TRICK PLAY SCHEMES FOR ADVANCED TELEVISION RECORDING ON DIGITAL VCR

Hou-Chun Ting and Huesh-Ming Hang
Dept. of Electronics Engineering
National Chiao Tung University
Hsinchu, Taiwan 30050, R.O.C.

Abstract— **In this paper, three technical elements that together implement trick-play modes of digital VCR for ATV recording are proposed: a data arrangement format, a trick-play tracking algorithm, and a coefficient extraction algorithm. First, an efficient tape format is carefully designed for fast forward and reverse playbacks at various speeds. It can accommodate two pickup head configurations. The success of this format relies on a reliable trick-play tracking scheme. This tracking scheme is capable of automatically driving the pickup heads to trace the correct trajectory in typically less than 350ms. Lastly, an adaptive coefficient extraction algorithm is presented, which produces fairly good quality trick-play images under severe bit limitation. A related issue is to choose a balanced bit budget that provides a good comprise between picture spatial quality and temporal quality. Simulations at a video recording bit rate of 25Mb/s have been conducted to demonstrate the effectiveness of our proposed system.**

I. Introduction

To match the functions of the traditional analog VCR, *trick-play* or *shuttle mode* operation is considered to be an essential feature of the all-digital VCR (DVCR). The trick-play mode in a traditional VCR is implemented by simply increasing the tape feeding speed to the desired trick-play speed. Consequently, the pickup head traverses several tracks on the tape and reads in a portion of data stored on each track. However, the picked-up signals from different tracks still form a complete picture field with minor defect caused by the border between two nearby tracks. Although the displayed picture is a concatenation of several non-overlapped segments from different pictures (tracks), the image quality appears to be acceptable when the trick-play speed is not very high. The key reason that makes the above operation working in the analog VCR is that each picture field is recorded evenly on one track.

Unlike traditional VCR, the huge amount of the uncompressed digital video data excludes the possibility of recording them directly on a 1/4 in tape using a low-cost home-use machine. Fortunately, the mature video compression techniques such as MPEG-1 [1] and MPEG-2 [2] make it possible to reduce data rate drastically without noticeable picture quality degradation. Playing time of compressed video on an ordinary tape can go beyond two hours. One significant drawback of the compressed digital data is its uneven bits distribution varying drastically from one part of an image to the other part. The bit rate changes also significantly from picture to picture for an MPEG-like compression technique. In addition, the reconstruction of a certain picture frame may rely on the (decoded) previous (and/or next) pictures. Only the intra-coded pictures can be independently decompressed. This side effect of video compression makes it difficult to produce reasonable video quality in trick playback without a carefully designed tape format and data allocation scheme.

Recently, the draft specification [3] of Advanced Television (ATV) has been submitted for approval for the High Definition Television (HDTV) standard in the United States. The ATV video compression syntax conforms to the MPEG-2 standard at a nominal data rate of approximately 18.4 Mbps. Together with Dolby AC-3 audio compression at a nominal data rate of 384 Kbps, video and audio data are multiplexed and packetized into fixed length packets with description headers. Each packet consists of 188 bytes including a 4-byte link header. In total, a 19.2 Mbps bit stream is used to deliver one ATV program. In order to increase compression efficiency and picture quality, both the predictive (P-frame) and the bi-directional predictive (B-frame) coded pictures are included in the proposed standard. These pictures are the so-called *inter-coded* frames and can not be decoded independently; therefore, they can not be used for trick play.

Along with the development of digital television compression, digital video recording technology also proceeds in parallel toward higher density. Three types of digital video recording systems have been developed and specified, namely, CCIR 601 pictures at 25 Mbps, HDTV resolution pictures at 50 Mbps, and (American) ATV coded bit stream at 25 Mbps [4]. The bit rate reduction algorithms in the first two systems contain only the intra-coded pictures. Since the picture size of ATV is about 6 times larger than that of the CCIR 601 pictures, it is believed that the inter-frame coding techniques must be fully used to achieve good image quality at a bit rate of 19.2 Mbps. However, because the lengths of the non-intra coded pictures vary drastically, it is difficult to implement the trick-play modes for this type of highly compressed video data [5] –[6].

This paper is organized as follows. The basic operations of an ATV DVCR system and the problems we need to solve are described in Section II. Due to the limitation of usable data rate for trick play, it is not efficient to simply duplicate the same data on several locations of which only one can be accessed at one time. To implement an efficient trick-play system, three elements are proposed in this paper. First in Section III, we suggest a tape format which allows us to record the duplicated data only once un-

This work was supported by a grant from Computer and Communication Research Laboratories, Industrial Technology Research Institute, Hsinchu, TAIWAN, ROC.

