Object Segmentation and Tracking Using Video Locales

James Au
UBVideo Inc.
Vancouver, BC, Canada
james@ubvideo.com

Ze-Nian Li
School of Computing Science
Simon Fraser University
Vancouver, BC, Canada
li@cs.sfu.ca

Mark S. Drew
School of Computing Science
Simon Fraser University
Vancouver, BC, Canada
mark@cs.sfu.ca

ABSTRACT

In this paper, we present a new technique based on feature localization for segmenting and tracking objects in videos. A *video locale* is a sequence of image feature locales that share similar features (color, texture, shape, and motion) in the spatio-temporal domain of videos. Image feature locales are grown from tiles (blocks of pixels) and can be non-disjoint and non-connected. To exploit the temporal redundancy in digital videos, two algorithms (intra-frame and inter-frame) are used to grow locales efficiently. Multiple motion tracking is achieved by tracking and performing tile-based dominant motion estimation for each locale separately.

1. INTRODUCTION

In many applications, the ability to automatically locate and track objects in videos is very important. The most intuitive way of accomplishing this task is to first generate temporally-tracked homogeneous regions and then apply further processing (automatic or human-aided) to identify the semantic objects.

In [8], we proposed *Feature Localization* as an alternative to traditional *image segmentation* in respect to object-based image retrieval. *Locales* are enclosures of local features that are not required to be connected, disjoint, or complete. Locales also operate at a higher level than pixels, as the basic building units used are *tiles* that correspond to blocks of pixels (e.g., 16×16 or 8×8). Localization is not merely a reduced-resolution method; pixel-based statistics are collected and used throughout the process without loss of high-resolution details. One of the advantages of localization is that it generates coarse locales that are more robust to noise and complex object surfaces. It is also much more attainable as it does not require complete pixel-level segmentation.

In this paper, we describe *video locales* that exploit the temporal redundancy in digital videos. The new method deals with *object segmentation and tracking* which are important issues for object-based video coding. Initially, we use a pyramidal probabilistic-unforced-linking algorithm to extract high quality color locales (using color as the main feature) within a video frame. For subsequent frames, a modified algorithm is used to exploit the similarities between consecutive video frames. Finally, extracted locales from consecutive frames are matched and dominant motion estimation is carried out for each matching pair of locales to provide temporal tracking. Since each locale is tracked separately, the method has the potential to tackle motions of multiple objects. Our motion estimation algorithm also follows the philosophy that all calculations are done based on tiles while keeping pixel-unit precision; hence we gain the advantage of utilizing both global and local information. Experiments have shown impressive results.

2. FEATURE LOCALIZATION CONCEPT

Feature localization was introduced and described in [8].

Definition 1: A *locale* L_f is a local enclosure of feature f.

A locale L_f uses blocks of pixels called *tiles* as its building units, and has the following descriptors:

1. Envelope $\Re_f$: a set of tiles representing the locality of L_f.
2. Geometry: mass $M(L_f)$, centroid $\boldsymbol{C}(L_f)$, and pixel variances.
3. Color, texture, and shape parameters of the locale.

After a localization process the following is often true:

1. Locales are not connected: $\exists f : L_f$ is not connected.
2. Locales are non-disjoint: $\exists f \exists g : L_f \cap L_g \neq \varnothing, f \neq g$.
3. Non-completeness: $\cup_f L_f \neq I$, not all pixels are represented.

3. VIDEO LOCALES ALGORITHM

Our approach is to create video locales that are coarse but accurate approximations for both object locations and movements. A *video locale* is a sequence of image feature locales that share similar features (color, texture, shape, and motion) in the spatio-temporal domain of videos. Using color as the main feature to localize, we extract image locales from each frame and then perform motion estimation based on tiles (while making use of pixel-precision statistics). We extend our localization algorithm to take advantage of temporal redundancies in videos.

There are 3 major components in the algorithm:

1. Intra-frame Locales generation using pyramidal probabilistic-unforced-linking.
2. Inter-frame Locales generation exploiting motion-compensated frame similarity for subsequent frames.
3. Locale motion estimation using tile-based energy minimization for 2D affine motion.

The Inter-frame algorithm automatically switches to Intra-frame processing if scene and shot changes occur (see Section 3.3).

