

Automatic 3D Video Format Detection

Tao Zhang^a, Zhe Wang^a, Jiefu Zhai^a, Didier Doyen^b

^aTechnicolor Research & Innovation, 2 Independence Way, Princeton, NJ, USA 08540

^bTechnicolor Research & Innovation, 1 Avenue de Belle-Fontaine 35576 Cesson-Sevigne, France

ABSTRACT

Many 3D formats exist and will probably co-exist for a long time even if 3D standards are today under definition. The support for multiple 3D formats will be important for bringing 3D into home. In this paper, we propose a novel and effective method to detect whether a video is a 3D video or not, and to further identify the exact 3D format. First, we present how to detect those 3D formats that encode a pair of stereo images into a single image. The proposed method detects features and establishes correspondences between features in the left and right view images, and applies the statistics from the distribution of the positional differences between corresponding features to detect the existence of a 3D format and to identify the format. Second, we present how to detect the frame sequential 3D format. In the frame sequential 3D format, the feature points are oscillating from frame to frame. Similarly, the proposed method tracks feature points over consecutive frames, computes the positional differences between features, and makes a detection decision based on whether the features are oscillating. Experiments show the effectiveness of our method.

Keywords: Stereo, 3D image format, 3D video format, feature detection, feature tracking, 3DTV, checkerboard pattern, horizontal interleaved format, side-by-side format, over-under format, frame sequential 3D format

1. INTRODUCTION

In recent years, 3D films are gaining popularity again. Many movie studios will produce new CG films in both 2D and 3D versions, and more live movies will be shot in 3D. With the success of 3D in cinema, how to bring 3D into home is becoming imminent. However, there are still many challenges. One of the most important problems is the format detection due to the fact that various 3D video formats exist today. Some popular 3D video formats are shown in Figure 1 and an example 3D image is shown in Figure 2. These 3D formats are currently used in different stages in a stereo production chain. For example, the horizontal interlaced format and the checkerboard pattern are popular native display formats in 3DTVs; the checkerboard sub-sampling is also used in the compression and transmission of 3D contents to utilize existing network bandwidth and hardware. In order to correctly process and display 3D contents, their 3D formats must be known beforehand.

The easiest way is to embed 3D format information into the video itself. Many standard bodies such as DVB [1], SMPTE [2] and MPEG are seeking to define 3D standards in order to tag the 3D format information into the video stream. The standardization and adoption process is in progress but not completed yet. The recently introduced HDMI 1.4a [4] adds support for many popular 3D formats. TVs or displays equipped with HDMI 1.4a can easily communicate with a set top box or Blu-ray player to determine the correct 3D format for 3D contents. One condition for HDMI 1.4a to work well in 3D environments is that the source end of the interface knows the 3D format of the content. This condition may be easily satisfied if the source is following a standard. This is the case today with a 3D Blu-ray player and it will be also true in the near future with DVB broadcasting for instance. But this won't be the case all the time especially with user generated content accessible via internet. . It is predictable that multiple 3D formats will co-exist in the near future.

In the meantime, many 3D products require the knowledge of the exact 3D format to work properly. For instance, a set top box needs to know the 3D format information so as to insert text or graphic overlay correctly for 3D contents. A 3D video player needs to know the 3D format in order to play the contents correctly. In some applications, the determination of 3D formats may be performed manually by users, which is tedious and inefficient. Therefore, how to automatically detect the exact 3D format from the video content is a crucial component. In this paper, we propose a novel and effective method to address it.

The remaining parts of the paper are organized as follow. Section 2 describes how to classify 3D formats in this paper; section 3 describes the algorithm to detect those 3D formats that encode both left and right views into a single frame; section 4 describes the algorithms to detect the frame sequential 3D formats; section 5 shows the experimental results; section 6 concludes this paper.

2. 3D FORMAT CLASSIFICATION

There are numerous 3D formats on market. Roughly, these 3D formats may be divided into two classes. The first class encodes the left and right views into a single frame, which is called single frame 3D formats in this paper. Example formats include the horizontal interlaced format, the over-under (also called top and bottom) format, the side-by-side format, the vertical interlaced image format, the checkerboard pattern format, and many color based methods such as Anaglyph. The second class is the frame sequential 3D format, which is a sequence of alternating left/right frames.

