Tracking of objects in spatiotemporal volume by graph-cuts

Braga Natarajan (nnatara@clemson.edu)

1 Abstract

Tracking of objects in the spatiotemporal volume is a recent approach to solving the tracking problem. Graph cut based tools exist for tracking objects in the spatiotemporal volume, but these algorithms are still in the rudimentary stage and they require user interaction for successful tracking. In this paper an attempt to automate the tracking process is done by choosing appropriate energy functions and using local optical flow data with background subtraction techniques. The energy functions used and their respective weight assignments are presented. The algorithm developed is tested with a synthetic sequence and a real sequence. Experimental results show excellent tracking for the synthetic sequence and reasonable tracking for the real sequence.

2 Introduction

'Tracking' is a broad class of problems in the field of computer vision. Generally, tracking an object through an image sequence is defined as the estimation of the object's location in every frame of that sequence. Tracking objects automatically is one of the most challenging tasks in computer vision and researchers have studied this problem for long. Numerous techniques have been proposed and many of the tracking algorithms in existence today can be classified based on 1) the type of application such as vehicle tracking, human head tracking and military target tracking or 2) the element of tracking such as region tracking, contour tracking and feature tracking or 3) the type of approach, which is more general, such as frame-to-frame tracking, incremental tracking and spatiotemporal tracking. Frame-to-frame tracking algorithms operate based on the information contained in a pair of images. Trackers of this type compute object location in any frame using the object's location in the previous frame. Incremental tracking uses information from a group of frames to determine object location. For example, the algorithm may use object trajectory information over the past 5 frames to locate object in the current frame.

Spatiotemporal tracking tracks objects in the spatiotemporal(ST) volume. The ST volume is the volume generated by stacking up all the frames in an image sequence. Hence spatiotemporal tracking uses all the frames in an image sequence for tracking. Although spatiotemporal tracking cannot be implemented in real time because it needs all the frames to start with, it has gained popularity due to its efficiency and its application in video compression [1].

Graph cut algorithms are emerging tools in the domain of tracking/motion segmentation. Although these concepts have been successfully applied to the stereo correspondence problem, not much of work has been done in analyzing motion with these methods. Tracking of objects in the spatiotemporal volume has been achieved only interactively [2] by graph cuts. In this case, appropriate user inputs are required to drive the algorithm. This paper attempts to automate the tracking process by using local optical flow, background subtraction and extra spatiotemporal information. The algorithm uses the min-cut/max-flow algorithm developed in [3]. Preliminary results obtained for some of the sequences show reasonable tracking.

3 Spatiotemporal Graphs

Spatiotemporal tracking needs setting up of the ST graph. ST graphs are three dimensional grid of nodes with the nodes representing the pixels. The image frames are arranged one behind the other in the order of increasing time index to construct an ST graph. An example ST graph is illustrated by figure 1A. The face of the grid contains all the pixels from the first frame and the next slice has the second frame and so on.

The 3D ST graph is shown for a small 3-frame 4x4 image. The vertices represent pixels and lines joining vertices represent edge links between pixel points. The links running between vertices in a

