Dynamic Video Synopsis

Energy minimization using Iterative Graph Cuts and Loopy Belief Propagation

Aadil Hayat and Manikanta Reddy D
CS676A: Computer Vision and Image Processing
Course Instructor: Prof. Vinay P. Namboodiri

1. Introduction

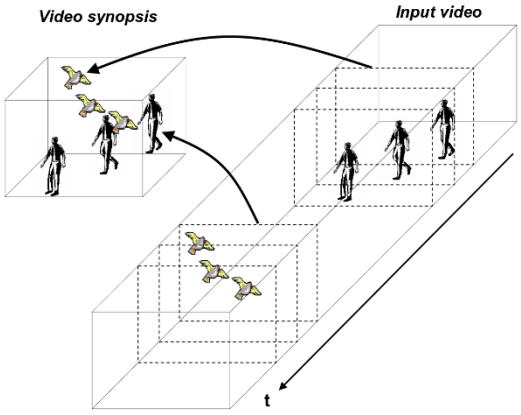
Video synopsis (or abstraction) is a temporally compact representation that aims to enable video browsing and retrieval. Dynamic video synopsis optimally reduces the spatial-temporal redundancy in video. Dynamic video synopsis is itself a video, expressing the dynamics of the scene. The relative timing between activities may change. One of the options presented in this work is the ability to display multiple dynamic appearances of a single object. This effect is a generalization of the "stroboscopic" pictures used in traditional video synopsis of moving objects.

In this project we follow the paper by *Alex Rav-Acha Yael Pritch Shmuel Peleg* on *Making a long video short – Dynamic Video Synopsis*

2. Previous Work

The previous work done on video synopsis includes methods that are based on key frame identification and removal of redundant frames along the temporal dimension. These are good when activity happens in large chunks with gaps in between events. But in cases of surveillance, where the activity happens all the time, but might be limited to a small part of the frame, we'll have a large amount of spatial redundancy.

We'll discuss a method that can reduce both spatial and temporal redundancy in a video.



In this case the key frame based methods would return 6 frames.

An energy based method would result 3 frames, in which both the bird and the person are present at the same moment in time.

3. Activity Detection

We define activity of a pixel as the state of being irregular. Active pixels are defined by characteristic function

$$\mathrm{X} = \begin{cases} 1\ if\ p\ is\ active \\ 0\ if\ p\ is\ inactive \end{cases}$$

In order to remove noise, we use a median filter is applied on X. This activity can be found in general by simple background foreground subtraction techniques.

4. Video Synopsis by energy minimization

Let N and K be the number of frames in the input video sequence and synopsis video sequence respectively. $I(x, y, t)$ represents a pixel in 3D space time volume of Input and $S(x, y, t)$ represents a pixel in 3D space volume of synopsis video.

Characteristics of S:

- K should be substantially smaller than N
- Maximum activity of I should be captured in S
- Object that *go* together in I should go together in S
- S should not be visually fragmented

We'll generate S with above properties by constructing a mapping $M(x, y, t)$ in S as coordinate of source pixel in I. For the moment we also restrict the pixels to come from the same spatial location. Thus any Synopsis pixel $S(x, y, t)$ can come from an input pixel $I(x, y, M(x, y, t))$. The time shift M can be obtained by solving an energy minimization on the energy equation.

$$E(M) = E_{\alpha}(M) + \alpha E_d(M)$$

Where $E_a(M)$ is called the Data cost and represents the activity loss given by

$$E_{\alpha}(M) = \sum_{(x,y,t)\in I} X(x, y, t) - \sum_{(x,y,t)\in S} X(x, y, M(x, y, t))$$