Contributed Paper
Manuscript received August 21, 1995

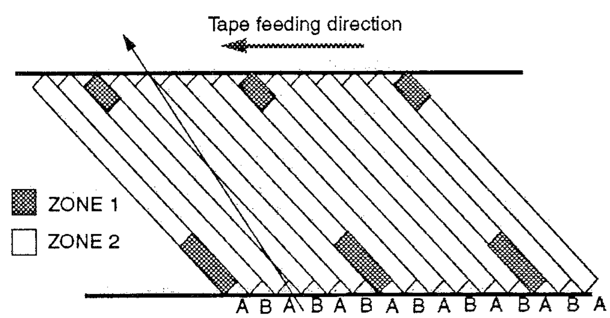


Fig. 1: Trick play scanning path.

der the assumption that the pickup heads can be precisely controlled to trace the data. Hence, a two-stage data path tracking algorithm is developed in Section IV. Motivated by the concept of phased-locked loop, this tracking method can trace and lock into the correct data path within a very short amount of time. Third, the limited data rate available for trick play should be utilized efficiently so that the reconstructed video has the best possible quality. An adaptive coefficient extraction scheme is therefore presented in Section V. It selects the DC component and a portion of the AC components from the intra-coded data based on a picture-dependent threshold. Simulation results in Section VI indicate that the trick-play video quality produced by our system is quite acceptable.

II. High Speed Playback Operation in Digital VCR

Figure 1 illustrates the scanning paths or trajectories of the pickup heads in the high speed playback mode. The regular track pattern (parallel tracks in Figure 1) is generated by the normal speed recording. Typically, adjacent tracks are recorded in opposite magnetic polarization to prevent interference between tracks. The shading area shows an example of the tape area used to store the trick-play data. It is clear that the trick-play data should be placed on the high speed trajectories so that they can be retrieved during high speed playback.

Because the inter-coded data length is irregular in an ATV compressed bit stream, the independently decodable data are randomly scattered on the tape tracks. It is very difficult to pick up the decodable data for trick-play use directly from the normal playback data tracks. A feasible approach is to replicate some of independently coded data and place them on the high speed playback paths. In our scheme, the trick-play data are located in zone 1. In this arrangement, if the pickup head falls in zone 2, no trick-play data can be retrieved as illustrated by the arrow line in Figure 1. Therefore, we need to investigate the data path tracking problem.

At the beginning of running the trick-play mode, there is a transient period in which the tape feeding speed is increased from the normal speed to the desired trick-play speed. If the pickup head falls in zone 1 when the tape speed reaches the desired speed, the trick-play data would be completely recovered. Otherwise, no trick-play data would be retrieved. Trick play tracking algorithm described in Section IV assures that the pickup head would stay in zone 1 and the tape feeding speed is maintained at a stable, desired speed after a short transient period.

In ATV recording, the original ATV transport bit stream is recorded for normal playback. In addition, the trick-play data are extracted from the ATV transport bit stream, packetized and stored in the specific areas (zone 1) on the tape for trick-play modes. Figure 2 is the block diagram showing the operations of an ATV recording system. In the trick-play data extraction process, it is desired to have fewer decoding and/or encoding procedures to reduce hardware complexity. It is often suggested that only the DCT coefficients of I-frames are extracted because they can be decoded independently. Since the DC components are most important among all the DCT coefficients, they are always extracted and replicated on the trick-play trajectories. However, to improve the recovered image quality in trick-play, we manage bit budget in such a way (Section VI) that a portion of AC coefficients can also be included.

III. Tape Recording Format

A carefully designed tape format which satisfies the specification of consumer-use digital VCR [4, 6] is proposed here. To provide versatile services to VCR users, it supports three different trick-play speeds: 3x, 9x, and 27x at both forward and reverse directions. To accommodate different magnetic head configurations of a two-head scanner, one more trick-play trajectory is placed to satisfy the two cases where the two heads are located side-by-side or they are diametrically opposite in 180 degree. Our data allocation conforms to the aforementioned 25 Mbps space limitation for consumer-use digital VCR. Figures 3 to 5 are the data distribution patterns for the 3x, 9x, and 27x forward and reverse trick-play modes. The slant pattern of real tracks is tilted to upright here for easy graphical drawing. The trick-play data are located in the *sync-blocks*. A sync-block is composed of 2 bytes sync-word, 3 bytes identification code, 77 bytes data payload and 8 bytes parity check code. These sync-blocks are marked with different mosaic textures to indicate their usage. One point we like to make here is that most sync-blocks in the 3x mode can be shared by both forward play and reverse play to save space. Unfortunately, the overlapped areas for the other two (9x and 27x) trick-play modes are much smaller, and hence data sharing is not worth considering.

The tables included in these figures describe the exact locations of the sync-blocks allocated to a particular trick-play speed. Because of periodicity, only one period is shown in each table. For example, in the forward 3x playback mode (Figure 3), *head A* picks up sync-blocks 26 to 53 on the 0th track, and *head B* picks up sync-blocks 21 to 40 on the 1st track (solid line, side-by-side head configuration) or on the 3rd track (dotted line, opposite head configuration), depending on the heads geometric configurations.