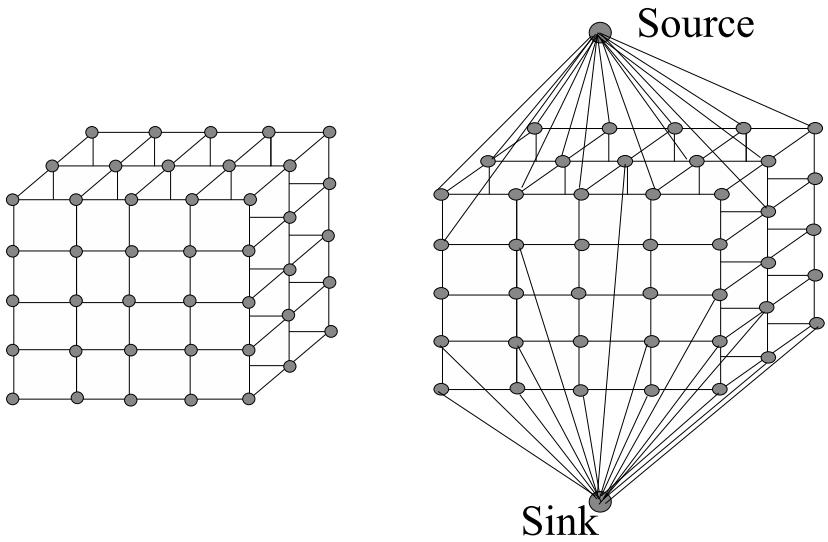

Figure 1: A. An example ST graph, B. Binary segmentation of the graph

single frame are called spatial links and links between nodes in different frames are called temporal link. The idea is to cut this 3D graph into two parts with possibly isolated regions. Each isolated region represents the volume subtended by a single moving object. This binary segmentation maps all the pixels either to the Source terminal or the Sink terminal, which represent motion or no motion. This segmentation, illustrated in figure 1B (not all links are shown for clarity), can achieve tracking of moving objects in the scene.

4 Energy Minimization and Graph Cuts

A classical approach to solving any segmentation problem is the energy minimization technique. The recent graph cut methods provide computationally cheap means to minimize energy functionals. Therefore the segmentation problem is treated as an energy minimization problem with energies associated to surfaces in the spatiotemporal volumes. The energy functional in our case can be expressed by

$$E = E_D + E_S + E_T$$

where E_D represents the cost of assigning labels to pixels, E_S enforces piecewise continuity along the spatial dimensions (x, y), and E_T ensures piecewise smoothness along time axis. Suppose we construct a spatiotemporal graph $G = (N, L)$, consisting of nodes N and links L, and assign edge weights based on the energy functional, graph cut algorithms can be used to find the least energy function. The binary segmentation result can be defined by the following equation,

$$\zeta(x, y, t) = \zeta(p) = s$$

$$s \in \{S, T\}$$

Thus all the points in the 3D graph gets associated with one of the two terminals, $\{S, T\}$, representing source and sink terminals. Point $p = (x, y, t)$ represents any point in the ST graph. The ST graph consists of all the pixels from all the frames, denoted by graph P, and the two terminal nodes. Hence the graph can be written as,

$$G = P \bigcup \{S, T\}$$

The graph G has three kinds of edge links. They are listed below, with spatial link meaning links between nodes in a particular frame, temporal links meaning links between nodes in adjacent frames and terminal links meaning links between points and the terminals. Here, p, q represent nodes and subscripts i, j represent frame numbers.

1. $l_{p_i,q_i}, p_i \neq q_i$, spatial link
2. $l_{p_i,q_j}, i \neq j$, temporal link
3. l_{p_i} , terminal link

Let F represent a set of arbitrary number of nodes in graph G, that is $F \subset G$, and let N_F be some neighborhood for F. Then the three different terms in the energy functional equation E, for the subgraph F, can be expressed as

$$E_D = \sum_{p \epsilon F} D(p)$$

$$E_S = \sum_{p \epsilon F, \{p,q\} \epsilon N_F} B^S(p,q) \ . \ \phi(\zeta(p), \zeta(q))$$

$$E_T = \sum_{p \epsilon F, \{p,q\} \epsilon N_F} B^T(p,q) \ . \ \phi(\zeta(p), \zeta(q))$$

where $D(p)$ represents the cost of assigning pixel p to a particular label. $B^S(p,q)$ and $B^T(p,q)$ represent the cost of dissimilarity between pixels p and q. The function ϕ represents the final dissimilarity of terminal assignments of pixel p and q.