And $E_d(M)$ represents the discontinuity cost defined by

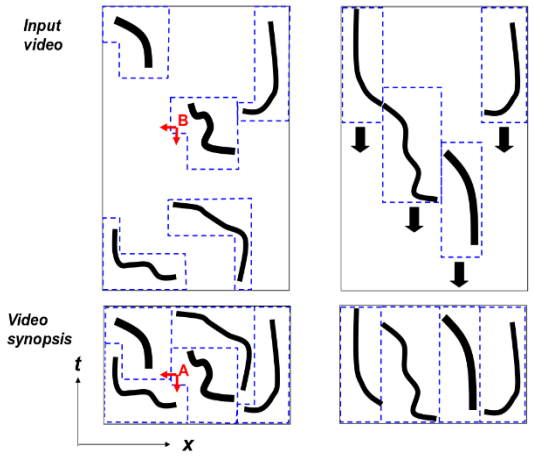
$$E_d = \sum_{(x,y,t)\in S} \sum_{i} \left\| S\big((x, y, t) + e_i\big) - I\left(\big(x, y, M(x, y, t)\big) + e_i\right) \right\|^2$$

As the sum of color differences across seams between spatiotemporal neighbors.

We notice that this energy equation represents a 3D Markov Random Field where each node corresponds to a pixel in the 3D volume of the input video. The weights are determined by the data cost and the weights on edges by the discontinuity cost.

5. Restricted Solution

The optimization of the energy equation directly is very expensive as for a simple problem where the input video is of 3 minutes summarized to 5 seconds results in a graph with approximately 2^{25} nodes, each having 5400 labels!



So we restrict the problem by imposing a condition that consecutive pixels in time in Input should remain consecutive pixels in time in the Synopsis.

The left side of the image represents how seams would move if the optimization was solved for the 3D volume.

The right side shows how seams would move in case of the restriction placed

Thus the label of each pixel $M(x, y)$ now becomes just the frame number t in I shown in first frame of S. Thus the energy equations reduce to

$$E_\alpha(M) = \sum_{x,y,}(\sum_{i=1}^{N} X(x, y, t) - \sum_{t=1}^{K} X(x, y, M(x, y) + t))$$

$$E_d(M) = \sum_{x,y}\sum_{i}\sum_{t=1}\left|\left|S\big((x, y, t) + e_i\big) - I((x, y, M(x, y) + t) + e_i)\right|\right|^2$$

The number of labels possible for each pixel is now $N - K$, where N and K are number of frames in the input and the output videos respectively. e_i now being the 4 spatial neighbors of p.

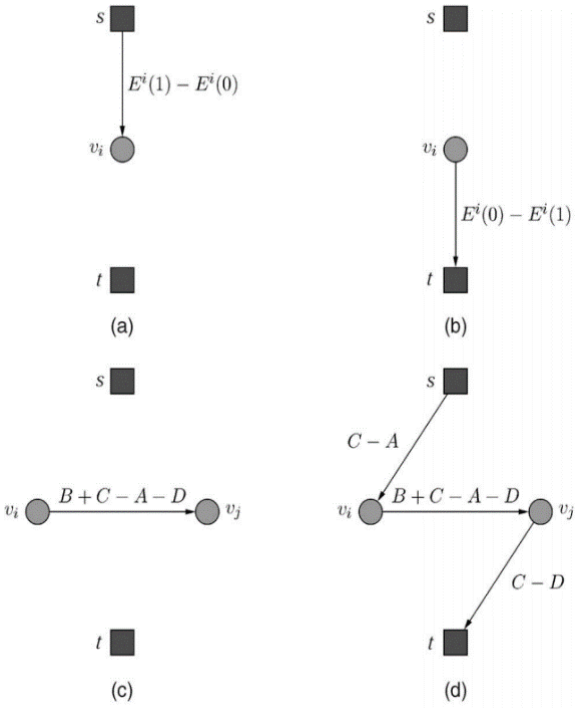
6. Solving the energy equation.

This energy function can be minimized by a number of techniques and we'll be using two of them, Iterative graph cuts and Loopy Belief Propagation. Both of these methods are computationally expensive and have a tunable number of iterations.

7. Iterative Graph Cuts

The energy is minimized by finding the max – flow through a graph by iterating the graph over M.