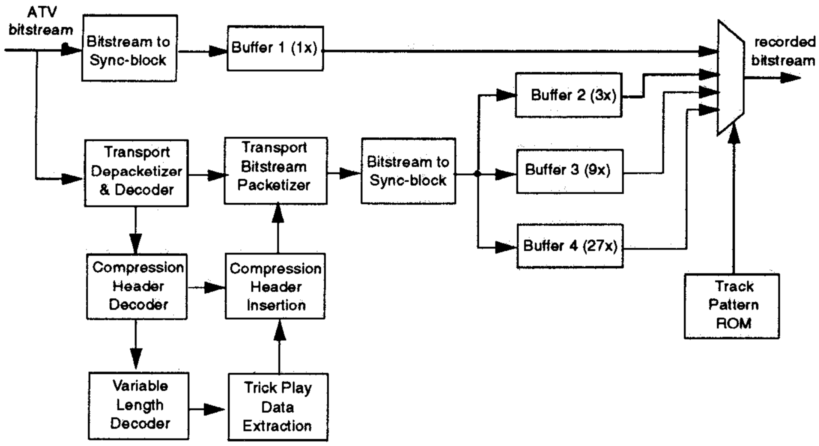
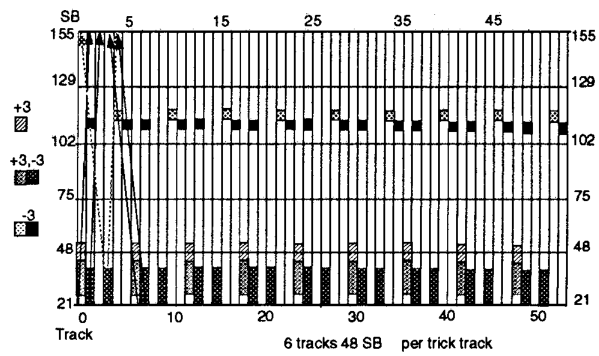


Fig. 2: Block diagram for ATV recording.

In reverse play, on the other hand, *head A* picks up sync-blocks 26 to 45 on the 6th track and blocks 114 to 117 on the 4th track, *head B* picks up sync-blocks 21 to 40 on the 7th track and blocks 109 to 112 on the 5th track (solid line, side-by-side head configuration), or sync-blocks 21 to 40 on the 3rd track and blocks 109 to 112 on the 1st track (dotted line, opposite head configuration), depending on the two-head configurations. In each play direction, 48 blocks are retrieved for every 6 tracks (one scan at 3x fast playback). One may notice that, many assigned blocks (26–45, 21–40) in 3x play speed are shared by the forward and the backward playback modes. Therefore, in total, only 80 sync-blocks per 6 tracks are allocated to 3x trick-play mode for both forward and reverse directions and for two different types of head configurations. On the average, 13.3 sync-blocks per track are reserved for the 3x trick-play.

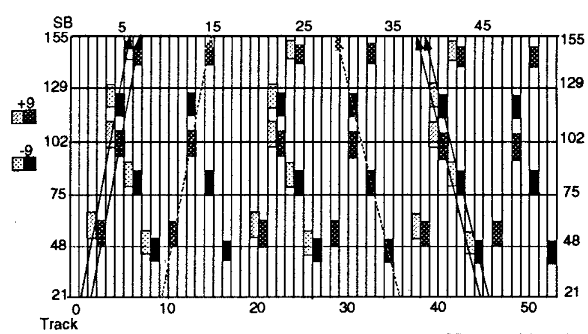
The other two figures can be interpreted similarly. In these two cases, 12 and 4 sync-blocks per track, on the average, are allocated to the 9x and the 27x trick-play modes, respectively. In the digital VCR's specifications [4], 135 sync-blocks per track are allocated for recording data. After subtracting the sync-blocks of all the trick-play modes, around 105 sync-blocks per track are available to store the normal play signal — the ATV input transport bit-stream. Since the pickup heads data scanning rate is about 300 tracks per second, there is enough room to store the 19.4 Mbs normal play data. One factor that complicates somewhat the format design is that the sync blocks corresponding to different speeds can not overlap. Also, we prefer to distribute these trick-play data blocks as evenly as possible so that the tracking algorithm described in the next section could be more reliable.



6 tracks 48 SB per trick track

track	0	1	3	4	5
SB	26-45 46-53	21-40 109-112	21-40	114-117	109-112

Fig. 3: Sync-block allocation for 3x trick-play.



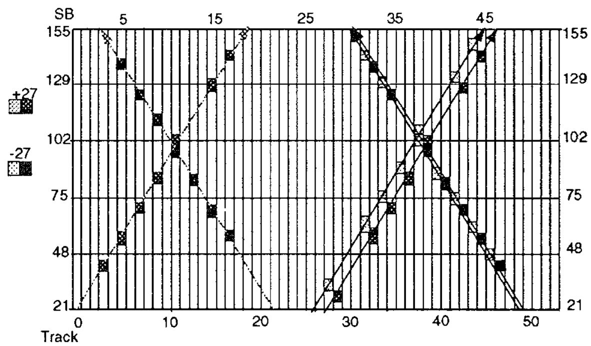
18 tracks 72 SB per trick track

track	2	3	4	5	6	7	8	9	11	13	15	17
SB	53-66	48-61	99-112 119-131	94-107 114-126	80-93 146-153	75-88 141-148	47-55	42-50	48-61	94-107 114-126	75-88 141-148	42-50

Fig. 4: Sync-block allocation for 9x trick-play.

