 3) needed for convergence.

B. Bayesian approach

As we deal with videos, the most natural question we could ask ourselves is: can we exploit the information carried by a frame to process the next incoming? And, if yes, how?

A very simple answer is given by the *Bayesian filtering* framework, also known as *recursive Bayesian estimation*. This well known probabilistic approach aims at estimating an unknown probability density function recursively over time using incoming measurements and a mathematical process model. Its simplest analytical implementation, the Kalman filter (KF) [22], is so widely employed that needs no introduction. Such filter provides an efficient computational (recursive) means to estimate the state of a process, in a way that minimizes

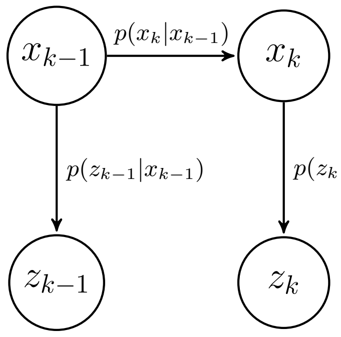


Fig. 4. Graphical model (Dynamic Bayesian Network) for a Bayes filter.

the mean of the squared error. The filter is very powerful in several aspects: it supports estimations of past, present, and even future states, and it can do so even when the precise nature of the modelled system is unknown. In particular, it consists of a *prediction* step, followed by an *update* step where the actual measure is incorporated in the estimation. Many extensions have been developed for non-linear equations and non-Gaussian noise (Extended KF, Unscented KF, Cubature Filter [23], Particle Filter [24]).

The estimation of new cluster centres can be modelled as a Markov process (Figure 4), where the hidden state at discrete time k is the set of all cluster centres $\mathbf{x}_k = [\mathbf{l}_k, \mathbf{a}_k, \mathbf{b}_k, \mathbf{x}_k, \mathbf{y}_k]$ and the measurement z_k is the whole k-th frame. Applying SLIC independently on frames of a video is equivalent to skip the *prediction* step in a KF, relying on measurements only. We here instead try to model the process entirely, as it is common understanding that even a trivial prediction can remove a lot of noise from the process.

Consider the general time-discrete process, identifying the dynamic model of the system and the measurement model.

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}, \tag{1}$$

$$z_k = Cx_k + Du_k + v_k, . \tag{2}$$

$$\text{where } p(w_k) = \mathcal{N}(0, Q) \ , \ p(v_k) = \mathcal{N}(0, V). \tag{3}$$

In a KF, A, B, C, D are matrices (both models are linear, though usually B and D are zero) and the random variables w_{k-1} and v_k are the process and measurement noise respectively (Gaussian, zero-mean).

As for the measurement model C (modelling $p(z_k|x_k)$), we here assume it is the superpixel algorithm itself which links the state $\mathbf{x}_k = [\mathbf{l}_k, \mathbf{a}_k, \mathbf{b}_k, \mathbf{x}_k, \mathbf{y}_k]$ and the measurement z_k (the whole frame). In this perspective SLIC can be associated to C^{-1}. It is not a linear measurement model, although SLIC does preserve linearity to some extent, as cluster centres x_k are averages over a certain fraction of pixels values.

On the other hand, for what concerns the dynamics ($p(k_k|x_{k-1})$), we enforce a linear model. Common practice when no knowledge on the actual dynamics of system is given (e.g a complex moving camera scenario, as in our case), is to simply allow a sufficient amount of noise over a constant state, i.e. $A = \mathbb{I}$. This means that we suppose that the cluster centre in the following frame will be somewhere in the surroundings (namely, in his previous position plus some shift extracted from $p(w_k)$). We note that this is not far from formulating SLIC as a Gaussian mixture as proposed in [8].

What we propose in this section is not a rigorous Bayesian state estimation (although we generously draw inspiration from the Kalman filter framework). We rather suggest that, instead of re-initializing the starting regular grid (Figure 2) in SLIC at each frame, a more effective choice could be to try a guess. Common practice in KFs tells us that if we do not have a clue on the precise dynamics of the problem, a still (statistically) good option is to suppose the state to be the same as in the previous time instant (plus some noise).

We are able to measure how better this new re-initialization method perform by evaluating the number of iterations needed for convergence at each frame. As shown in the next section, less iterations are needed, since, statistically, clusters centres are closer to the actual ones and SLIC needs less iterations to converge.