For this we'll construct a graph whose node capacities are given by the Data Cost and the edge weights by the smoothness cost.



(a) Graph of E^i for $E^i(0) > E^i(1)$ (b) Graph of E^i for $E^i(0) < E^i(1)$. (c) Edge E^{ij} (d) Complete graph for s, i, j, t.

The graph edges are constructed as shown.

When the energy function is of the form $E = \sum_i E^i(x_i) + \sum_{i<j} E^{i,j}(x_i, x_j)$ we can construct the graph as follows.

The graph will contain $n + 2$ vertices including all the nodes $(x_i \; \forall i)$ and two other nodes source(s) and target(t).

First we consider the term E^i depending on only one variable x_i. If $E^i(0) < E^i(1)$ then we add an edge between source and this node, else we add a node between this node and the target.

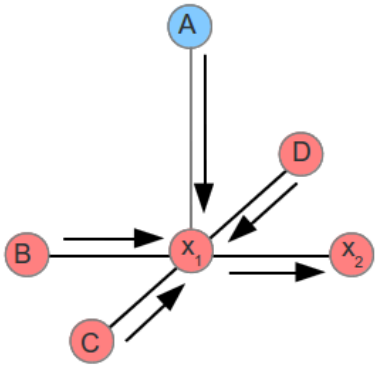
For the smoothness cost, we'll split up $E^{i,j}$ based on x_i, x_j as follows and create edges as shown in the image.

$E^{i,j}(0,0) = A,\ E^{i,j}(0,1) = B,\ E^{i,j}(1,0) = C, E^{i,j}(1,1) = D$. After the construction of the graph, we can efficiently minimize E by computing the minimum $s - t - cut$ on the graph (G). Each cut on G has some cost; therefore G is justified representation of the Energy function.

We have followed the paper by *Vladimir Kolmogrov and Ramin Zabih on What energy functions can be minimized via Graph cuts?* And used the library provided by them to minimize our enrgy function.

8. Loopy Belief Propagation

Loopy belief propagation is a message passing algorithm. A node passes a belief to its neighboring node only after receiving beliefs from its neighboring nodes!



In the figure shown, x_1 weights for messages from A, B, C, D before sending a message to x_2.

A message can be formulated as

$$msg_{ij}(l)$$

Which implies that this is a message from node i to node j about label l. These messages are passed between nodes of the hidden layer only.

In general the scheme of a message is that node i sends messages to node j with information on how probable the label l is the actual label of j.

Based on all such messages received by node j, it forms a belief that its label is something with some probability with an associated cost/penalty for each belief. The belief of node j about its label governs what it believes the label of its neighbor is and what message it sends to its neighbors. Thus in some sense the belief of node i propogated to node j then to neighbors of node j. Hence the term belief propagation.

The loopy belief propagation algorithm is simply an iterative loop over all the message passing routines for a fixed number of iterations and finally a form a belief at every node on what its most possible label is.

The LBP algorithm contains three main components message update, message initialization and formation of belief.

We discussed how message passing is done, now a message is updated by different algorithms. We'll be using the min - sum message update routine.

8.1 Min Sum

The min – sum finds max marginal at each node and operates in log – space. The update is given as

$$msg_{ij}(l) = \min_{\forall labels\ l'} \left(Data(y_i, l') + Smooth(l, l') + \sum_{neighbor\ of\ i\ except\ j\ (k)} msg_{ki}(l') \right)$$

All messages are initialized to 0.

Normalization is straight forward in log – space and is formulated as

$$msg_{ij} = msg_{ij} - \log\left(\sum_{l} e^{msg_{ij}(l)} \right)$$

As this update rule operates in log space its hard to reach overflow/underflow limits.