IV. Trick-Play Tracking Methods

Because the trick-play data are distributed only on the assigned scanning trajectories, it is necessary to keep the pickup heads precisely on these trajectories so that they



54 tracks 72 SB per trick track

track	3	5	7	9	11	13	15	17	28	29	32	33	34
SB	41-45	54-60 137-140	69-73 123-127	83-88 109-113	95-100 101-103	81-86	68-72 126-130	55-59 141-145	31-35	26-30	59-65 142-145	54-60 137-140	74-78 128-132
track	35	36	37	38	39	40	41	42	43	44	45	46	47
SB	69-73 123-127	88-93	83-88	100-105 106-108	95-100 101-103	86-91	81-86	73-77 131-135	68-72 126-130	60-64 146-150	55-59 141-145	46-50	41-45

Fig. 5: Sync-block allocation for 27x trick-play.

could retrieve all the corresponding trick-play data. Without loss of generality, the 3x forward trick-play track structure shown in Figure 6 is used as an example to illustrate our tracking algorithm. As described earlier, each track is segmented into a number of blocks, called *sync-blocks*. Each sync-block consists of a sync word, an identification code and a data area [4]. The identification code can be assigned during recording to assist in tracking or recognizing data during playback. In our proposal, one distinct identification code is assigned to each mode that specifies a particular trick-play speed and direction.

Figure 7 is the functional block diagram of the proposed two-stage tracking system for trick-play. The tape feeding speed is controlled by the *servo speed control* block. Magnetic heads in the *scanner assembly* read in the recorded signals from the tape and pass them to the *signal processing* block. The signal processing block then demodulates the recorded signals and also corrects the impaired data. The brain of this tracking system is the proposed *two-stage tracking algorithm* block. It has two inputs. One input comes from the *sync word detection* block that detects the sync-words and counts the number of matched identification codes. This count is then handed to the tracking algorithm block. The other input comes from the conventional pick-up signal sampling and detection circuit. It detects the signal strength difference between the sampled signals from the two magnetic heads and this information is then fed to the tracking algorithm block to keep the heads stay on the correct trajectory. This process is generally called *Automatic Track Finding* (ATF). The blocks enclosed by the dashed line represent the parts which already exist in the current digital VCR specifications. The rest of this diagram are needed to support ATV recording.

The complete tracking operation is divided into two stages. At the first stage, the tape feeding speed is increased to the desired speed and the head position is adjusted at the same time to trace the correct trajectory. We call this stage the *searching stage* and will be the focus of discussions. At the second stage, called *tracking stage*, tracking control is exercised to keep the scanner head from drifting out of the trick-play trajectory after the first stage is completed; that is, the tape speed has reached the correct trick-play speed and the head position has been placed on the right track. It is interesting that although there is only one adjustable variable — tape speed — under our control, two targets, head position and tape speed, can be changed to the desired values simultaneously.

At the beginning of searching stage, the tape driving servo is accelerated towards the desired trick-play speed. During this acceleration period, the scanner heads still pick up the recorded data; however, these data are incomplete and thus can not be decoded to produce pictures. But they can be used to assist tracking in the searching stage. As shown in Figure 6, (line with arrow) some sync-words and identification codes can be detected though the tape feeding speed does not match the trick-play speed exactly. In fact, we do not want to drive the tape speed directly to the final speed as the traditional VCR does. This is because the head position likely falls outside the correct trick-play trajectory and if this happens none of the trick-play data can be retrieved; therefore, we gain very little information about the position of the pickup heads. Instead, the tape initial speed is set to a speed close but not identical to the final speed. An instance is shown in Figure 6. Because there exists a speed difference between the current scanning path (solid arrow line) and the desired one (dashed line), the heads pick up a portion of trick-play data and this information (the number of trick-play data blocks) will be used to tune the tape speed and the heads position. The idea is simply to increase/decrease the tape driving speed iteratively so that the sync block count (in one scan) would gradually approach the correct (maximum) sync block count. Eventually, when we achieve the maximum sync block count, the tape speed matches the desired speed and, at the same time, the pickup heads are positioned on the correct trajectory as shown in Figure 6.

Based on essentially the same principle, two tracking methods for the *searching stage* have been developed. One is called *difference method* and the other is *zero-crossing method*.

A. Difference Method

This method is rather simple. It counts the matched identification (ID) codes, computes the difference between two successive scans and then accelerates or decelerates the tape driving servo aiming at increasing the ID count. Figure 8 is the flowchart of the proposed tracking algorithm. After the trick-play command is issued, the tape driving servo is accelerated toward a pre-determined target speed. During this period, the control algorithm is disabled to avoid making wrong decisions. When the feeding speed reaches the initial target speed $(nV_1 - \Delta V)$, where V_1 is the nominal normal playback speed, n, the desired trick-play multiples, and ΔV, the selected speed increment, we begin to count the matched identification (ID) code num-

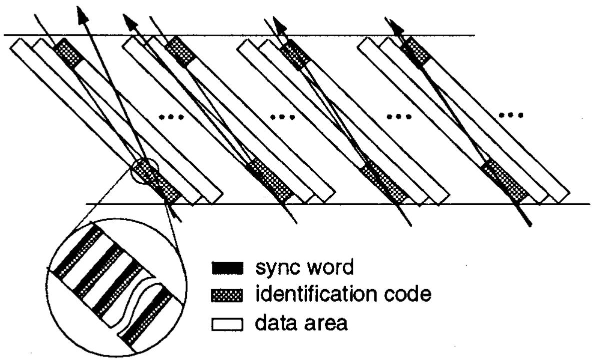


Fig. 6: An example to illustrate tracking algorithm.