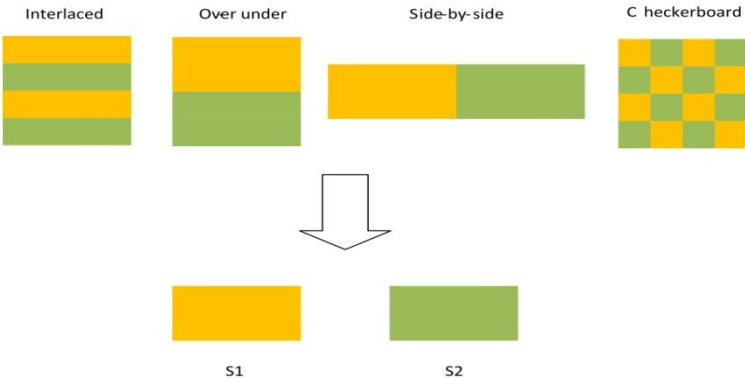


Figure 1. Blending formats (horizontal interlaced and checkerboard pattern), non-blending formats (over-under and side-by-side) and the view reconstruction process to generate the left (S_1) and right (S_2) view images. Different colors mean pixels from different views.



Figure 2. An example of 3D image in the horizontal interlaced format

Single frame 3D formats can be further divided into blending formats and non-blending formats. In blending formats (such as the horizontal interlaced format and the checkerboard pattern), pixels are always surrounded by pixels from the other view; while in non-blending formats (such as the over-under format and the side-by-side format), pixels are surrounded by pixels from the same view except at view boundaries.

The difference between these two classes results in different detection algorithms, which will be detailed in the following sections.

3. SINGLE FRAME 3D FORMAT DETECTION

The detection for single frame 3D formats is based on the following observation: the left and right view images captured by a stereo camera or generated by virtual cameras show similarity in scene structure but differ in depth, since the cameras have identical configurations but with positional differences and view angle differences. The difference in depth is reflected as the variations of positional differences between the same scene points from different views. For a 2D image, this kind of variations of positional differences does not exist.

