A Combined Optical Flow and Graph Cut Approach for Foreground Extraction in Videoconference Applications

Mihai FĂGĂDAR-COSMA[1,2], Marwen NOURI[3], Vladimir-Ioan CREȚU[2], Mihai Victor MICEA[2]

[1] Alcatel-Lucent Bell N.V.,
Copernicuslaan 50, 2018 Antwerp, Belgium,
mihai.fagadar@alcatel-lucent.com

[2] "Politehnica" University of Timisoara, Department of Computer Science,
Blvd. Vasile Pârvan 2, 300223 Timişoara, Romania,
vcretu@cs.upt.ro, mihai.micea@cs.upt.ro

[3] Alcatel-Lucent Bell Labs, Centre de Villarceaux,
Route de Villejust, 91620 Nozay, France,
marwen.nouri@alcatel-lucent.com

Abstract: Immersive videoconferences have added a new dimension to remote collaboration by bringing participants together in a common virtual space. To achieve this, the conferencing system must extract in real-time the foreground from each incoming video stream and translate it into the shared virtual space. The method presented in this paper differentiates itself in the sense that no prior training or assumptions on the video content are used during foreground extraction. A temporally coherent mask is created based on motion cues obtained from the video stream and is used to provide a set of hard constraints. Based on these constraints, a graph cut algorithm is employed to produce the pixel-accurate foreground segmentation. The obtained results are evaluated using a state-of-the-art perceptual metric to provide an objective assessment of the method accuracy and reliability. Furthermore, the presented approach makes use of parallel execution in order to achieve real-time processing capabilities.

Keywords: Foreground extraction, videoconference, optical flow, graph cut, motion segmentation, real-time video processing.

1. Introduction and Related Work

In the last decade videoconferencing has gained a lot of momentum, supported by the introduction of fixed and mobile broadband Internet and the availability of affordable and easy to use video capture hardware. Having achieved the desiderate of real-time audio, video and document sharing, the next step in videoconferencing is to deliver an immersive experience by gathering participants into a common virtual space that further enhances collaboration options [1]. At the root of this concept rests the ability of the conferencing system to accurately extract foreground information from each incoming video stream and use it to populate the virtual space which is shared with all participants. The perceived quality of foreground segmentation represents a key aspect for achieving a true immersive experience, further accentuated by the fact that foreground segmentation is in itself an ill-posed problem [2]. The implementation of an immersive videoconferencing system must therefore rely on a real-time, automatic foreground extraction algorithm, capable of handling monocular video sequences that may exhibit illumination changes, multimodal and cluttered backgrounds, camera noise and video compression artifacts.

Foreground / background segmentation has been an active research area of video sequence processing, with many algorithms and methods being developed [3-6]. The most accurate results are obtained by methods that rely on dedicated setups involving stereoscopic [7] or multiple cameras [8, 9]. While highly robust, these methods are not feasible for common videoconferencing scenarios which use off-the-shelf or integrated monocular webcams.

In the field of monocular object segmentation, the majority of encountered algorithms rely on background subtraction [4] based on an empty image of the scene provided during the initialization stage. In [10] the foreground layer is extracted by combining background subtraction with color and contrast cues. The key concept revolves around background contrast attenuation, which reduces contrast in the background layer while preserving it around object boundaries. The method proposed in [11] uses a known and stationary background image and a frontal human body detector in order to perform an initial segmentation of the person in the scene. The

result is subject to a coarse to fine segmentation process [12] that relies on a GMM model of foreground and background pixels to provide the input for an unsupervised graph cut segmentation [13, 14]. In addition, a self-adaptive initialization level sets scheme is applied in order to find the most salient edges along the person's contour. The major drawback of these otherwise accurate methods is the requirement for an initial clean background image. This cannot be satisfied in videoconference scenarios, since people are usually in the scene starting from the first frame and the number of potential backgrounds is virtually infinite.

Another approach is to replace the need for an initial background image with a learning model trained using manually labeled video sequences. Criminisi et al. [15] have adapted stereoscopic approaches to monocular video by using a probabilistic framework to fuse motion, color and contrast cues with spatio-temporal (S-T) priors generated during training phase. The accuracy of this method is similar to the one in [7], except for cases when foreground color distribution resembles the one in the background or when there is insufficient motion. Further improvements described in [16] have replaced the Hidden Markov Model with tree-based classifiers trained on ground-truth segmentations that imitate depth masks used in stereoscopic vision. The classifiers operate on motion information encoded in the form of motons (motion descriptors similar to textons, which encode texture information). This method allows a better segmentation of the foreground which is closest to the camera, being able to discard background motion. Despite their relatively high accuracy, both methods can be prohibitive due to the need to calibrate the learned priors for different types of scenes using manually labeled sequences.

A third way of addressing the foreground segmentation problem takes the form of constraints placed on the nature and position of foreground objects. Kim et al. [17, 18] propose an algorithm which targets the part of the MPEG-4 standard related to object-based video compression and handling. The algorithm performs S-T motion segmentation by combining a low-complexity spatial technique with a marker extraction and update process followed by a region growing phase. The low complexity and relative accuracy of the approach makes it suitable for use in mobile devices, but the a priori assumption that the foreground object is placed in the center of the frame limits the number of applicable scenarios. For videoconferencing, segmentation needs to take into account extra movements, other than only those related to head and torso. For example, the system must handle cases in which a person uses hand gestures and body language in order to support the presentation of a topic or to show an exhibit.