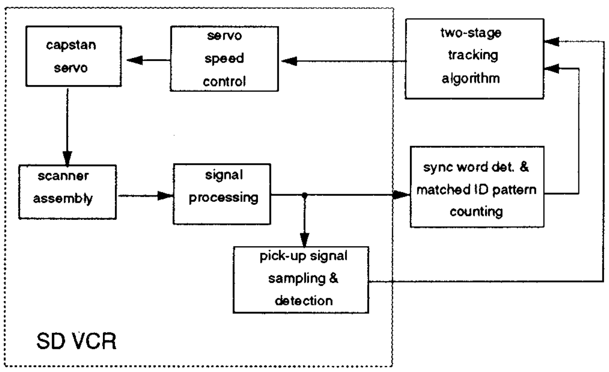


Fig. 7: Block diagram of the two-stage tracking algorithm.

ber C_m in one scan. Then we compare the current ID count, C_m, to the previous count, C_{m-1}. If the heads are already on the right track, we should receive the maximum count C. (C is a constant and represents the number of the allocated sync-blocks for one scan during trick play.) If C_m is smaller than C, there is a mismatch of either tape speed or head position.

It is found empirically that C_m is a quadratic concave function of the distance between the current scanning path and the desired one for our trick-play block distribution pattern. If $C_m < C_{m-1}$, ΔV is divided by L and its sign is reversed. It is found that $L = 4$ is a preferred choice according to our experiments. Then, the tape speed is increased/decreased to the new value, $V_t = nV_1 - \Delta V$, and we again check whether $C_m = C$ or not. The above procedure is repeated until the answer becomes yes. We, then, are on the right track and the tape speed is not changed. If we stay on the right track for several scans we then enter the *tracking stage*; otherwise, we continue adjusting the servo speed. Note that V_1 is the nominal normal playback speed of the player. Although it should be close to the normal playback speed of the recorder, they, the recorder and player speeds, in general are not exactly the same. Therefore, the final ΔV is proportional to the difference between the recorder speed and the player nominal speed.

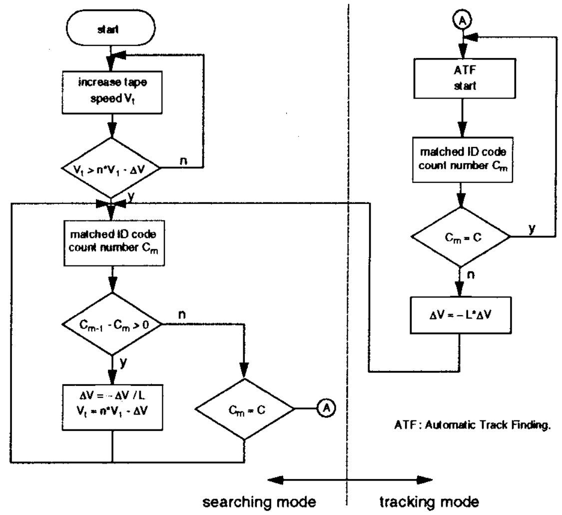


Fig. 8: Difference method.

After entering the tracking stage, tape speed is maintained at V_t. The tracking control is now taken over by a different mechanism which keeps the pickup heads inside the desired trajectory. However, this is a fine tuning mechanism and is effective only when the deviation is less than a track pitch. This mechanism has been widely discussed in the traditional recording system and thus is not described here. In addition to the fine-tune tracking, the ID code counter is continuously running to ensure precise tracking. If $C_m < C$, then we jump back to the *searching stage* with ΔV equal to the last ΔV multiplied by L with reversed sign.

B. Zero-Crossing Method

Phase-locked loop (PLL) is a well-known and powerful technology in communication systems. The problem that PLL tries to solve is rather similar to the tracking problem in DVCR. Namely, the goal is to adjust one quantity (*frequency* in PLL and *speed* in tracking) at the receiver (player) so that this quantity and its integral (*phase* in PLL and *location* in tracking) would match those set at the transmitter (recorder). However, there are differences in these two types of systems as will be discussed below.

The digital version of PLL (DPLL) has been widely investigated and many versions have been developed for digital communications. According to the structure of phase detector, DPLL is classified into four categories [7]: (1) Flip-Flop DPLL; (2) Nyquist Rate DPLL; (3) Zero-Crossing DPLL; and (4) Lead/Lag DPLL. Zero-crossing (ZC) DPLL is the simplest one among them in implementation. Furthermore, the characteristics of its measurements are close to those in our DVCR tracking system.