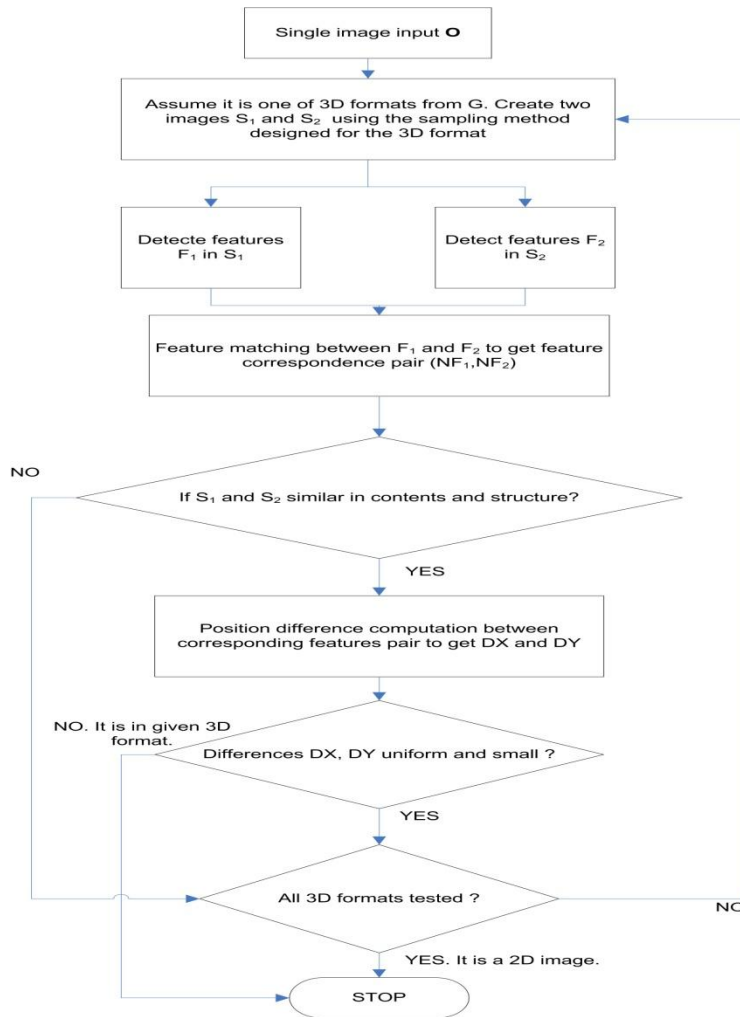


Figure 3. Flowchart for the single frame 3D format detection

The proposed algorithm based on above observation is given in Figure 3, which tests an individual format from the candidate 3D format set G alternatively until a 3D format is identified. The set G consists of 3D formats supported, which may be extended or shrank according to requirements. The default image format for the algorithm is a 2D format. The algorithm roughly consists of following four key steps.

3.1 View reconstruction

Our algorithm deals with those 3D formats that can encode the left view image S_1 and the right view image S_2 into a single frame O , as shown in Figure 1. Without loss of generality, the algorithm may be applied to a single frame instead of a sequence of frames. Before we can proceed, we need to extract the left/right views from the single frame first. The process to obtain S_1 and S_2 from the single frame is called view reconstruction. Every 3D format has its own view reconstruction method, which can be easily defined. For example, for the horizontal interlaced format, the view reconstruction process may extract odd lines as S_1 and even lines as S_2 , see Figure 1. The criteria are that the view reconstruction process should guarantee that the obtained left (right) view not have pixels from right (left) view.

3.2 Feature comparison

The positional differences of same points between different views can be obtained by establishing feature correspondences between views [5, 6]. The features may be feature points, lines or regions [7]. In this paper, we use feature points. The feature detection can be performed on S_1 and S_2 . The obtained features are denoted as:

$$F_1 = \{F_{1i} \mid i = 1, 2, \dots, n_1\} \quad (1)$$

$$F_2 = \{F_{2i} \mid i = 1, 2, \dots, n_2\} \quad (2)$$

where n_1 and n_2 are the numbers of features detected in S_1 and S_2 . The feature correspondences between F_1 and F_2 can be established [7]. This matching process removes those features that do not have correspondences. This new feature sets are denoted by

$$NF_1 = \{NF_{1i} \mid i = 1, 2, \dots, n\} \quad (3)$$

$$NF_2 = \{NF_{2i} \mid i = 1, 2, \dots, n\} \quad (4)$$

where (NF_{1i}, NF_{2i}) is a pair of matching features from S_1 and S_2 , and n is the total number of matching feature pairs. The positional differences can be easily computed from the matching feature pairs as

$$DX = \{DX_i = |x_{1i} - x_{2i}| \mid i = 1, 2, \dots, n\} \quad (5)$$

where x_1 and x_2 are horizontal positions of features from NF_1 and NF_2 .

Above algorithm detects features and establishes feature correspondences. An important variation is to use feature tracking methods [6] instead, which detects features in one image and tracks features in the second image. A feature tracking method [6] is employed in our experiments.

3.3 Blending format detection

For a 2D image, no matter which view reconstruction methods for blending formats are applied, the resulting S_1 and S_2 will be similar in structure and depth. For a 3D image in a blending format D , when the view reconstruction method used is corresponding to the format D , the resulting left and right images will show variations in depth. The variations are reflected as the positional differences of features, see Figure 6 row 1; otherwise, no significant variations (see Figure 6 row 2). So it is possible to distinguish a 3D image from a 2D image, and identify the 3D format by simply applying different view reconstruction methods and examining whether the resulting views have non-uniform positional differences for features.

The positional differences of features DX can be analyzed statistically. A small mean value of positional differences or a non-uniform distribution of the positional differences is an indication of large variations of positional differences. The mean and the standard variation of the positional differences are given by $\mu = \mu(DX)$ and $\sigma = \sigma(DX)$. A decision on whether a 3D format exists can be made by the comparison of μ and σ with some predefined thresholds π_1 and π_2 . If $\mu > \pi_1$ or $\sigma > \pi_2$, the currently tested 3D format is taken as the correct format. In practice, when the distribution of the positional differences is uniform, the positional differences will not be greater than 2 pixels. Our experiments show that $\pi_1 = 1.5$ pixel and $\pi_2 = 1.5$ pixel can provide satisfactory detection results.