The foreground extraction method proposed in the present paper eliminates the need for initial training as well as any a priori assumptions or knowledge related to the nature of the observed scene. Starting from accurate motion cues obtained through aggregation of dense and sparse optical flow information [19], the system builds a temporally coherent mask (TCM) of foreground detected through motion. The temporal coherence of the mask in absence of motion is achieved through the use of image statistics, similar to other methods encountered in the state-of-the-art [3, 18]. To obtain the final pixel-accurate segmentation, a heuristic approach combines the TCM and sparse optic flow information in order to generate the hard foreground and background constraints for a graph-cut algorithm. The accuracy and reliability of the obtained results are evaluated using the state-of-the-art perceptual objective metric described in [20]. The proposed approach supports parallelization, enabling it to achieve real-time execution capabilities.

2. Optical Flow-based Motion Segmentation

The motion detection and segmentation part of the proposed method uses the algorithm described by the authors in [19], based on the aggregation of dense and sparse optic flow (OF) information. This approach is capable of producing accurate results while ensuring resilience to noise and compression artifacts.

The algorithm comports three steps, as follows:

1. identifying a set φ of control points on moving objects from the set Φ of sparse OF features computed between the edge images of two consecutive frames;
2. computing the concave hull $\mathcal{C}_i$ of moving objects from the set φ of control points and dense OF information R_i;
3. determining the accurate boundary $\mathcal{C}_i^S$ of

each moving region using an active contour initialized from the concave hull.

The result, shown in Figure 1, is an accurate representation of the silhouette of each moving object in the current frame. The accuracy is higher for large objects that are characterized by a higher number of control points. This matches the case of videoconferencing applications, where participants occupy a significant part of the scene.

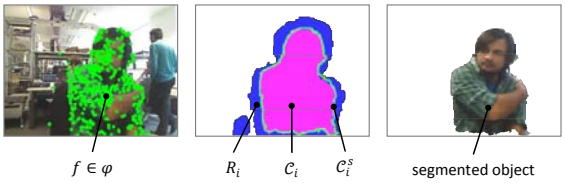

Figure 1. Concave hull and accurate boundary extraction for moving objects

3. Temporal Aggregation of Detected Movement

At each moment t relative to the beginning of the video sequence, a binary foreground mask M_t^F is generated as a reunion of the bodies of all contours $\mathcal{C}_i^S$ generated by the motion segmentation algorithm. This motion cue exposes a part of the real foreground and is valid only at the time t of its creation. In order to obtain a coherent foreground mask over a longer, ideally indefinite time frame, a method is required to integrate all motion cues $\{M_1^F, M_2^F, \dots M_t^F\}$ into a single temporally coherent mask, defined as:

$$TCM_t \coloneqq \mathcal{F}(M_i^F, i = \overline{1..t}) \quad (1)$$

In an ideal case where, at any given moment t, TCM_t is free of false positives and contains the up-to-date positions of all pixels that were exposed by at least one motion cue M_i^F, $i \leq t$, the formula can be expressed recursively:

$$TCM_t \coloneqq \mathcal{F}(TSM_{t-1}, M_t^F) \quad (2).$$

The temporal coherence approach introduced in the present article assumes that once an image region is exposed as foreground by a certain motion cue M_i^F, it will remain labeled as such in all $TCM_{t \geq i}$ until sufficient evidence is obtained to prove otherwise. This evidence comes from image statistics, in the form of a similarity measure.

3.1 Statistical Indicators for Foreground Labeling and Persistence

Image statistics are computed using a block-based approach, which offers higher robustness against noise and lighting changes [3]. The image plane of I_t is divided into equally-sized blocks of $m \times n$ pixels for a total of $\mathcal{M} \times \mathcal{N}$ blocks, and for each channel the pixel color information is modeled as a normal distribution:

$$K_{i,j,c} \sim \mathcal{N}(\mu_{i,j,c}, {\sigma^2}_{i,j,c}) \quad (3)$$

where i, j are the block coordinates and c is the color channel index. μ and σ^2 denote the mean and respectively the variance of the pixel color intensities inside the block.

Next, a similarity measure is defined between two block models. This measure indicates if the color information in a block has changed enough to trigger a change from foreground to background status. Formally, the measure is expressed as:

$$\delta(K_1, K_2) = \begin{cases} 1\,, if\ d(K_1, K_2) < \tau_d \\ 0\,, otherwise \end{cases} \quad (4)$$

where d is a distance function between two Gaussian kernels and τ_d is a threshold which depends on the chosen distance function d.

In the context of the presented algorithm, d was substituted with the Hellinger distance $H(K_1, K_2)$ [21], which produces a result in the $[0, 1]$ domain. A distance close to 0 indicates a high degree of similarity between two probability distributions, while a value of 1 means total dissimilarity.