5 Weight Assignments

From the expressions of the energy function terms the edge weight assignments for all types of edges can be set up. This is shown in Table 1.

link	nodes	weight
terminal link	$\{p_i, s\}$	$D(p_i)$
spatial link	$\{p_i, q_i\}$	$B^S(p_i, q_i)$
temporal link	$\{p_i, q_j\}$	$B^T(p_i, q_j)$

Table 1: ST graph weight assignments

Once the graph is generated with all the edge weights and all the labelling, the graph cuts algorithm can be implemented. Let C denote the cut through the graph G. The cut C on G, which is a subset of L, $C \subset L$, is a collection of terminal link edges, spatial link edges and temporal link edges. C satisfies the following properties,

1. $\{p_i, S\} \in C$, iff $\{p_i, T\} \notin C$
2. $\{p_i, T\} \in C$, iff $\{p_i, S\} \notin C$
3. $\{p_i, q_j\} \in C$, iff $\{\{p_i, S\}, \{q_j, T\}\} \in C$ or $\{\{p_i, T\}, \{q_j, S\}\} \in C$

The resulting segmentation defined by cut C can be tuned by recursive graph cuts by adjusting the weights iteratively as in the recent stereo correspondence work in [4]. But then the computational constraints prohibit this method for application in 3D graphs. Hence a proper one-time weight assignments have to be made.

6 Experimental Results

The binary segmentation of the spatiotemporal volume for motion/ no motion has been implemented using the min-cut/ max-flow algorithm of [3]. The codes obtained were compiled to generate the necessary max-flow algorithm files. A version of the graph cut codes that are tuned to compute maximum flow on common image graphs is used. The Vislib library was used to do basic image manipulation tasks. A GUI program was written to get input images from the user and to display the segmentation results.

As a first step, graph construction functions were written specifically for processing a pseudo-synthetic sequence ('bunny' images) exhibiting simple translation motion. Two different functions were written to give the weight assignments, one using background subtraction and other using frame differencing. The ST graph uses the 6-node neighborhood, though 26-node neighborhood can be used to refine results. The weight assignments used for the sequence are listed in Table 2. The weight assignments were based on the likelihood of intensity values of foreground and background pixels.

link	weight
$\{p_i, S\}$	$D_s + I_{p_i}$
$\{p_i, T\}$	$\lvert D_t - I_{p_i}\rvert$
$\{p_i, q_i\}$	$B_s - \lvert I_{p_i} - I_{q_i}\rvert$
$\{p_i, q_j\}$	$B_t - \lvert I_{p_i} - I_{q_j}\rvert$

Table 2: Weights for 'bunny' sequence

The I_{p_i} denotes the intensity value for the pixel p_i. The spatiotemporal graph cuts were performed on blocks of five frames at a time due to computational memory constraints. The segmentation results for the 'bunny translation' video are shown for frames 1,17 and 95. The values of the parameters were set at $D_s = 25$, $D_t = 75$, $B_s = 170$ and $B_t = 130$ for the results shown in figure 2A.