3.4 Structure similarity

When applying a non-blending view reconstruction method to an image, The resulting S_1 and S_2 will generally be different in structure (see Figure 6 row 5), with the exception when a 3D image in a non-blending format is applied with its own non-blending view reconstruction method. Obviously, images that are different in structure have non-uniform positional differences of features, which generally lead to a detection error. In our algorithm, if the assumed 3D format is

a non-blending format, a structure similarity check is performed to eliminate possible detection errors. The structure similarity check is not needed for a blending format, since the structure similarity is guaranteed.

Structure similarity has been extensively studied in content-based image retrieval [8]. Since a KLT feature tracking method [6] has been employed in our experiments, we use the loss of features as a measure to determine the structure similarity for simplicity. The loss of features is computed by $L = (n_1 - n)/n_1$. Larger L means more loss of features. This loss rate may be compared with a heuristic threshold value β : if $L > \beta$, then two images are not similar. In our experiments, $\beta = 0.5$.

4. FRAME SEQUENTIAL 3D FORMAT DETECTION

However, the frame sequential 3D format (also called page flip format) is different from single frame 3D formats. In the frame sequential 3D format, the left and right view images are not encoded in a single frame. They are encoded or showed in sequential instead. The detection method proposed in section 3 cannot effectively detect the frame sequential 3D format due to this difference. For single frame 3D formats, the left/right views that encoded into a single frame are taken at the same time. No motion exists between reconstructed views. An important implicit assumption is that the positional differences between features are results of camera positional or view angle differences only. For the frame sequential 3D format detection problem, adjacent frames may not be taken at the same time and motion may exist between them. Motion can also lead to non-uniform positional differences between features. The existence of motion violates the implicit assumption for the detection algorithm for single frame 3D formats. Therefore, we need a new detection algorithm for the frame sequential 3D format.

Since videos encoded in single frame 3D formats show same behavior to a pure 2D video in the following observation, we call videos in single frame 3D formats and pure 2D videos as ordinary videos. Thus, we may detect videos as ordinary videos or frame sequential 3D videos. For ordinary videos, we can further apply the algorithm in section 3 to find out the exact format.

The basic idea for detecting frame sequential 3D video sequences is based on the following observation: for an ordinary video sequence, you will not perceive any scene vibration or oscillation; for a frame sequential 3D video sequence, objects or features away from the screen plane (objects on screen plane have zero parallax) will oscillate in high frequency in the horizontal direction. So if we can detect such oscillation, the sequence is in the frame sequential 3D format; otherwise, it is an ordinary sequence.

The algorithm based on this observation is shown in Figure 4. The algorithm has roughly two steps, which are described below:

4.1 Correspondence determination

Unlike the algorithm in section 3, which can work on a single frame, the new algorithm needs a sequence of frames to work correctly. Let the sequence of frames be f_1, f_2, \dots, f_n , where $n \geq 3$.

This step determines the correspondences between adjacent frames f_1 and f_2 , and computes positional differences between these correspondences. The algorithm is given in section 3.1. For adjacent frames f_i and f_{i-1} , the set of horizontal positional differences in (5) is denoted as DX_i . Let DX_{ij} be the j th positional differences in DX_i .

4.2 Oscillation determination

Suppose we have n frames f_1, f_2, \dots, f_n . For simplicity, we only use one feature correspondence in each frame, DX_{ij} , $i=1..n-1$, j is a fixed number, namely, the j th correspondence in every frame (See Figure 5).

If the sequence is an ordinary video sequence, the non-uniformness of positional differences for a feature is caused by motion only. The change of these differences is generally smooth over a number of frames (see Figure 5, Top). The direction of changes (reflected as the sign of the positional difference) is same over a number of frames.