3.1 Overall Algorithm

The overall algorithm for video object segmentation and tracking is as follows:

1. Read a frame from video.
2. If this is first frame,
   a. Generate **Intra-frame Locales**
3. Else
   a. Generate **Inter-frame Locales**
   b. **Match locales** from current frame to reference frame
   c. Perform **motion estimation** for each locale
4. Repeat 1 – 3 until end of video.

3.2 Intra-Frame Locale Generation

The intra-frame algorithm uses information within the video frame only. There are two major steps. Tiles are first generated

from the image and then they are linked into locales in a pyramidal scheme.

3.2.1 Tile Generation

A relatively simple histogram analysis is performed to estimate the different dominant colors in a block of pixels. For each tile, an RGB histogram with bin widths $32 \times 32 \times 32$ is constructed and the 5 most frequent color bins become the dominant colors for the tile. After the dominant colors are determined, each pixel in the tile is assigned to the closest dominant color. Each dominant color becomes a tile-feature and the average RGB is used as the final color. Geometrical statistics mentioned in Section 2 are calculated for each tile-feature. If there are more than one tile-feature in a tile, we say that the tile-features are overlapped.

Integer RGB values are used in this step for computational simplicity. Although RGB is not as perceptually accurate as other measures, we find that it gives good enough estimates. In subsequent processing, chromaticity and luminance are used instead.

Pseudo-code for the entire tile generation process and transitional/noise pixel determination is outlined in Procedure 1 and 2 respectively.

Procedure 1: Tile Creation

1. Calculate Dominant Colors:
    a. Create an RGB histogram ($32 \times 32 \times 32$ RGB bins) for the non-transitional and non-noise pixels.
    b. Merge color bins using intensity and chromaticity.
    c. 5 most frequent color bins are the *Dominant Colors*.
2. Assign each pixel (incl. transitional & noise) to closest dominant color.
3. A Tile-Feature (Locale) is created for each dominant color.

Procedure 2: Noise/Transitional Pixel Determination

1. Transitional: intensity is on a steep slope of any 2 of 8 neighbour pixels.
2. Noise: < than 2 neighbours with similar intensity.

3.2.2 Locale Growing Pyramid

In previous papers [8] we employed a bottom-up pyramid linking method for merging the tiles into locales; however, this is plagued by one of the most confounding problems for bottom-up image segmentation methods: forced-decisions with local or incomplete information. Most decisions have to be made at the lowest level, which is characterized by local and noisy data, and wrong decisions propagate through the pyramid to cause inaccurate results. In [5], T. H. Hong and A. Rosenfeld introduced an unforced-linking algorithm in which classification decisions are deferred until more information is available. Instead of forcing a pixel in the lower level to link with one parent in the upper level, the pixel is linked to multiple parents with probabilistic weights derived from their geometric and intensity similarities. Multiple iterations update the weights until they finalize. Pixels are then assigned to their most strongly linked parent (region). The advantage of this method is that errors made at low levels can be recovered after a few iterations.

This is the strategy we employ where overlapping tile-features and locales are used in the place of pixels. Parent locale statistics are updated using probabilistic weighting for two purposes: (1) reduce impact of outliers and noise, and (2) propagate weights to next level. When a child locale is merged into the parent locale wrongly or as an outlier, the link probability is low so it would not contribute as much to the parent locale as others. Note that each child locale can be compared to more than 4 parents because parent locales may overlap (each node is a list of locales); hence, as we move up in pyramid levels, tile-details (which are measured with pixel precision) are not lost while global information is gained. This is the main reason why this algorithm is not merely a multi-resolution analysis.

3.3 Inter-Frame Locale Generation

Discrepancies between two consecutive frames are mostly caused by four things: noise, object and camera movements, introduction of new objects, and special visual effects including cut, wipe and dissolve transitions. Since locales are fairly robust to small noise, we do not worry about noise. Object and camera movements account for most of the changes in videos. Our goal is to model and predict these movements. Motion estimation enables us to generate inter-frame locales quickly and at the same time provides locale tracking. The other two causes cannot be predicted so we always scan all pixels at least once to update such changes.

In the intra-frame algorithm, most of the computation time is spent on calculating dominant colors within the tiles and on iteratively linking and re-linking nodes at the bottom-most level in the pyramid. Dominant colors calculation is costly because it is mostly a pixel-level computation. Similarly, the bottom pyramid level has as many nodes as tiles.