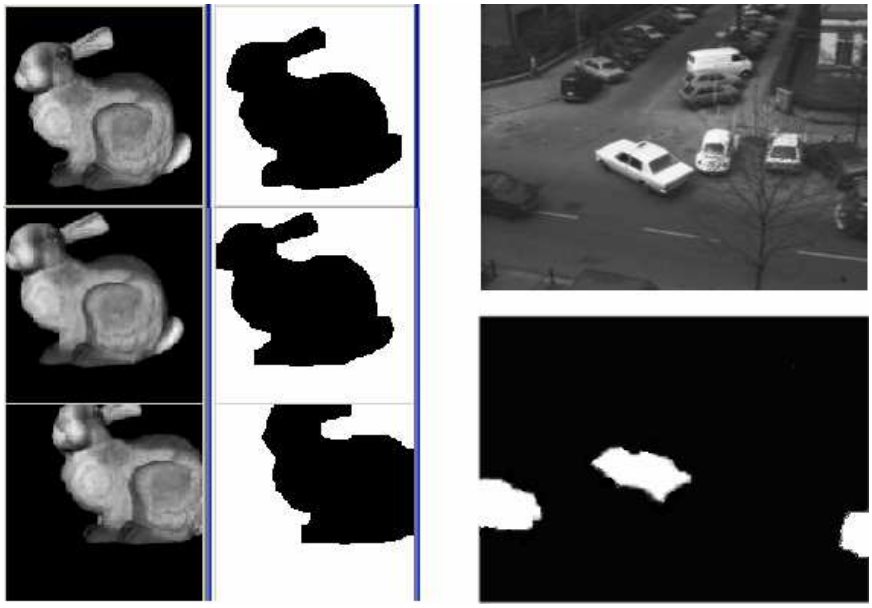

Figure 2: Tracking results for A. 'bunny' sequence B. 'Hamburg' sequence

The results show excellent segmentation and tracking for this synthetic sequence. When the link weights to the source terminal and the sink terminal are approximately the same for all the nodes, then the segmentation is completely governed by the node to node links, which are due to

the boundary term in the energy functional equation. This term works for smoothing out all regions except for discontinuities. Since motion discontinuities in the bunny image relate clearly with intensity gradients, the weight assignments that were discussed above perform good tracking.

The 'Hamburg' sequence is then considered for the spatiotemporal processing. The motion field in this sequence are hard to estimate due to very little texture on the moving objects. Therefore energy functions to track these objects cannot be easily devised. In this case, a sparse local optical flow image is computed at any given frame and this flow image is smoothed. 'OpenCV' functions are used to perform this task. Since the camera is stationary in this case, the background image is constant and it is extracted from the entire sequence containing 41 frames. The smoothed optical flow image and the background are compared in assigning weights to links in the ST graph. Some of the routines in this case involve *ad-hoc* functions and needs refinement.The weights to source and sink from any node depends on the mean of flow field V around its neighborhood. The weight assignments are shown in table 3. A two-frame ST graph was implemented and moving objects were detected and located properly. The location result for frame 10 is shown in figure 2B.

link	weight
$\{p, S\}$	$\frac{1}{N}\sum_{N_p} V(p)$
$\{p, T\}$	$K - \frac{1}{N}\sum_{N_p} V(p)$
$\{p, q\}$	$B - \lvert I_p - I_q \rvert$

Table 3: Weights for 'Hamburg' sequence

7 Conclusion

Many of the tracking algorithms, in existence, can be classified as hybrids of 'batch' and 'incremental' processing. These algorithms do not give a good treatment of performance of feature motion over time and are deficient in considering essential data along the temporal dimension for tracking. Spatiotemporal tracking posses good memory and gives a wholistic approach to the problem of object tracking. This paper deals with the tracking of objects in the spatiotemporal volume by graph cuts. Graph cut combinatorial optimization tools are efficient in processing three dimensional data and hence well suited for our problem. This paper deals with processing of a synthetic and a real sequence. Weight assignments for the ST graphs, of the two sequences, were formulated based on separate energy functionals. The real sequence uses sparse optical flow and background subtraction techniques to adjust weight assignments. Reliable tracking is achieved for the experimental sequences. The energy function used in this paper are crude and attempts to devise generalized energy functions for spatiotemporal tracking is currently being done.

8 References

1. Y. Wexler, E. Shechtman, and M. Irani, Space-Time Video Completion, IEEE Conference on Computer Vision and Pattern Recognition, 2004.
2. Y. Boykov and M.P. Jolly, Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in N-D Images, IEEE International Conference on Computer Vision, 2001.
3. Y. Boykov and V. Kolmogorov, An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision, Intl. Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition, 2001.
4. S. Birchfield and C Tomasi, Correspondence as Energy-based Segmentation, 2004.
5. A. Mitiche, R. Feghali and A. Mansouri, Motion tracking as spatio-temporal motion boundary detection, Robotics and Autonomous Systems, 2001.
6. M. J. Black, Combining intensity and motion for incremental segmentation and tracking over long image sequences, European Conference on Computer Vision, 1992.