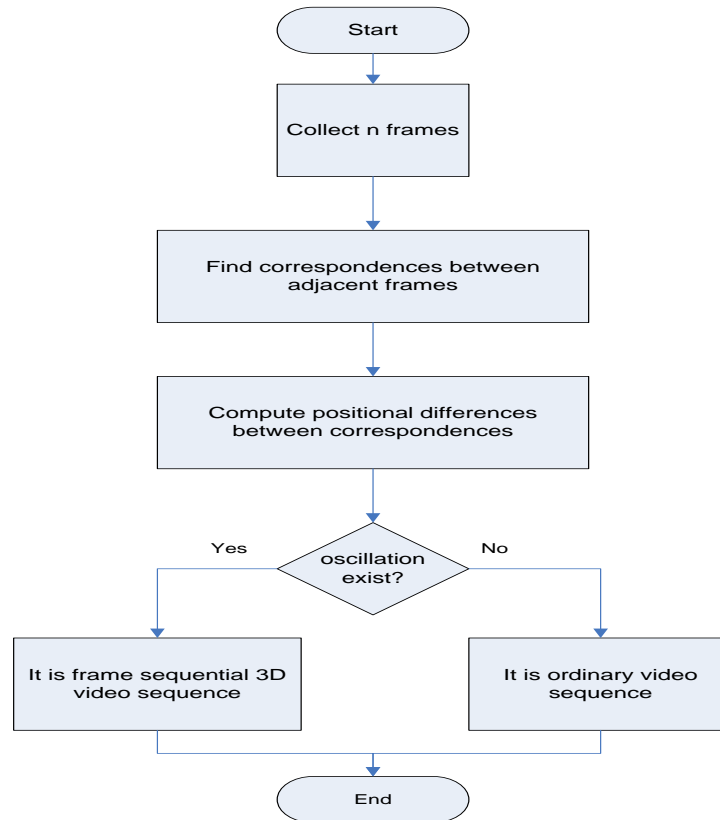


Figure 4. Flowchart for the frame sequential 3D format detection.

If the sequence is a frame sequential 3D video sequence, both motion and stereo camera positional/view angle difference contribute to the non-uniformness of positional differences for features. The positional differences for features are generally larger. More importantly, the sign of positional differences will change for almost every frame if the feature points are located at places away from the TV screen (this means large screen parallax). For example, f_1, f_2, f_3, f_4 are left, right, left, right views, when computing positional differences, (f_1, f_2) are taken as the first pair, (f_2, f_3) are taken as the second pair. Note f_2 is treated as the right view in the first pair, and as the left view in the second pair. f_1 and f_3 are always sequential frames from the same view, which generally has small differences. So the resulting positional difference changes sign for every frame (that is, DX_{1j} and DX_{2j} have different signs). For the same reason, DX_{2j} and DX_{3j} have different signs, and so on. Simply speaking, it is oscillating (see Figure 5, Bottom).

Shaking cameras will cause similar scene oscillation for an ordinary video. However, the oscillation rate from a shaking camera is much lower. The algorithm presented here will not detect ordinary video shot by shaking cameras as a frame sequential 3D sequence.

The oscillation detection method is given below. Let the sequence of frames be $f_1, f_2, f_3, \dots, f_n$, where $n \geq 3$. Let the positional difference sets computed for each frame be $DX_1, DX_2, DX_3, \dots, DX_{n-1}$. Each positional difference set has m common features, which can be accessed by the second subscript. For example, $DX_{1j}, DX_{2j}, \dots, DX_{n-1,j}$ refer to the positional differences for a same feature, the j th feature, from each adjacent frame pair.

1. Select k ($k \geq 1$, such as $k=10$) features with the largest absolute positional difference values. Such features are selected because they will show significant oscillation if any. For simplicity, suppose the selected k is the first k features in $DX_i, i=1 \dots n-1$.
2. Determine the rate of sign change for the k correspondences selected above

$$s = \frac{\sum_{i=1}^{n-2} \sum_{j=1}^k \text{sign}(DX_{ij} \times DX_{(i+1)j})}{k(n-2)} \quad (6)$$

$$\text{sign}(x) = \begin{cases} 1 & x < 0 \\ 0 & x \geq 0 \end{cases} \quad (7)$$

3. If the rate of sign change s is large, then oscillation exists. It is detected as a frame sequential 3D video sequence; otherwise, it is an ordinary video sequence. This may be accomplished by comparison with a threshold value T (such as $T=0.9$). If $s > T$, then a frame sequential 3D sequence; otherwise, ordinary video sequence.