Figure 9 shows the basic operation of a ZC phase detector. Typically, the *input signal* is an analog signal and

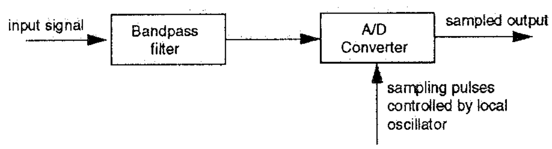


Fig. 9: Phase detector for ZC-DPLL.

the local oscillator produces a pulsive periodic signal. The *sampled output* is a discrete signal having values only on the instances of pulses. Based on the sampled outputs, we adjust the frequency of the local oscillator so that its output (sampling pulses) would match both the frequency and the phase of the input signal. Once they are matched, the sampled outputs become zero.

To track the recorder trick-play speed and locate the correct trajectory, the basic operation of the *searching stage* in DVCR tracking is similar to that of ZC-DPLL. The player trick-play speed and the cut-in position in DVCR play the same roles as the local oscillator frequency and the phase difference between input and local oscillator. The matched identification code count (per scan) C_m can be viewed as the sampled output in ZC-DPLL. But our goal in DVCR is to drive C_m to its maximum value C rather than to drive it to zero. There are other differences between these two systems. For one, the sampled output in ZC-DPLL can be either positive or negative but the count number can only be nonnegative in DVCR. For the other, the nonzero sampled outputs are detected whenever there is a mismatch in either frequency or phase in ZC-DPLL but the nonzero C_m can be measured only when the cut-in position is near the correct trick-play trajectory. Because of these differences, the original ZC-DPLL structure can not be adopted without modifications.

A modified ZC-DPLL structure, shown in Figure 10, is proposed to handle the trick-play tracking problem. The sampled outputs in the original ZC-DPLL converge to zero. To use its design philosophy, we add a bias to our measured inputs. We take the difference between the sync-block count C_m and the maximum count C and use it as the sampled output. However, because this difference value is always nonpositive, the current count C_m must be compared to the previous one, C_{m-1}, to decide whether the current scanning path is approaching or leaving the desired path. Another modification is the addition of a feedback path that takes care of the control signal polarity. Without this, wrong decision may be made if the scanning path crosses over the desired trajectory. A higher order loop filter D(z) could be used to reduce the steady state error as described in [7]. A high order filter can also provide a wider range of tolerance for the speed discrepancy between different recorders. However, a simple delay element seems to be sufficient for our system.

The loop gain k is an important parameter which affects the stability of this feedback structure. If k is small, the system stability is ensured but the transient time to the desired state is often long. If k is very large, the entire system could be unstable. It is rather difficult to derive the exact mathematical model of the sync-block count as a function of tape speed, because it depends on the tape format and the input/output relation of this nonlinear feedback system. Therefore, the trial-and-error approach is used to find appropriate k values. Equation (1) shows how the k value affect the servo control signal ΔV_{m+1}:

$$\Delta V_{m+1} = k \cdot f(C_m - C_{m-1}) \cdot (C - C_m) \cdot f(\Delta V_m) \cdot V_1, \quad (1)$$

where V_1 is the nominal normal playback speed (a constant at player), and $f(\cdot)$ is the thresholding function, i.e.,

$$f(x) = \begin{cases} 1, & \text{for } x \geq 0 \\ -1, & \text{for } x < 0. \end{cases} \quad (2)$$

As mentioned earlier, a nonzero C_m is received only when the current cut-in position is near the desired trick-play path. In this case, we do not know the head location other than the fact that it is away from the correct trajectory. This situation is different from that of ZC-DPLL, where the sampled output value always indicates the phase difference. The cut-in position is random in real situation. To shorten the transient period, the initial speed deviation ΔV can not be too small. When ΔV is small, two consecutive scanning paths would be close to each other. Thus, if the initial cut-in position is away from the correct trajectory, it would take many scans before the pickup heads hit a portion of the correct trajectory and start sensing sync-blocks. Only then, C_m can provide information about the distance between the current head location and the correct trajectory. Consequently, the zero crossing feedback structure begins to work effectively. Therefore, the choice of k value becomes a dilemma because k needs to be small enough to ensure stability. A way to solve this problem is to change k during the trajectory searching process [8]. We begin with a larger value, which is about 0.02 to 0.04 in our simulations. And then we divide k by L whenever C_{m-1} is zero and C_m is nonzero. It is found empirically that $L = 4$ seems to be a good value, which happens to be the same as the L value in the difference method in the previous subsection. Once the correct trajectory is locked, the process enters the *tracking stage* and the fine-tuning speed control over there is the same as before.

C. Simulation Results

We simulate the above two tracking methods using the designated tape formats in Section III. In our simulation, the tape feeding servo acceleration is assumed to be a fixed constant in both directions. The value of acceleration (depending on the motor) only affects the transient time; it does not affect the characteristics (such as stability property) of both methods. Figure 11 shows the simulation results of the 9x forward trick-play case. The solid lines represent the outcomes of the zero-crossing method and the dashed lines represent those of the difference method. Both methods are stable as demonstrated by these figures.