To conclude this section, we would like to stress again what we propose is not a Kalman filter, although it draws inspiration from it, in a not completely rigorous fashion, in order to cope with the practical issue of speeding a superpixel algorithm up to real time performances, which are needed for first-person vision applications.

C. Video coding approach

A minor issue arises however when adopting the approach proposed in the previous section. As shown in Figures 5 and 6, small rectangular patterns appear in many superpixels after a while. The reason why these artifacts arise is to be ascribed to the fact that, as shown in Figure 2, SLIC does not implement an exact k-means clustering, but searches in a $2S \times 2S$ window. Therefore, due to divergent flows in the video scene, sometimes cluster centres are pull too far apart, leaving a gap between adjacent windows, which is not searched. Such a gap has a fairly regular shape, being delimited by the borders of two or more square boxes. Gaps' pixels are not elaborated in the current frame, and maintain the labels they were assigned in the previous time instant. This issue mainly arises around object moving in contrast with the general dynamic of the scene (e.g. hands performing gestures).

To overcome this drifting issue we adopt a strategy similar to the one exploited for similar reasons in video codecs such as h264 [25]. Here I-frames (Intra frames or key frames) are employed to decode from scratch without reference to any other frame. Similarly, we insert I-frames, where SLIC is applied from scratch, in order to prevent the aforementioned drifting. In a Bayesian perspective, this is equivalent to regularly cut the Markov chain and force the model with a new prior, namely SLIC's standard initialization grid.

We have found that sending I-frames at a ratio of $1/30$ is usually enough. Increasing the dimension of the window used by SLIC would be a solution, but it should be done in an adaptive way, which would be quite expensive. In addition, the method should also deal with superpixels' births and deaths once windows' sizes grow too big. The issue of superpixels merging-splitting and birth-death mechanism is addressed in [8] and proves to be extremely time consuming.

TABLE I
PERFORMANCES

METHOD	Execution time (ms)	Number of iterations	fps
1. SLIC	537	17,6	1,9
2. NAIVE	74	3,1	13,5
3. NAIVE + INTRA	316	10,9	3,2
4. NOISE	256	10,1	3,9
5. NOISE + INTRA	393	13,4	2,5

III. RESULTS

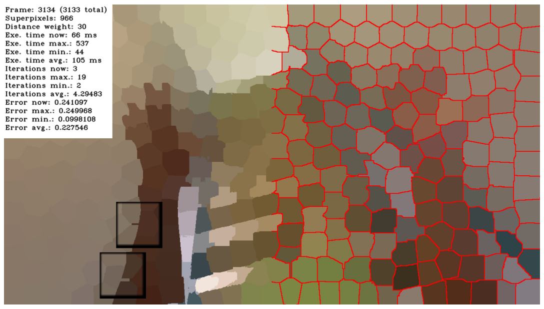
We run experiments on the egocentric video dataset provided by Kitani and colleagues [26]. The approximately six minute long video `EDSH1.avi` counts 11290 frames and records the perspective of a single user along different indoor and outdoor scenes, with really heterogeneous illumination conditions. Frames are 1280x720 pixels. SLIC parameters are set as follows $m = 30$, $N = 1000$. Residual error is $E = 0.25$ and it is of course fixed for comparison purposes.

Table I presents quantitative data averaged over the 11290 frames. Performances are referred to the tests we run on an Intel Xeon 2.66 GHz processor with 4 GB RAM. Our code is publicly available at https://github.com/ClaudiuGeorgiu/VideoSLIC. Applying SLIC from scratch at each frame results in an execution time of more than half a second, which means not even 2 fps. In fact over 17 iterations are on overage needed for having the k-means algorithm converge up to the residual error E.

By naively initializing SLIC's centre states with the values outputted by the previous frame (with no noise), produces the astonishing boosts in performances of almost 13 frames per second. This is to be ascribed to the fact that consecutive frames are extremely similar and, neglecting border effects and occlusions, superpixels require little effort to be adjusted. However, results are corrupted by some drifting effects as shown in Figure 5. The rectangular patterns or sharp angles appear after a while. This drifting phenomenon is due to the fact that SLIC does not implement an exact k-means clustering, but searches in $2S \times 2S$ windows, which can leave gaps in case of divergent flows. The absence of noise in dynamic model makes the phenomenon even more evident.