Considering two consecutive frames I_{t-1} and I_t in the RGB color space, the similarity measure between two corresponding image blocks located at coordinates i, j can be expressed as:

$$\delta_{i,j}^t = \begin{cases} 1\,, if\ \max_{c \in \{R,G,B\}} H(K_{i,j,c}^{t-1}, K_{i,j,c}^t) < \tau_d \\ 0\,, otherwise \end{cases} \quad (5)$$

When $\delta_{i,j}^t < \tau_d$, block labeling persistence in achieved, otherwise the previous block label is invalidated. The distance threshold in case of using the Hellinger distance has been determined experimentally as $\tau_{d=H} = 0.7$.

3.2 Temporal Integration Algorithm

Each block carries a label $L_{i,j}^t$ which can take values in the $\{BG,\ FG\}$ domain. At the initial

moment $t = 0$, all blocks are assigned by default a BG label.

At each given moment $t > 0$, a block is labeled as FG if the motion cue M_t^F exposes a significant number of foreground pixels in the block. Considering $\eta_{i,j}^t$ as the ratio between the number of foreground pixels exposed by the motion cue in block i,j and the number of block pixels ($m \times n$), the labels are propagated from the previous to the current frame according to the rule:

$$L_{i,j}^t = \begin{cases} FG & ,if\ \eta_{i,j}^t \geq \tau_{FG} \\ BG, & ,if\ \eta_{i,j}^t < \tau_{FG}\ and\ \delta_{i,j}^t = 0 \\ L_{i,j}^{t-1} & ,otherwise \end{cases} \quad (6)$$

where τ_{FG} represents a threshold that controls the minimum number of foreground pixels required to mark a block with the FG label, set to a value of 0.33.

Once a block is labeled as FG, its label is persisted as long as the distribution of pixel color intensities does not exhibit significant variations between frames. A significant change in the distribution pushes the label back to BG if no motion cue is present.

The temporally coherent mask TCM_t is obtained as a binary mask from L^t by scaling it with a vertical factor of m and a horizontal factor of n and by replacing the FG and BG labels with 1 and 0 values respectively. Figure 2 shows the output of the temporal integration algorithm on a set of videoconference videos.

In Sequence a, the obtained TCM is accurate and reflects the real foreground due to the presence of significant and complementing motion cues. These motion cues draw their quality from the fact that the person's clothing presents a rich texture on which OF can be accurately estimated. Even in the case of a cluttered background and compression artifacts, the TCM maintains a lock on the FG as it can be seen in frame #78 when the person turns his head. In Sequence b the influence of the saturated background is visible in frame #299 where an added region is present on the left side of the person. The hand movement creates a temporary occlusion that causes a hole in the TCM, but overall the result covers the foreground well. In Sequence c, the smoothness, lack of color and low reflectivity of the person's clothing affects the OF motion estimation and the unstable statistical model prevents the TCM from maintaining those areas in the foreground. In contrast, the participant's head is locked onto properly and so are other regions like the second person in frame #568. These observations outline the role of the TCM as a coarse indicator of foreground areas exposed by motion and an intermediate result on the path to a pixel-accurate segmentation.

4. A Graph Cut Approach for Pixel-accurate Segmentation

The final stage of the proposed method introduces an unsupervised graph-cut segmentation process responsible of regenerating the foreground starting from the coarse result represented by the temporally coherent mask. This stage labels not only image blocks but every pixel in the image I_t, using the predefined label set $\Omega = \{BG,\ FG\}$.

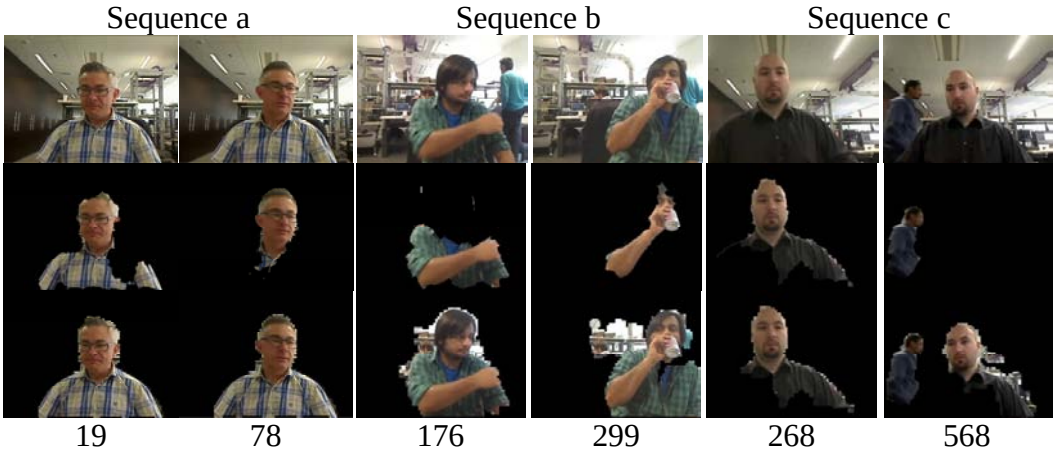

Figure 2. Temporal coherence results on different video sequences. The top row shows the captured frame, the mid row contains the motion cues and the bottom row shows $\boldsymbol{TCM_t}$ applied as a mask on the original frame.