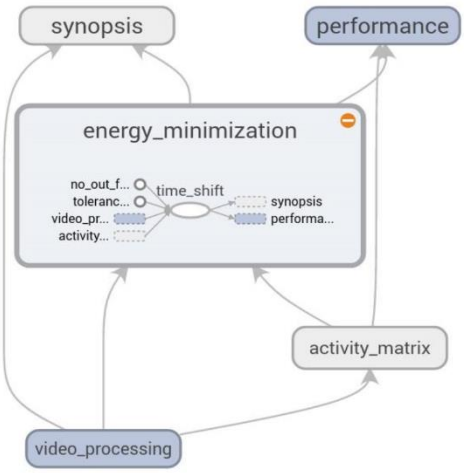
Now the Belief is then given by

$$penalty(x_i = l) = Data(y_{i'}, l) + \sum_{neighbors\ of\ x_i} msg_{ki}(l)$$

The lower the penalty the better the chance that the label is j.

9. Implementation

We've implemented all the routines in the TensorFlow frame work in order to exploit the flexibility of porting all the matrix computation onto nodes for effective computation.



The graph flow of the implementation shows us the independence of different nodes of computation.

The video processing node generates all the frames individually and will be used by all its children nodes for energy computation.

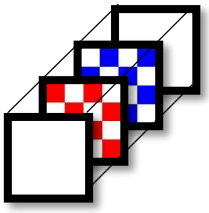
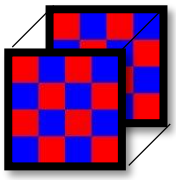
Every node individually can be executed on different architectures.

10. Results

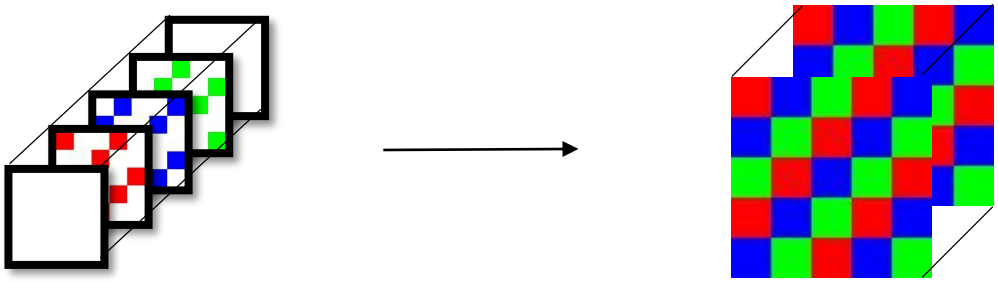
We've tested our examples on numerous toy examples for realizing the energy minimization, as the computation power required for the optimization is really high.

The first toy video consists of 40 frames with first 10 frames being white, the next 10 frame contain alternate pixels colored red and white, the next ten frames contain alternate pixels contain blue and white and the last ten frames are white.

If we were to generate a synopsis of 10 frames, we expect the 10 frames to be full of color with alternating red and blue pixels. Our results match entirely with our hypothesis

Input  Synopsis 

In other example we varied three colors.



As you can see the synopsis generated is of 10 frames with alternating red, blue and green pixels.

11. Conclusion.

In this report we present how dynamic video synopsis can be achieved with stroboscopic effects by solving a minimization on an energy equation. We discuss two methods of energy reduction that provide us with good results.

Loopy belief algorithm can be highly parallelized and can possible decrease the computation time. We can try to formulate similar energy equations in case of moving cameras, which is generally the case in surveillance footages.

12. References

1. Making a Long Video Short: Dynamic Video Synopsis * Alex Rav-Acha Yael Pritch Shmuel Peleg
2. Vladimir Kolmogorov and Ramin Zabih. What Energy Functions Can Be Minimized via Graph Cuts?
3. Loopy Belief Propagation, Markov Random Fields, Stereo Vision. http://nghiaho.com/?page_id=1366
4. Min Xu1 , Stan Z. Li2 , Bin Li1 , Xiao-Tong Yuan2 , Shi-Ming Xiang A Set Theoretical Method for Video Synopsis