Our strategy is to minimize those two steps. Assume that we have 2D affine motion estimation for each locale on the reference frame (discussion for motion estimation is deferred until Section 3.4). We first transform the envelopes of the locales onto the current frame according to their motion estimation. This prediction carries with it two major pieces of information for each tile in the current frame: predicted dominant colors and predicted tile-feature ownership. Each locale appends its color to the dominant color list of the tiles within its envelope, and all tile-features created for that color are assumed to link to that locale; hence, dominant color calculation is skipped, and the unforced-linking step is also skipped. If the predicted motion is 100% accurate, this will produce optimal results with almost no work.

After the dominant colors are predicted, we examine each pixel in each tile, update tile-features' statistics, and extract any unpredicted tile-features (new colours). These new tile-features are then linked into locales in a partial pyramid that involves only the new tile-features. The new locales are merged or appended to the predicted locales. Scene changes or large occlusions will generate many unpredicted tile-features, and full pyramid linking (as in Intra-processing) is performed in such cases.

Procedure 3 shows pseudo-code for the algorithm used for generating inter-frame locales. Inter-frame quality is almost identical to intra-frame but the average inter-frame computation time is only 25% of that of the intra-frame. For frames that have little motion or those that contain very predictable movements, computation times are as low as 10% of the intra-frame time.

Procedure 3: Inter-frame Locales Algorithm

1. Predict a locale for each reference locale in previous frame.
2. Update Tiles
    a. For each reference locale that is big enough:
        i. Predict envelope from motion vector in last frame.
        ii. Append its color to all intersecting tiles.

b. For each tile:
   i. Assign each pixel to closest predicted dominant color and keep top 5.
   ii. Each tile-feature is predicted to belong to the color's corresponding locale.
   iii. If more than 20% pixels have unpredicted colors, recalculate tiles as in Intra and recover ownership:
      1. For each tile-feature, match it to closest dominant color list and link it to corresponding locale.
      2. Count # tiles that contain unclaimed tile-features.

3. Locales Linking
   a. If # unpredicted tiles < ¼ of image
      i. Pyramid-grow the unclaimed tile-features.
      ii. Merge the grown locales with predicted locales.
   b. Else, **re-grow** all tiles as in intra-frame.

3.4 Locale Motion Estimation

The simplest way to estimating locale motion is locale-centroid displacement. The vector given by the movements of a locale's centroid provides a rough estimation to the translation vector for the whole locale; however, a two-dimensional translation model may not be sufficient in the presence of scaling and rotation. Furthermore, centroids are calculated as an average and therefore they are sensitive to outliers. It is more intuitive to examine the locale envelope directly as the goal is to predict how the envelope transforms across time.

Many motion segmentation algorithms exist in the literature. There are two main approaches: dominant motion method that derives motion directly from spatiotemporal image intensity gradient information; and indirect methods that first estimate optical flow field and then determine parameters and support for multiple motion models, such as motion parameter clustering, maximum likelihood (ML) segmentation, and maximum *a posteriori* probability (MAP) segmentation [11].

In this experiment, we adopt the dominant motion approach using over-lapping tiles as the base unit to extract a good approximation for the overall motion of each locale. It is possible that a single locale may exhibit multiple motions, especially locales from non-rigid and articulate objects. The dominant motion approach provides a very intuitive and direct way of extracting the overall movement.

3.4.1 Matching Locales

Since we are tracking color locales, we make the assumption that we can model most changes in the locale envelopes by a 2D affine model. This is usually the case when the 3D scene is sufficiently distant from the camera, which is quite common in most videos [7]. In this paper, we use the 6-parameter affine model.

Suppose we call the affine transform $a_x, b_x, c_x, a_y, b_y, c_y$, then the 2-D motion $(\Delta x, \Delta y)$ of a pixel is:

$$\Delta x = a_x x + b_x y + c_x,\ \Delta y = a_y x + b_y y + c_y \qquad (3.5)$$

If we assume image motion is small, then a Taylor series expanded only to linear terms is expected to be sufficiently accurate to describe the image motion. This results in the optical flow equations [6, 9]. For an affine motion, the least-square solution over a region of interest is the solution of a 6×6 matrix times the unknown 6-vector ($a_x, b_x, c_x, a_y, b_y, c_y$), with the right-hand-side being a forcing vector.