4.1 The graph cut algorithm

Boykov and Jolly [13] introduced a segmentation technique that represents the image as a graph and provided a new min-cut/max-flow algorithm to segment (or cut) the graph. The segmentation problem is formulated as a MAP estimation of a Markov Random Field that requires the minimization of a posteriori energy. The pixels in the current frame correspond to nodes on the graph and the label space is defined as $\Omega = \{BG,\ FG\}$.

Graph edges known as n-links are established between neighbor pixels. Their weights represent the pairwise smoothing term in the energy functional and are defined by a cost function, $B_{p,q}$. In the context of binary labeling two terminal nodes are added, named Source and Sink. The terminals are linked to each pixel by weighted edges, called t-links. The relation between terminal nodes and pixels represents the unary data term in the energy functional, describing how pixels fit the foreground and background color distributions respectively. The concept is illustrated in Figure 3.

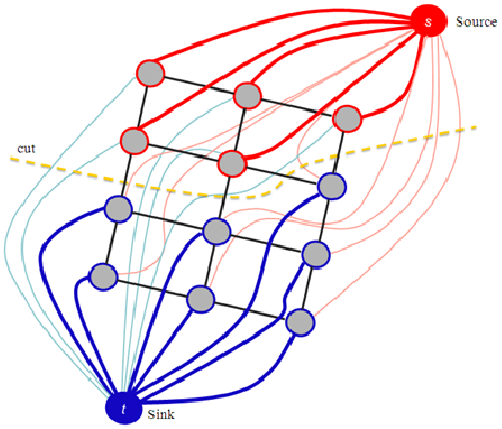

Figure 3. A graph constructed from the image data with a min-cut/max-flow graph cut.

Considering $\mathcal{P}$ the set of pixels in the frame and $\mathcal{N}$ a neighborhood system represented by a set of all pairs $\{p,\ q\}$ of neighboring elements in $\mathcal{P}$, the energy functional that corresponds to the labeling y of all pixels in the image is:

$$E(y) = \lambda \sum_{p\in\mathcal{P}} R_p(y_p) + \sum_{\{p,q\}\in\mathcal{N}} B_{p,q} \cdot \delta_{y_p \neq y_q} \quad (7)$$

where λ is a weighting factor, $y_p \in \Omega$ is the label of pixel p, R_p represents the data term, $B_{p,q}$ is the smoothness term and $\delta_{y_p \neq y_q} = \begin{cases} 1\ if\ y_p \neq y_q \\ 0\ otherwise \end{cases}$ is a Kronecker delta function that represents pixel interaction potential.

4.1.1 The Data Term t-link

In order to adhere to the real-time requirements of a videoconferencing system, an intensity histogram is used to store the pixels' probability density of color information found in the image. Although less adapted in case of color images than a GMM [23], this restriction is necessary due to performance requirements. To reduce the impact of color variety and to improve the t-link's formulation found in [24] in respect to handling color images, a regularization term which is the mean color of each label has been introduced:

$$R_p(FG) = -\frac{\ln P(I_p|FG)}{P(I_{meanFG}|FG)} \quad (8)$$

$$R_p(BG) = -\frac{\ln P(I_p|BG)}{P(I_{meanBG}|BG)} \quad (9)$$

with I_p being the color vector for a pixel p across all image channels.

4.1.2 The Smoothness Term n-link

The n-link models the cohesion between neighboring pixels, with the purpose of ensuring that labeling varies smoothly inside objects but changes at their boundaries [22]. Although the concept of pixel connectivity can be defined at different levels, the presented approach refers to the case of 4-connected neighbor pixels. The $B_{p,q}$ term exhibits large values for similar pixels and close to zero if the pixels are very different (e.g. at a region boundary). Its value is expressed as:

$$B_{p,q} = e^{-\frac{||Ip-Iq||^2}{2\sigma^2}} \cdot \frac{1}{\|p,q\|} \quad (10)$$

where $||.||$ represents the vector Euclidian norm and σ is a fixed value that estimates the video capture noise [24].

4.2 Automatic constraints generation for the graph cut algorithm

Graph cut is known to produce very accurate results in applications that rely on interactive segmentation [13, 25, 26]. In such scenarios, the user provides two sets of seeds on the image, $\mathcal{O}$ and $\mathcal{B}$ ($\mathcal{O} \cap \mathcal{B} = \emptyset$), that correspond to foreground and to background objects respectively. They act as hard constraints among all possible segmentations, linking with infinite-cost t-links any pixel $p \in \mathcal{O}$ to the

Source and any pixel $q \in \mathcal{B}$ to the Sink. The graph-cut algorithm computes the globally optimal minimum cut and separates the two terminals. Image pixels that remain connected to the Source denote the foreground, while the rest define the background.

The present paper introduces an automated approach to constraint creation, based on the information provided by the temporally coherent mask and the sparse OF.

