A Motion-Tolerant Dissolve Detection Algorithm

Chih-Wen Su1,2, Hong-Yuan Mark Liao1, Hsiao-Rong Tyan3, Kuo-Chin Fan2, Liang-Hua Chen1

Abstract
Gradual shot change detection is one of the most important research issues in the field of video indexing/retrieval. Among the numerous types of gradual transitions, the dissolve-type gradual transition is considered the most common one, but it is also the most difficult one to detect. In most of the existing dissolve detection algorithms, the false/miss detection problem caused by motion is very serious. We present a novel dissolve-type transition detection algorithm that can correctly distinguish dissolves from disturbance caused by motion. We carefully model a dissolve based on its nature and then use the model to filter out possible confusion caused by the effect of motion. Experimental results show that the proposed algorithm is indeed powerful.

Introduction
In the past decade, hundreds of shot change detection algorithms have been proposed [1-4]. Some early researchers focused on the detection of hard cuts, but due to the complicated nature of gradual transitions, general gradual transition detection algorithms are still inadequate. Although there may be numerous types of gradual transitions in a video, these gradual transitions can be roughly categorized into two types: wipes and dissolves (Figure 1). Recently, a large number of researchers have focused on the detection of dissolve-type transitions [5-8]. The common drawback of the existing algorithms is that they cannot distinguish a dissolve transition from

Figure 1. (a) A wipe transition; (b) A dissolve transition

1Institute of Information Science, Academia Sinica, Taipei, Taiwan
2Department of Computer Science and Information Engineering, National Central University, Chungli, Taiwan
3Institute of Information and Computer Engineering, Chung-Yuan Christian University, Chungli, Taiwan
4Department of Computer Science and Information Engineering, Fu Jen University, Taipei, Taiwan
motion. Due to this deficiency, the false detection/misdetection rates of the existing algorithms are still very high.

**Modeling a Dissolve Transition**

Usually, researchers make two assumptions before designing their detection algorithms. First, they assume that the dissolve process should be modeled in a strictly linear manner. Second, they assume that only a little motion or no motion could occur in the two related shots. The above two assumptions are too strict and lead to a dissolve-type transition modeled as follows [5]:

\[ D_t(x,y) = \frac{1}{n} F_t(x,y) + (1 - \frac{1}{n}) G_t(x,y), \forall t \in [0,n], \]  

where \( D_t(x,y) \) is the pixel intensity positioned at \((x,y)\) in the \(t\)-th frame of a dissolve period, \( F_t \) is composed of the last \(n+1\) frames in the first shot and \( G_t \) is composed of the first \(n+1\) frames in the second shot. There are two reasons why (1) is unable to capture the behavior of a dissolve. First, when these local/global motions are mixed up with other non-dissolve events, they are very easy to mistakenly identify as legal dissolves due to their combined characteristics. Second, it is not able to capture the “non-linear” increase/decrease of the pixel intensity in real data. Therefore, we redefine (1) into a new form and assume that the pixel intensity difference between two consecutive frames at \((x,y)\) in an “ideal” dissolve is defined as follows:

\[ D_{t+1}(x,y) - D_t(x,y) = \left\{ \begin{array}{ll} C_1, & \text{if } F_t(x,y) > G_t(x,y) \\ C_2, & \text{if } F_t(x,y) < G_t(x,y) \\ 0, & \text{if } F_t(x,y) = G_t(x,y) \end{array} \right. \]  

where \( C_1 \) and \( C_2 \) are two constants, and \( m \) stands for the number of frames covered by the dissolve period. It is noted that the pixel intensity change is always defined as either an ascending function or a descending function in an ideal dissolve period. Here, we use a sliding window that covers a number of frames to detect the dissolve effect in a video sequence. In order to model the behavior of a dissolve transition, we have to classify the pixels in an observation window into three different categories. The pixels belonging to the first category are called “proponents.” The intensity change of these pixels is either monotonously increasing or decreasing in an observation window. The pixels belonging to the second category are called “fence-sitters.” The intensity of these pixels remains unchanged in an observation window. The third category consists of pixels that do not follow the behavior defined in (2). We call such pixels “opponents.” Since most of the pixels that undergo motion belong to the third category, the above classification scheme is an effective means of distinguishing a real dissolve from local or global motion. Assume that the observation window size is \( L+1 \) frames and it can be chosen as the shortest duration of dissolves in a video, to ensure all dissolves are detectable using the proposed method. In order to compute the numbers of proponents and opponents within an observation window, we define

\[ S_t = \left\{ \begin{array}{ll} \text{# of proponents}, & \text{if } N > \lambda \\ \text{# of proponents+ # of proponents}, & \text{if } N = \lambda \\ 0, & \text{otherwise} \end{array} \right. \]  

where \( N \) is the number of pixels whose intensities have changed within the observation window. \( \lambda \) is used to prevent the false detection of a local dissolve that exits within a small segment of a video. If \( S_t \) is larger than a certain threshold, we believe that there is a dissolve-type transition existing within the window. Figure 2 shows a typical detected dissolve sequence and the detection results.