Since locale dominant motion estimation is carried out for each locale separately, only the envelope of one object is considered and the dominant motion assumption should hold valid for most cases. It is rare that an individual object will exhibit equal competing motions. Also, the iterative process and change detection can be avoided because there is only one motion to extract. Pixel-noise is minimized when tiles are used as the observation points. Finally, in the case of low spatial gradient regions, this can be detected by a motion reliability measure defined in [7] and we can recover by falling back on locale-centroid displacement to give us an estimation of the translation vector. Procedure 4 illustrates the pseudo-code for the motion estimation process.

Procedure 4: Locale Motion Estimation

1. For each (locale, reference locale) pair:
   a. If both Locale and reference Locale have > 5 tiles:
      i. Region of interest = union of tile envelopes
      ii. Estimate Dominant Translation (d_x, d_y)
      iii. Translate Reference locale tiles by (d_x, d_y)
      iv. Estimate Affine Transform $a_x, b_x, c_x, a_y, b_y, c_y$
      v. Compose the two transforms
      vi. If near-singular condition, go to step b.
   b. Else:
      i. (c_x, c_y) = Centroid displacement.

4. EXPERIMENTAL RESULTS

We have implemented the above algorithms in our system which can read any video formats and display locales in real-time. Figure 1 compares the results of inter-frame algorithm with those of intra-frame algorithm. The difference in quality is very small even though on average the Inter-frame algorithm is 70% faster than the intra-frame algorithm (see Table 1). On a modern PC our algorithm should be able to run at 20-30 frames per second.

We evaluate the quality of our motion estimation algorithm by monitoring the number of unpredicted tiles. The tile-based dominant motion estimation algorithm (DM) is compared against other methods: (1) assume all objects are stationary (SA), (2) predict translation by locale-centroid displacement (CD). Table 2 summarizes the results. As expected, stationary assumption method performs the worst while tile-based dominant motion estimation has the best performance. When motion cannot be well predicted by the affine model (as in the wiggling fish scene), centroid displacement is as good as dominant motion estimation.

In Figure 2, a video sequence with lots of motion and occlusions is tested under the full algorithm (intra-frame & inter-frame). Color localization is very effective even though it is a difficult video. Figure 3 shows the tracking of a hockey player by his blue jersey. Notice that the localization approach overcomes the problems of disconnected regions (stripes, logos, jersey) and complex motion to provide good tracking result.

5. CONCLUSION

In this paper, we presented a new algorithm for video object segmentation and tracking in a localization framework. An unforced-linking pyramidal scheme is used to achieve good localization, and temporal redundancy is exploited by an inter-frame algorithm. Fast, effective, object-based dominant motion estimation is possible because of the introduction of video locales. Experimental results show that video objects from a variety of natural scenes can be effectively segmented and tracked.

Potential applications of this work include object segmentation and tracking in sports coverage and surveillance, video summarization, and object-based video coding.

Many future improvements are possible. Currently the motion estimation does not address the merge/split problem. If a locale is split, only part of the envelope is used in estimating motion. In future we will investigate ways of allowing multiple-locales matching to track down all the various parts of a split or merged-locale. An adaptive threshold scheme will also help.

6. REFERENCES

[1] D.H. Ballard, C.M. Brown. *Computer Vision*. Prentice Hall, 1982.

[2] J. R. Bergen, E.H. Adelson, "Hierarchical, computaionally efficient motion estimation algorithm," *J. Opt. Soc. Am.*., 4:35, 1997.

[3] P.J. Burt, C. Yen, and X.Xu, "Multi-resolution flow through motion analysis," *CVPR83*, pp. 246-252, 1983.

[4] M.S. Drew, J. Wei and Z.N. Li, "Illumination-invariant color object recognition via compressed chromaticity histograms of color-channel-normalized images", *ICCV98*, pp. 533-540, 1998.

[5] Hong, Rosenfield, "Compact Region Extraction Using Weighted Pixel Linking in a Pyramid", *IEEE PAMI*, 6(2), pp. 222-228, 1984.

[6] B.K.P. Horn, *Robot Vision*, MIT Press, 1986.

[7] M. Irani, B. Rousso, and S. Peleg, "Computing occluding and transparent motions," *Int. J. Comput. Vision* 12, pp. 5-16, 1994.