4.2.1 Foreground Constraints

The process of generating foreground constraints, illustrated in Figure 4, starts by finding the locations in the TCM that involve a high degree of certitude in respect to their foreground label. In Section 2, the subset φ of sparse OF features was used as a confident source of control points on moving image objects. The features exposed by φ at a given moment t in time can be tracked across a larger time window as part of a bigger set $\mathcal{D}_t$, by applying the following algorithm:

- all features in the subset φ are added as part of the new set $\mathcal{D}_t$;
- all features in $\Phi - \varphi$ that are located within a $r_f = 3$ pixel radius from a feature in $\mathcal{D}_{t-1}$ are also added to $\mathcal{D}_t$ in order to account for features that belong on previously moving foreground regions, now stationary;
- the features in the newly-formed set $\mathcal{D}_t$ that are not labeled as FG in TCM_t are eliminated.

Formally, set $\mathcal{D}$ can be expressed as:

$$D_t = \varphi \cup \{f_i \in \Phi - \varphi | \exists f_j \in D_{t-1}, |f_i - f_j| < r_f\} - \{f_i \in \Phi | TCM_t(f_i) = BG\} \quad (11)$$

Features in $\mathcal{D}_t$ are first candidates for foreground hard constraints. To extend the set $\mathcal{O}$ and capture even more color information from the foreground, a Delaunay triangulation is applied on $\mathcal{D}_t$. As the reunion of all 2-simplices produced by the triangulation gives the convex hull of set $\mathcal{D}_t$, there may be facets that exceed the boundaries of TCM_t. By eliminating the 2-simplices not included in the TCM, a subset of triangulation facets is obtained. The edges of each facet in the subset are used to build a wireframe inside TCM_t. For every pixel on the wireframe, a seed of radius $r_s = 2$ pixels is added to $\mathcal{O}$. In order to avoid overextending the foreground into the background, seeds that are located too close to the TCM boundary are eliminated from set $\mathcal{O}$. This is achieved by constraining them within a mask $\widehat{TCM_t}$ obtained as the erosion of TCM_t.

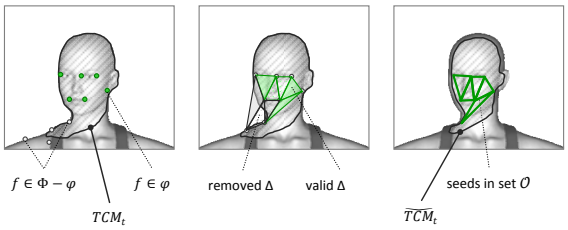

Figure 4. Process for generating hard FG constraints

4.2.2 Background Constraints

Background seeds are generated using a drape model that descends pixel by pixel from the top row of the current frame, until it reaches the proximity of TCM_t or the bottom row of the frame. The proximity mask $\widehat{TCM_t}$ is obtained by dilating the TCM over a number of iterations $i = 2 * r_s$, where r_s is the seed radius. Leaving enough space between the background and foreground seeds around the TCM border ensures that the drape does not get too close to the foreground to adversely affect the accurate boundary extraction.

At each point where the drape meets $\widehat{TCM_t}$, a new BG seed of radius r_s is added to the set $\mathcal{B}$, as shown in Figure 5-a. No seeds are added for drape pixels that reach the bottom row of the frame, in order to leave room for foreground expansion. Foreground expansion is required for cases that involve smooth regions (like the one shown in Figure 2, Sequence c).

To account for cluttered backgrounds or background elements with similar color distributions as the foreground, every $j = 3 * r_s$ columns a complete line of smaller seeds with radiuses of $r_s/2$ is added to $\mathcal{B}$, connecting the top row of the frame with the newly added BG seed (Figure 5-b).

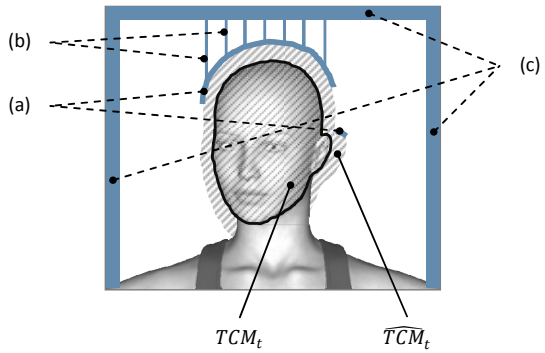

Figure 5. Automatic creation of background hard constraints for the graph-cut algorithm