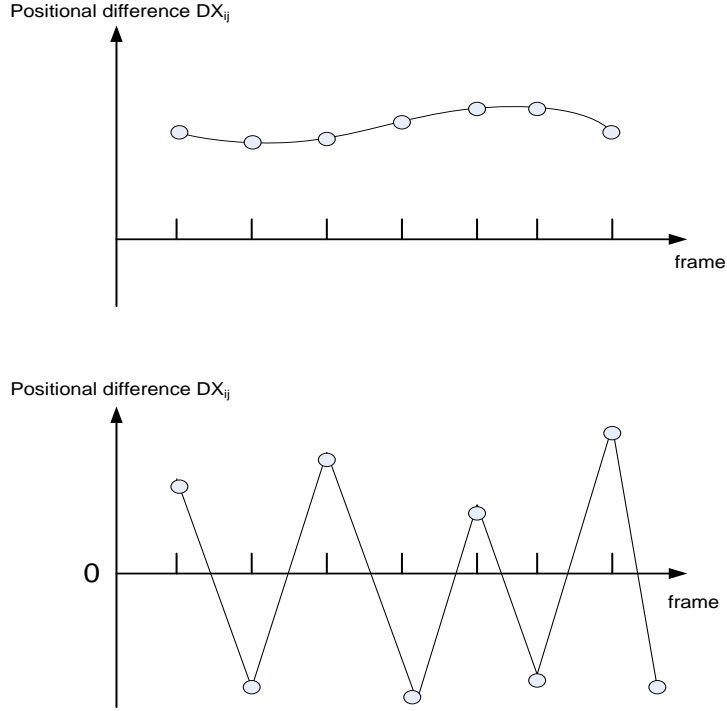


Figure 5. Different positional difference trends for an ordinary video sequence and a frame sequential 3D sequence. The circle represents the actual positional differences at a frame, the lines or curves represent the change trend of the positional differences. Top: ordinary video sequence. The changes are smooth. Bottom: frame sequential 3D video sequence. The changes are abrupt and oscillating. The sign generally changes every frame.

5. EXPERIMENTS

5.1 Single frame 3D format detection

We collected 109 test images. Among them, 9 images are in 2D; 33 images are in the horizontal interlaced format; 27 images are in the side-by-side image format. The other formats (vertical interlaced, over-under, checkerboard and Anaglyph) have 10 test images each. The image sizes are either 1920x1080 or 1280x1040. The test codes are implemented in Matlab, while the feature tracking algorithm is a publicly available C implementation [9]. The maximum number of features to track is 500 features in the experiments. Figure 2 shows a 3D image in the horizontal interlaced format. Figure 6 shows the intermediate results for each assumed 3D format. The results include reconstructed left and right views S_1 and S_2 , detected features, tracked features and their correspondences.

Four tests have been conducted and the results are shown in Table 1:

1. The first one tests only blending formats;
2. The second one tests all 3D formats except Anaglyph;
3. The third one tests all 3D formats including Anaglyph with random orders;
4. The last one is similar to the third one, with the exception that checking order is blending formats, non-blending formats and Anaglyph.