[8] Z.N. Li , O.R. Zaiane, Z. Tauber, "Illumination invariance and object model in content-based image and video retrieval", *J. Vis. Commun. Image Rep.*, Vol. 10, No. 3, pp. 219-244, 1999.

[9] B.D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision", *Im. Und. Wk.*, pp.121-130, 1981.

[10] P. Salembier and F. Marques. "Region-based representations of image and video: segmentation tools for multimedia services". *IEEE Trans. Circ. Sys. Vid. Tech.*, 9(8), pp. 1147-1169, 1999.

[11] M. Tekalp, "Chap. 4.9:Video Segmentation", *Handbook of Image and Video Processing*, Academic, pp. 383-399, 2000.

Table 1. Timing on Pentium II 400 (ms per frame)

~800 320x240 frames	Intra-frame	Inter-frame
Mean	330.3	95.9
Minimum	208.0	37.0
Maximum	587.0	455.0
Standard Deviation	95.5	42.0

Table 2. Motion Estimation Performance:

Scene Sequences	Method	Unpredicted Tiles per frame	% of Total Tiles (1600)
Simple linear motion (Scrolling texts)	SA	249.52	15.595%
	CD	0.34	0.022%
	DM	0.24	0.015%
Non-linear motion (Wiggling fish)	SA	177.89	11.118%
	CD	56.44	3.528%
	DM	57.51	3.594%
High motion & Complex scene (Football)	SA	342.75	21.422%
	CD	105.65	6.603%
	DM	83.66	5.229%

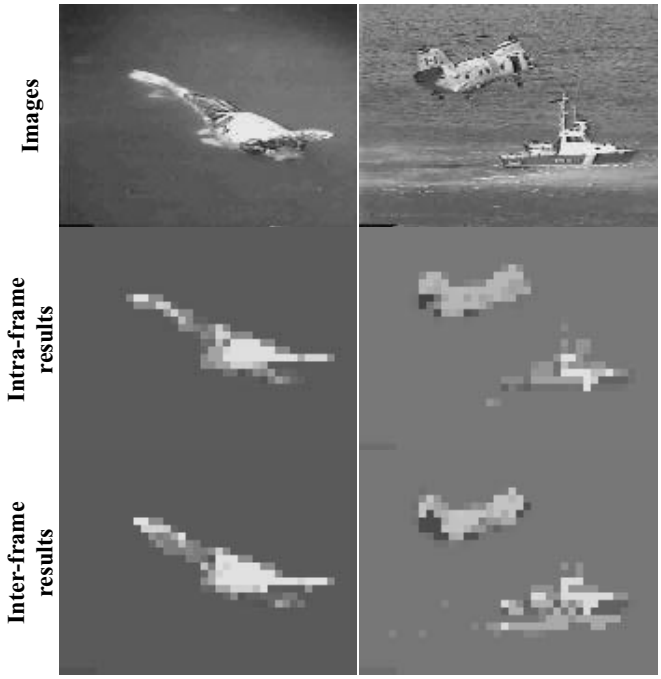
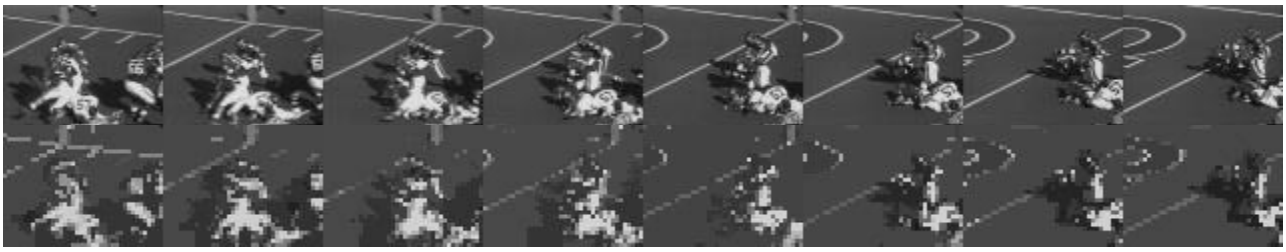


Figure 1. Intra-frame & Inter-frame algorithms results



20 17 18 19 17 16 15 16

Figure 2. Video sequences: original frames are shown on top, followed by composed locales. The numbers below the sequences show the number of locales in each frame.

Figure 3. Object Tracking: the blue jersey of the hockey player is being tracked. Figure shows the blue locale by its envelope and bounding box.