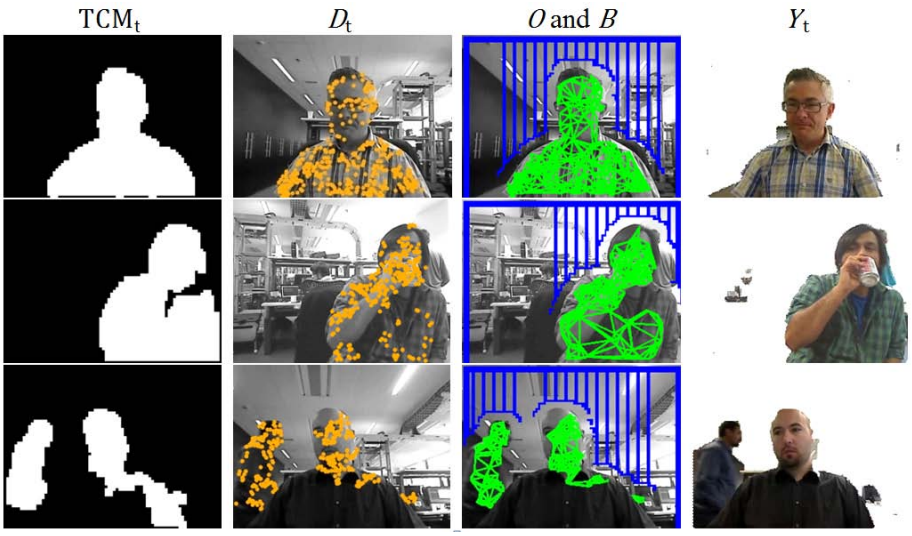

Figure 6. Foreground extraction results after graph-cut segmentation.

In order to favor the common case where foreground is mostly located in the middle of the screen, the top row, the leftmost and the rightmost columns are by default bordered with BG seeds (Figure 5-c), as long as those seeds do not reach $\widehat{TCM}_t$. The aim is to prevent an uncontrolled foreground expansion caused by the min-flow/max-cut algorithm, especially at the start of the sequence when the TCM is yet incomplete.

4.3 Final segmentation and TCM correction

After applying the graph cut algorithm, the final foreground mask $\mathcal{Y}_t$ is given by the pixels in I_t that remain connected to the Source terminal, as shown in Figure 6.

As the last step in the presented method, the TCM is corrected by removing the foreground pixels that are not labeled as FG in $\mathcal{Y}_t$. This eliminates leaking background regions, like those illustrated in Figure 2, Sequence b, so that they are not propagated into TCM_{t+1}.

5. Results and Discussion

5.1 Implementation details

The proposed foreground extraction method was implemented using the C# programming language on top of Microsoft .NET 4.0 framework. Additional image processing support was provided by the Emgu CV 2.2.1 .NET wrappers to the Intel OpenCV image processing library.

The implementation takes advantage of the multiprocessor capabilities of the host machine by exploiting the parallelization support introduced by the Parallel Extensions of .NET framework, as illustrated in Figure 7.

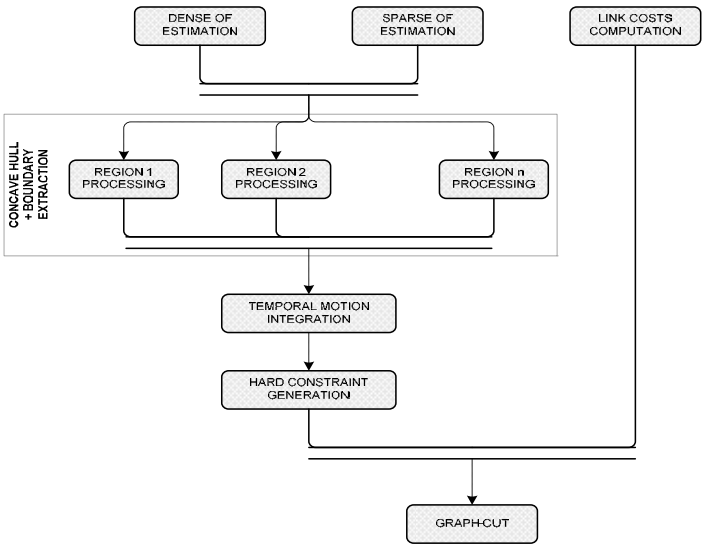

Figure 7. Parallel execution paths in the video frame processing pipeline

5.2 Execution performance

Execution performance was measured on two different test machines, using a database of test videos recorded at a resolution of 320 x 240 pixels and 30 FPS, common in most videoconferencing systems.

The first machine in the test setup was an older generation laptop equipped with a dual-core Intel Centrino P8400 CPU and 2 GB RAM, while the second, a modern desktop, had a quad-core Intel Core i5 – 2400 CPU and 4 GB of RAM.

Table 1. Algorithm execution times

CPU	**Cores**	**Load**	**FPS** (Exec. time)
Centrino P8400	2	65%	**5.8** (172.4 ms)
Core i5 – 2400	4	40%	**11.9** (84.0 ms)

The average performance results are listed in Table 1. As expected, the 4 core CPU is able to better exploit the parallelization potential achieving an average FPS of 12. Considering the managed programming language implementation and the overhead introduced by the Parallel Extensions context switching, the results prove the real-time capabilities of the proposed method on current generation hardware. Given the existing GPU implementations of optical flow estimators [28], active contours [29] and graph-cuts [30], the method has potential for significantly faster execution if implemented using video hardware acceleration.

5.3 Foreground extraction quality assessment

Gelasca and Ebrahimi [20] have proposed a set of perceptually-driven metrics based on psychophysical experiments, which enable the measurement of segmentation quality from a human perspective. For immersive conferencing, which falls into a broader group called mixed reality, their experiments have shown that the following artifacts have the most impact on the perceived FG segmentation quality (in descending order of their annoyance effect to a human observer): flickering, inside holes (HI), border holes (HB), added regions (AR) and added background (AB).