Table 1. Single frame 3D format detection: detection errors and detection rate

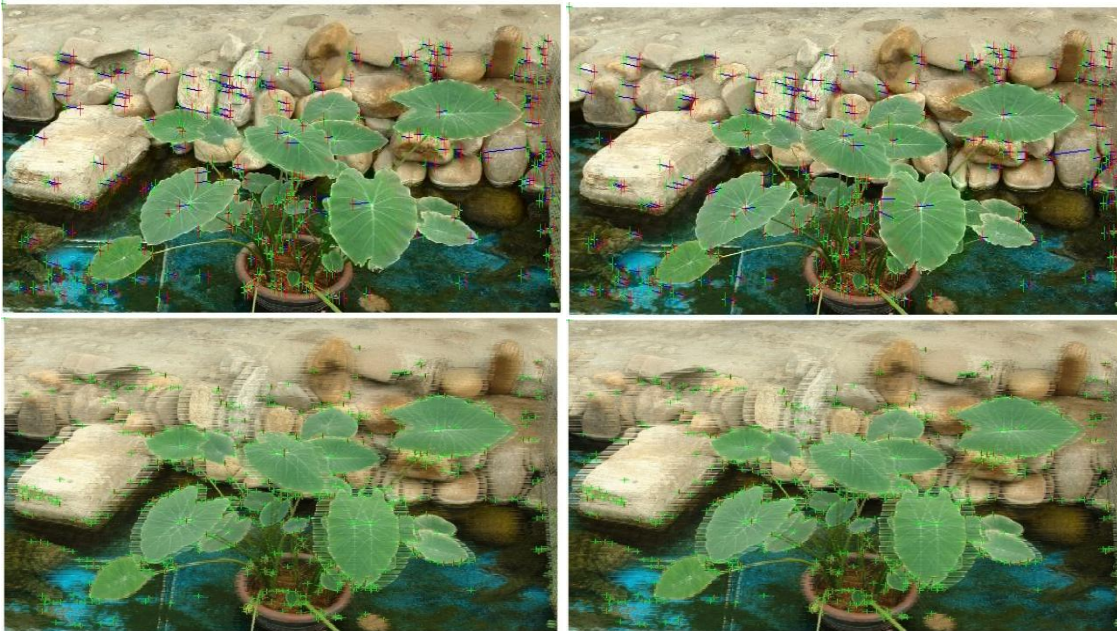
Test	Detection Errors	Detection rate
1	0	100%
2	3	97.2%
3	30	74.2%
4	9	91.7%

From the test results, we can conclude that:

- The proposed 3D format detection by feature comparison is effective in practice;
- The order of checking is important. Blending formats should be checked before non-blending formats, otherwise, passing structure similarity check generally leads to the acceptance of currently assumed 3D format. Such errors can be effectively eliminated simply by limiting checking orders. (refer to test results of test 3 and test 4 in Table 1).
- Feature tracker is the key to improving the algorithm performance since all detection errors can be traced to the feature tracker failure such as the failure to track enough features, or poor handling of intensity changes.

5.2 Frame sequential 3D format detection

We conducted experiments on 3D video for the frame sequential format detection. The test sequences are from a 500 frame 3D movie trailer “Journey to the Center of the Earth”. The test sequences include left view only sequence, right view only sequence and their corresponding frame sequential sequence. In the experiments, 200 point features are tracked over three frames for each possible frame. The overall detection rate is 98% with the threshold of 0.75. The detection failures are due to the feature detection errors, such as when the scene is fading into black, many point features cannot be tracked reliably or lost over frames