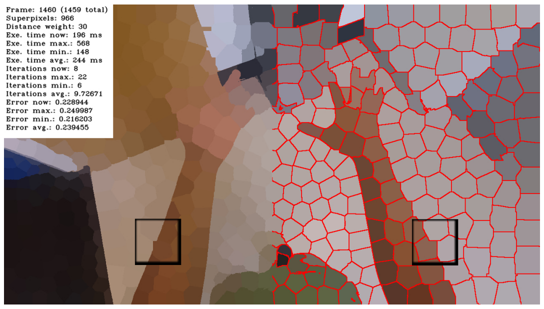
The issue of rectangular patterns becomes less predominant when injecting Gaussian noise in the dynamic model, as drifting is somehow mitigated by noise. Noise injection has of course a computational cost as random Gaussian-distributed numbers must be generated at each time step. In the test, noise was extracted from the distribution $\mathcal{N}(\mu = 0, \sigma = S/5)$. The higher the noise, the lower the drifting effect although of course more iterations are needed for convergence.

Inserting intra frames removes drifting effects both in the noisy and non-noisy case, at the price of increasing on average the number of iterations needed for convergence. Here again, the more frequently intra frames are sent, the slower the convergence. Results in Table I are obtained by sending I-frames at a frequency of $1/30$. However, the rate at which intra frames are to be inserted can depend on several factors: Superpixels' dimension, frame rate and Dynamics of the scene, to mention some. The algorithm is not self-adaptive on this parameter, which must be fixed heuristically, finding a correct trade-off between performance and other parameters.



```
Frame: 3134 (3133 total)
Superpixels: 966
Distance weight: 30
Exe. time now: 66 ms
Exe. time max.: 537
Exe. time min.: 44
Exe. time avg.: 105 ms
Iterations now: 3
Iterations max.: 19
Iterations min.: 2
Iterations avg.: 4.29483
Error now: 0.241097
Error max.: 0.249968
Error min.: 0.0998108
Error avg.: 0.227546
```

Fig. 5. Rectangular patterns appear after a while. This drifting phenomenon is due to the fact that SLIC does not implement an exact k-means clustering, but searches in a $2S \times 2S$ window, which can leave gaps in case of divergent flows. The absence of noise in dynamic model makes the phenomenon even more evident



```
Frame: 1460 (1459 total)
Superpixels: 966
Distance weight: 30
Exe. time now: 196 ms
Exe. time max.: 568
Exe. time min.: 148
Exe. time avg.: 244 ms
Iterations now: 8
Iterations max.: 22
Iterations min.: 6
Iterations avg.: 9.72671
Error now: 0.228944
Error max.: 0.249987
Error min.: 0.216203
Error avg.: 0.239455
```

Fig. 6. The issue of rectangular patterns is less predominant when injecting Gaussian noise in the dynamic model.

IV. CONCLUSION AND FUTURE WORK

In this work, we have presented a general framework for exploiting superpixels algorithms in videos. The approach does not simply consist in Bayesian filtering of cluster centres 5D positions, since the measurement model $p(z|x)$ is embedded in the superpixel algorithm itself. The practical drifting issue arising specifically in SLIC, due to the non exact nature of the k-means algorithm is addressed by periodically giving new priors $p(x_0)$ to the Bayesian filter. The inspiration comes from video coding where I-frames are periodically sent precisely to avoid drifting effects.

Substantial improvement in performance shows the bounty of the proposed approach, which was tested on Kitani's egocentric activities dataset. This represents a good starting point in optimizing computer vision techniques for egocentric video analysis. Such optimization must so far be done on the software side, as dedicated hardware (as GPUs) is yet to be available on wearable devices.

Future research directions include the applications of the presented framework to other superpixels algorithms. However, some work can still be done on SLIC (which has already proved to be an extremely powerful and versatile algorithm in many applications), for instance by measuring the drifting and making the method self-aware and self-adaptive for what concerns I-frames and noise insertion.

References

[1] F. Navarro, M. Escudero-Viñolo, and J. Bescos, "Sp-sift: enhancing sift discrimination via super-pixel-based foreground-background segregation," *Electronics Letters*, vol. 50, no. 4, pp. 272–274, February 2014.

[2] B. Fulkerson, A. Vedaldi, and S. Soatto, "Class segmentation and object localization with superpixel neighborhoods," in *Computer Vision, 2009 IEEE 12th International Conference on*, Sept 2009, pp. 670–677.