The quality of the extracted foreground was evaluated using the optimized perceptual objective metric (PST) from [20], considering the artifact weights applicable to the mixed reality scenario. An unsupervised evaluation was performed on a database of 30 videos recorded using a chroma key background. By removing the color keyed component, the ground truth segmentation is obtained. The ground truth is overlaid on a background video sequence and the result is subject to foreground extraction using the proposed method. The result is compared frame-by-frame with the ground truth in order to identify segmentation artifacts and compute their contribution to the perceptual objective metric. A total of 10 background video sequences recorded in various environments and lighting conditions were used with each chroma key video, for a total of 300 assessments. The evaluation results are summarized in Table 2.

Table 2. Perceptual quality indicators for the proposed foreground extraction method

	$\mathbf{PST_{AR}}$	$\mathbf{PST_{AB}}$	$\mathbf{PST_{HI}}$	$\mathbf{PST_{HB}}$	$\mathbf{PST_{total}}$
Mean	0.17	0.69	0.32	0.99	**19.79**
St.dev	0.16	0.31	0.25	0.02	**5.38**

Compared with the results presented in [20], the proposed method performs better in terms of overall perceived quality than the top performers [31, 32] evaluated with the same metric. Inner holes (HI), the second most annoying artifact after flickering, present a very low occurrence, as well as added regions (AR). This comes from the way hard constraints are generated for the graph-cut algorithm, as a clear separation between foreground and background is achieved. The higher value for the border holes (HB) metric is partly due to lack of sufficient motion at the beginning of the video sequences under test. The ground truth is instantly available but foreground extraction requires a sufficient number of motion cues to reach the same level of completeness. During this phase, HB artifacts are likely to occur as false negatives, sometimes in conjunction with the flickering effect.

6. Conclusions

A new method has been introduced for foreground extraction in monocular video streams, with applicability to videoconferencing systems. Without employing a priori knowledge in the form of assumptions or training relative to the scene or the object to

be segmented, the method relies on the concept of foreground exposed through motion to perform the task of foreground extraction. The approach introduces an incremental way of building a foreground image from motion cues and a novel heuristic algorithm for unsupervised graph-cut segmentation to obtain pixel-accurate object representations.

Experimental results and quality evaluations using perceptual objective metrics confirm the accuracy and the robustness of the approach, provided that motion is present in the video stream. The method achieves real-time performance by exploiting multi-core processor architectures. Research perspectives include the topic of initialization in absence of significant motion and improved statistical modeling to increase the robustness of temporal motion integration.

Acknowledgements

The authors would like to thank Mukul Dhankar and Maarten Aerts for their valuable inputs, as well as Hendrik Van Hove for recording the test videos used in the paper.

REFERENCES

1. KAUFF, P., O. SCHREER, **An Immersive 3D Video-conferencing System using Shared Virtual Team User Environments**, Proc. 4th International Conference on Collaborative Virtual Environments, ACM, 2002, pp. 105-112.
2. GRADY, L., M. P. JOLLY, A. SEITZ, **Segmentation from a Box**, Proceedings of IEEE International Conference on Computer Vision, 2011, pp. 367-374.
3. BOUWMANS, T.,F. E. BAF, B. VACHON, **Statistical Background Modeling for Foreground Detection: A Survey**, Handbook of Pattern Recognition and Computer Vision, W.S. Publishing, 2010, pp. 181-199.
4. PICCARDI, M., **Background Subtraction Techniques: a Review**, Proc. of IEEE International Conference on Systems, Man and Cybernetics, 2004, pp. 3099-3104.
5. ZAPPELLA, L.,X. LLADO, J. SALVI, **Motion Segmentation: a Review**, Proc. 11th International Conference of the Catalan Association for Artificial Intelligence, IOS Press, 2008, pp. 398-407.
6. ZHANG, D., G. LU, **Segmentation of Moving Objects in Image Sequence: A Review**, Circuits, Systems, and Signal Processing, vol. 20, 2001, pp. 143-183.
7. KOLMOGOROV, V., A. CRIMINISI, A. BLAKE, G. CROSS, C. ROTHER, **Bi-Layer Segmentation of Binocular Stereo Video**, Proceedings of 2005 IEEE Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, vol. 2, 2005, pp. 407-414.
8. DEVINCENZI, A., L. YAO, H. ISHII, R. RASKAR, **Kinected Conference: Augmenting Video Imaging with Calibrated Depth and Audio**, Proceeding of ACM Conference on Computer supported cooperative work, ACM, 2011, pp. 621-624.
9. KIM, H., R. SAKAMOTO, I. KITAHARA, T. TORIYAMA, K. KOGURE, **Robust Foreground Segmentation from Color Video Sequences Using Background Subtraction with Multiple Thresholds**, Technical report of IEICE. PRMU vol. 106, 2006, pp. 135-140.
10. SUN, J., W. ZHANG, X. TANG, H.-Y. SHUM, **Background Cut**, Proc. 9th European conference on Computer Vision, Springer-Verlag, 2006, pp. 628-641.
11. LIU, Q., H. LI, K. N. NGAN, **Automatic Body Segmentation with Graph Cut and Self-adaptive Initialization Level Set (SAILS)**, J. Vis. Comun. Image Represent. 22 (2011) 367-377.
12. LI, H., K. N. NGAN, Q. LIU, **FaceSeg: Automatic Face Segmentation for Real-time Video**, Trans. Multi. vol. 11, 2009, pp. 77-88.
13. BOYKOV, Y., M. P. JOLLY, **Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D Images**, Proceedings of the 8th IEEE International Conference on Computer Vision, 2001, pp. 105-112.
14. BOYKOV, Y., V. KOLMOGOROV, **An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision**, IEEE Trans.