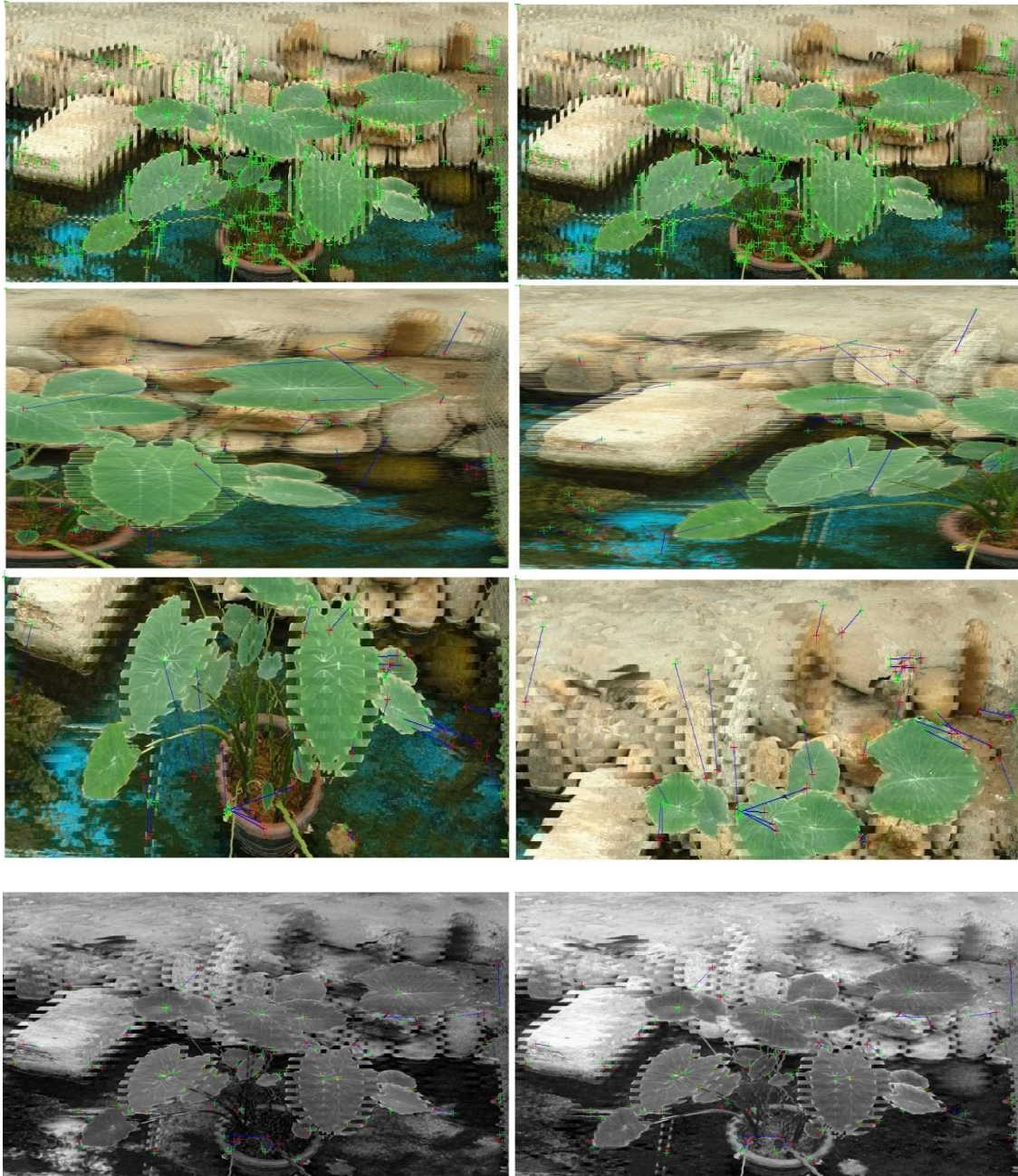


Figure 6. Experimental results when assuming the underlying 3D format is the horizontal interlaced format (row 1), the checkerboard pattern (row 2), the vertical interlaced format (row 3), the side-by-side format (row 4), the over-under format (row 5) and Anaglyph (row 6). The true 3D format is the horizontal interlaced format shown in Figure 2. For each format, the reconstructed left view S_1 and right view S_2 are shown. The red feature points mean the detected features; the green feature points mean tracked features; the blue lines indicated feature correspondences. Lost features are not shown.

6. CONCLUSION

The problem of identifying 3D video formats has been addressed in this paper. This is of great importance for device manufacturers to add support for multiple 3D image formats when the only information is the video itself. We propose two algorithms. The first is a simple algorithm based on feature comparison and structure similarity analysis to address those 3D formats that can encode both views into a single frame. The second one is designed to deal with the 3D frame sequential format. The experiments show our proposed methods are effective for popular 3D formats. The experiments also show that the detection accuracy highly depends on the feature detection and tracking accuracy.

REFERENCES

- [1] Digital Video Broadcasting, <http://www.dvb.org>.
- [2] SMPTE, <http://www.smpte.org>.
- [3] Blu-ray disk association, <http://www.blu-raydisc.com>.
- [4] HDMI, <http://www.hdmi.org>.
- [5] David Lowe, "Object recognition from local scale-invariant features," in Proceedings of the International Conference on Computer Vision, 1999.
- [6] Bruce D. Lucas and Takeo Kanade, "An iterative image registration technique with an application to stereo vision," in International Joint Conference on Artificial Intelligence, pp. 674–679, 1981.
- [7] A. Ardeshir Goshtasbv, 2D and 3D image registration: from medical, remote sensing and industrial applications, Wiley Interscience, 2005.
- [8] Linda G. Shapiro, Computer Vision, Prentice Hall, 2001.
- [9] KLT code, <http://www.ces.clemson.edu/stb/klt/>.