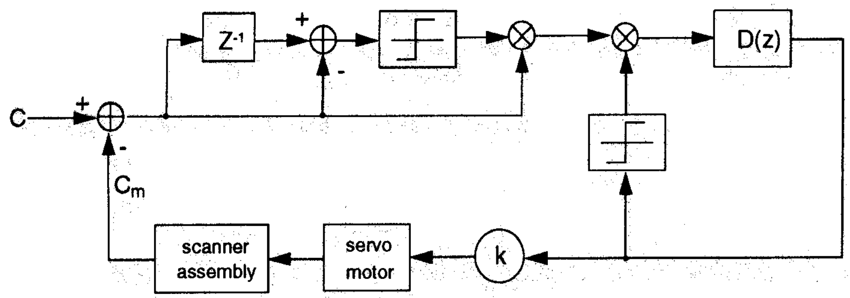


Fig. 10: Block diagram of zero crossing method.

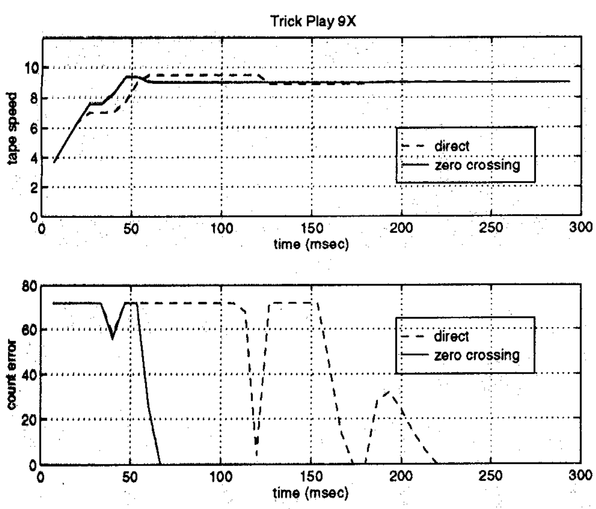


Fig. 11: Comparison of difference method and zero crossing method.

Although we did not perform detailed mathematical analysis on the proposed tracking schemes, exhaustive simulations have been conducted for every cut-in position spacing 1 μm apart. Table 1 summarizes the entire set of results. It shows the average settling time (the time spending on the *searching stage*) of all the tested cut-in positions in each trick-play mode. It is clear from these results that the performance of the zero crossing method is better than that of the difference method. From the set of simulation data we gathered, it is interesting to observe that the peak settling time is due to mostly the long elapsed time — the time before receiving the first nonzero ID count C_m. As discussed before, increasing the initial k value could reduce this elapse time, but it then increases the time needed for the scanning path to converge to the correct path. There may exist a best trade-off in selecting and adjusting the k value. However, we only tried a number of initial k values with a heuristic adjustment algorithm as described in the above. Nevertheless, the results are satisfactory.

V. Adaptive Coefficient Extraction for Trick Play

The third element of an ATV DVCR system is to select the compressed data for reconstructing pictures in trick-play modes. We propose a coefficient extraction criterion based on the allocated bit rate budget and the magnitude of DCT coefficients. Using the methodology of MPEG, the DCT coefficients of a block are quantized and coded by a variable-wordlength table, which is a combination of the well-known run-length coding and Huffman coding. Among all the DCT coefficients, the DC coefficient is coded separately because of its importance and distinct characteristics. On the other hand, AC coefficients are concatenated to form runs. A (nonzero) run indicates a variable number of zero coefficients followed by one nonzero coefficient. To reduce hardware complexity, it is preferred to extract AC coefficients at the borders of runs because no further data decoding/encoding processes are needed for extracting and recording purpose. To decide which runs should be extracted, an adaptive extraction criterion derived based on the image contents complexity and the allocated bit budget is developed to achieve better image quality.

Let A_n represent the absolute value (or the squared value) of the first nonzero AC coefficient in the nth DCT block of an I-frame. We then calculate the AC coefficient average $A = \frac{1}{N}\sum_{n=1}^{N} A_n$, where N is the total number of blocks in an I-frame. Observing the fact that the A value is an indication of the image complexity, a threshold $T = w(B - B_{dc}) \cdot A$ is defined, where B is the bits allocated to a decoded trick-play picture, B_{dc} is the bits consumed by the DC coefficients of all the image blocks, and $w(\cdot)$ is a non-increasing weighting function. If A_n is larger than T, the first run and the associated AC coefficient of the nth DCT block is extracted and followed by EOB (end of block) code; otherwise, it is discarded. We repeat the above procedure until all the bits are running out. Because $w(\cdot)$ is a function derived empirically and because the image contents vary drastically form sequence to sequence, the optimum T value should be tuned for different image sequences. To fully utilize the very limited trick-play space and also to take advantage of the temporal correlation be-

Table 1: Average settling time for various trick-play modes.