[3] A. Moore, S. Prince, J. Warrell, U. Mohammed, and G. Jones, "Superpixel lattices," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, June 2008, pp. 1–8.

[4] S. Wang, H. Lu, F. Yang, and M.-H. Yang, "Superpixel tracking," in *Proceedings of the 2011 International Conference on Computer Vision*, ser. ICCV '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 1323–1330. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2011.6126385

[5] P. Zhang, E. Milios, and J. Gu, "Graph-based automatic consistent image mosaicking," in *Robotics and Biomimetics, 2004. ROBIO 2004. IEEE International Conference on*, Aug 2004, pp. 558–563.

[6] Y. Xie, H. Lu, and M.-H. Yang, "Bayesian saliency via low and mid level cues," *Image Processing, IEEE Transactions on*, vol. 22, no. 5, pp. 1689–1698, May 2013.

[7] A. Lucchi, K. Smith, R. Achanta, G. Knott, and P. Fua, "Supervoxel-based segmentation of mitochondria in em image stacks with learned shape features," *Medical Imaging, IEEE Transactions on*, vol. 31, no. 2, pp. 474–486, Feb 2012.

[8] J. Chang, D. Wei, and J. Fisher, "A video representation using temporal superpixels," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, June 2013, pp. 2051–2058.

[9] J. Shi and J. Malik, "Normalized cuts and image segmentation," in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, Jun 1997, pp. 731–737.

[10] P. Morerio, L. Marcenaro, and C. S. Regazzoni, "A generative superpixel method," in *17th IEEE International Conference on Information Fusion (FUSION 2014)*, 2014.

[11] A. Levinshtein, A. Stere, K. Kutulakos, D. Fleet, S. Dickinson, and K. Siddiqi, "Turbopixels: Fast superpixels using geometric flows," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 12, pp. 2290–2297, Dec 2009.

[12] L. Vincent and P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 13, no. 6, pp. 583–598, Jun 1991.

[13] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 888–905, 2000.

[14] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vision*, vol. 59, no. 2, pp. 167–181, Sep. 2004. [Online]. Available: http://dx.doi.org/10.1023/B:VISI.0000022288.19776.77

[15] A. Vedaldi and S. Soatto, "Quick shift and kernel methods for mode seeking." in *ECCV (4)*, ser. Lecture Notes in Computer Science, D. A. Forsyth, P. H. S. Torr, and A. Zisserman, Eds., vol. 5305. Springer, 2008, pp. 705–718.

[16] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 11, pp. 2274–2282, 2012.

[17] G. Serra, M. Camurri, L. Baraldi, M. Benedetti, and R. Cucchiara, "Hand segmentation for gesture recognition in ego-vision," in *Proceedings of the 3rd ACM International Workshop on Interactive Multimedia on Mobile &#38; Portable Devices*, ser. IMMPD '13. New York, NY, USA: ACM, 2013, pp. 31–36. [Online]. Available: http://doi.acm.org/10.1145/2505483.2505490

[18] C. Y. Ren and I. Reid, "gslic: a real-time implementation of slic superpixel segmentation," University of Oxford, Department of Engineering Science, Tech. Rep., 2011.

[19] P. Siva and A. Wong, "Grid seams: A fast superpixel algorithm for real-time applications," in *Computer and Robot Vision (CRV), 2014 Canadian Conference on*, May 2014, pp. 127–134.

[20] F. Drucker and J. MacCormick, "Fast superpixels for video analysis," in *Motion and Video Computing, 2009. WMVC '09. Workshop on*, Dec 2009, pp. 1–8.

[21] C. Xu and J. Corso, "Evaluation of super-voxel methods for early video processing," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, June 2012, pp. 1202–1209.

[22] R. Kalman, "A new approach to linear filtering and prediction problems," *T-ASME*, vol. 1960, pp. 35–45, March 1960.

[23] I. Arasaratnam and S. Haykin, "Cubature kalman filters," *Automatic Control, IEEE Transactions on*, vol. 54, no. 6, pp. 1254–1269, 2009.

[24] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*, ser. Artech House Radar Library. Artech House, 2004.

[25] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 7, pp. 560–576, July 2003.

[26] C. Li and K. M. Kitani, "Model recommendation with virtual probes for egocentric hand detection," *Computer Vision, IEEE International Conference on*, vol. 0, pp. 2624–2631, 2013.