![Figure 2](image-url)
Threshold Selection

The threshold determination issue is basically an ill-posed problem by nature. In order to effectively separate a dissolve from motion, we have to calculate the maximum possible background value of \( S' \) that is actually caused by non-dissolve factors. First, we assume that only those pixels that have intensity change within the window are considered. Therefore, the probability that the intensity value of a pixel \( p \) will increase or decrease between two consecutive frames can be reasonably set to 1/2. Under these circumstances, the probability that the intensity value of \( p \) monotonously increases/decreases successively within the window \((L+1)\) frames can be computed as follows:

\[
Pr(p \text{ is a proponent}) = \left(\frac{1}{2}\right)^{L+1} \tag{4}
\]

Therefore, we can easily calculate the probability of the remaining cases as follows:

\[
Pr(p \text{ is an opponent}) = 1 - \left(\frac{1}{2}\right)^{L+1} \tag{5}
\]

Suppose there are \( N' \) pixels that undergo at least one intensity value update within the window. Under these circumstances, we can use a binomial expansion based on the events “\( p \) is a proponent” and “\( p \) is an opponent” to express a CDF (cumulative density function) as \((6)\). Here, \( \delta' \) represents the maximum number of proponents that corresponds to the background environment. \((6)\) is used to determine a threshold value \( \delta' \), such that if we use it as a cut-off threshold, then the corresponding \( CDF(S', \leq \delta) \) will approach 1. Since the formula in \((6)\) is calculated under natural (background) conditions, any case that has more than \( \delta' \) proponents in the pseudo binary image can be classified as abnormal and is very likely to be a dissolve.

\[
CDF(S', \leq \delta) = \sum_{k=0}^{\delta'} \binom{N'}{k} Pr(p \text{ is a proponent}) Pr(p \text{ is an opponent})^{N' - k} \tag{6}
\]

Since the number \( N' \) is not a fixed number for different observation windows, we may need different values of \( \delta' \) to make \( CDF(S', \leq \delta) \) approach 1. Through the help of the central limit theorem, we can find a lower bound for the thresholds with different \( N' \). Figure 3 shows the cumulative density functions (CDFs) of binomial distribution of success which were obtained using different trials. The trials here can be regarded as the different active pixel number \( N' \) and each success denotes a “proponent” within an observation window. It is obvious that the shapes of the curves depend on the number of trials. If the value of \( \lambda \) represents the minimum number of trials that can form a valid distribution, then we can be assured that it will take the longest time to saturate as indicated in Figure 3. Therefore, if we use the very moment when the CDF saturates to 1 as a cut-off threshold, then this corresponds to the worst case. Any video transition for which the number of active pixels exceeds the abovementioned threshold can be classified as a dissolve.

Discussion of False Detection and Misdetection of Dissolves

In this section, we focus on the issues of false detection and misdetection of dissolves. First, it is possible that some consecutive frames may show no intensity change during longer dissolve sequences. To correctly detect longer dissolve sequences while maintaining correct detection in the original sequences, we may need different values of \( \delta' \) to make \( CDF(S', \leq \delta) \) approach 1. In contrast to the detection results obtained from the non-sampled video sequence, our algorithm worked better on the sampled video sequence. Second, we have even noted that the probability that the intensity of a pixel either
increases or decreases between two consecutive frames is 1/2 in a non-dissolve sequence. However, a moving region with color shading may not support our assumption. For example, the intensity of the sky as shown in Figure 5(a) decreased vertically. If vertical camera motion exists, the abovementioned region may be identified as a dissolve region. Basically, we can make use of the motion vectors that already exist in an MPEG video to prevent false detection from happening. The algorithms that describe motion vector reconstruction can find a corresponding position in the previous frame. Figure 5 shows an example of detection results obtained with and without motion vectors involved. After introducing the motion vector information, we were able to correct the abovementioned false detections.

**Conclusion**

In this paper, we have proposed an efficient method for detecting dissolve-type gradual transitions. Recall values obtained by this algorithm are in the [77% - 85%] range while precision values are in the [66% - 82%] range. The main contributions of this paper are summarized as follows. First, the threshold of our algorithm can be determined theoretically. A binomial distribution model is used to distinguish real dissolves from motion. Second, since most general motions can be filtered out by using a very low threshold, a real dissolve effect can be easily detected.


**References:**