Pattern Anal. Mach. Intell., vol. 26, 2004, pp. 1124-1137.

15. CRIMINISI, A., G. CROSS, A. BLAKE, V. KOLMOGOROV, **Bilayer Segmentation of Live Video**, Proc. 2006 IEEE Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, vol. 1, 2006, pp. 53-60.
16. YIN, P., A. CRIMINISI, J. WINN, I. A. ESSA, **Bilayer Segmentation of Webcam Videos Using Tree-Based Classifiers**, IEEE Trans. Pattern Anal. Mach. Intell. 33, 2011, pp. 30-42.
17. KIM, J., H.-J. LEE, T.-H. LEE, M. CHO, J.-B. LEE, **Hardware/Software Partitioned Implementation of Real-time Object-oriented Camera for Arbitrary-shaped MPEG-4 Contents**, Proc. 2006 IEEE/ACM/IFIP Workshop on Embedded Systems for Real Time Multimedia, IEEE Computer Society, 2006, pp. 7-12.
18. KIM, J., J. ZHU, H.-J. LEE, **Block-Level Processing of a Video Object Segmentation Algorithm for Real-Time Systems**, CORD Conference Proceedings, 2007, pp. 2066-2069.
19. FAGADAR-COSMA, M., V. I. CRETU, M. V. MICEA, **Dense and Sparse Optic Flows Aggregation for Accurate Motion Segmentation in Monocular Video Sequences**, Proc. 2012 International Conference on Image Analysis and Recognition, Springer LNCS, vol. 7324, 2012, pp. 208-215.
20. DRELIE-GELASCA, E., T. EBRAHIMI, **On Evaluating Video Object Segmentation Quality: A Perceptually Driven Objective Metric**, IEEE Journal of Selective Topics Signal Process., vol. 3, 2009, pp. 319-335.
21. NIKULIN, M. S., **Hellinger Distance**, Encyclopaedia of Mathematics, Springer, 2001.
22. BOYKOV, Y., O. VEKSLER, R. ZABIH, **Fast Approximate Energy Minimization via Graph Cuts**, IEEE Transactions Pattern Anal. Mach. Intell., vol. 23, 2001, pp. 1222-1239.
23. ROTHER, C., V. KOLMOGOROV, A. BLAKE, **"GrabCut": Interactive Foreground Extraction using Iterated Graph Cuts**, ACM Trans. Graph., vol. 23, 2004, pp. 309-314.
24. BOYKOV, Y., G. FUNKA-LEA, **Graph Cuts and Efficient N-D Image Segmentation**, Int. J. Comput. Vision, vol. 70, 2006, pp. 109-131.
25. BOYKOV, Y., G. FUNKA-LEA, **Optimal Object Extraction via Constrained Graph-cuts**, Int. J. Comput. Vision, 2004.
26. LI, Y., J. SUN, C.-K. TANG, H.-Y. SHUM, **Lazy Snapping**, ACM Trans. Graph.,vol. 23, 2004, pp. 303-308.
27. KUMAR, P., K. SENGUPTA, S. RANGANATH, **Real Time Detection and Recognition of Human Profiles Using Inexpensive Desktop Cameras**, Pattern Recognition, International Conference on, 2000, pp. 1096-1099.
28. SUNDARAM, N., T. BROX, K. KEUTZER, **Dense Point Trajectories by GPU-accelerated Large Displacement Optical Flow**, Proc. 11th European conference on Computer vision: Part I, Springer-Verlag, 2010, pp. 438-451.
29. HE, Z., F. KUESTER, **GPU-based Active Contour Segmentation using Gradient Vector Flow**, Proceedings of the 2nd International Conference on Advances in Visual Computing, Springer-Verlag, 2006, pp. 191-201.
30. VINEET, V., P.J. NARAYANAN, **CUDA Cuts: Fast Graph Cuts on the GPU**, 2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2008, pp. 1-8.
31. HORPRASERT, T., D. HARWOOD, L. DAVIS, **A Statistical Approach for Real-time Robust Background Subtraction and Shadow Detection**, ICCV Frame-Rate WS, 1999, pp. 1–19.
32. JABRI, S., Z. DURIC, H. WECHSLER, A. ROSENFELD, **Detection and Location of People in Video Images using Adaptive Fusion of Color and Edge Information**, International Conference on Pattern Recognition, 2000, pp. 627–630.