Trick Play Mode	3x	-3x	9x	-9x	27x	-27x
Difference Method	148ms	148ms	222ms	280ms	383ms	446ms
Zero-Crossing Method	95ms	90ms	125ms	133ms	289ms	311ms

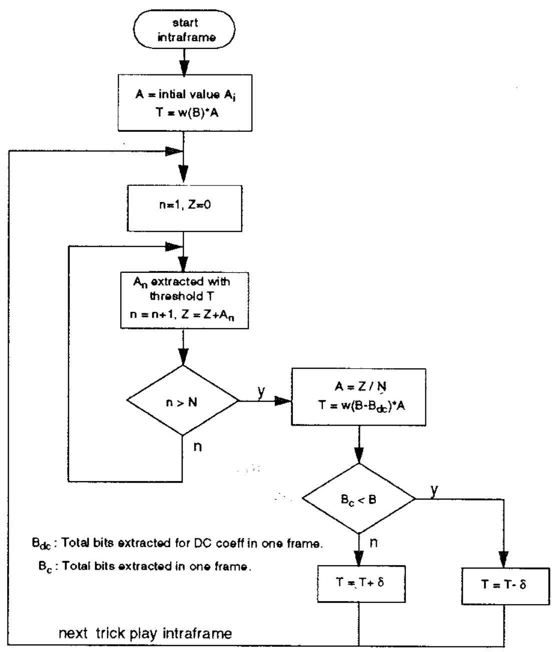


Fig. 12: Adaptive coefficient extraction.

tween nearby image frames, T value is modified to $T+\delta$ or $T-\delta$ depending upon whether the allocated budget B has been consumed completely or not in the previous extracted frame. Figure 12 shows the flowchart of this adaptive extraction algorithm. To simplify hardware implementation, A and B_{dc} can be calculated from the previous trick-play frame instead of the current frame.

VI. Bits Allocation and Simulation Results

Here comes another issue for trick-play. There is a trade-off between the spatial resolution and the temporal resolution of the displayed trick-play sequence. The spatial resolution here is referred to the still image quality and the temporal resolution is the smoothness of movement in a video sequence. Since the total available bits number is fixed, if we allocate more bits to a coded frame, fewer frames can be coded. Hence, we have better spatial resolution (each displayed picture looks good) but the object motion is jerky. On the other hand, a higher coded frame rate leads to smooth motion but lower spatial image quality. After observing the decoded sequences of both high and low frame rates, we feel that under the severe limitation of trick-play bit rate, it is preferred to have a reasonable spatial image quality with lower temporal resolution.

Another limitation in selecting displayed frames comes from the GOP (group of pictures) structure in the MPEG standards. There is only one I-frame per GOP. For example, if GOP=9, we have only one I-frame for every 9 pictures in the MPEG-compressed bit stream. Therefore, the true 3x trick-play mode, one displayed picture per 3 frames, cannot be implemented unless we want to go through the complicated decoding/encoding processes at the recorder. And even we want to do so, the limited available bits for trick-play would lead to very poor spatial quality pictures. A simple approach but with more acceptable overall video quality is to extract the only one I-frame for every 9 frames. In other words, one extracted picture is displayed for 3-frame time in the 3x trick-play mode.

Under the consideration of all the above factors, the displayed picture sequence arrangement for three popular GOP numbers are designed as shown in Figure 13. If the GOP number is 9, each extracted frame is repeated three times for all trick-play modes. Similarly, each extracted frame is repeated four and five times for GOP=12 and GOP=15, respectively. Using the tape formats defined in Section III, the number of allocated sync-blocks for ±3x trick-play modes is 24 blocks per track on the average. Though the average allocated sync-blocks is 36 for ±9x and ±27x trick-play modes, only 2/3 of them store the recorded trick-play data. The rest are reserved for further error correction because the error probability is higher for faster trick-play modes. A two-head scanner assembly rotates at 150 rounds per second is taken as an example here. In this case, 300 tracks per second are accessed. If the picture refreshing rate is 30 frames per second and the extracted frames are repeated n times, the bit budget of each reconstructed trick-play frame is calculated as follows,

$$\begin{aligned} B &= 300\ tracks/sec \cdot \frac{n}{30}\ sec/frame \cdot 24\ SB/track \\ &\quad \cdot 77\ bytes/SB \\ &= n \cdot 147840\ bits/frame \qquad (3) \end{aligned}$$

The bit budgets corresponding to three GOP structures are summarized in Table 2. The bit budget for GOP=9 is less than those of the other two structures, but its picture refreshing rate is higher. Hence, the image temporal resolution for GOP=9 is better and could compensate somewhat the loss in spatial quality due to lower frame budget. Because the bit budget is the same for all the trick-play modes for a given GOP structure, only one coefficient extraction process (and one set of parameters) is needed.

Simulations are conducted to demonstrate the effectiveness of the above bit allocation scheme and the coefficient extraction method. Three speeds — 3x, 9x and 27x — at

Frame	0	3	6	9	12	15	18	21	24	27	30	33	36	39	42	45	48	51	54	57	60	63	66	69	72	75	78	81	84	87	90	93	96	99	102	105	108
GOP = 9																																					
Intra	0			9			18			27			36			45			54			63			72			81			90			99			108
±3	0	0	0	9	9	9	18	18	18	27	27	27	36	36	36	45	45	45	54	54	54	63	63	63	72	72	72	81	81	81	90	90	90	99	99	99	108
±9	0			0			0			27			27			27			54			54			54			81			81			81			108
±27	0									0									0									81									81
GOP = 12																																					
Intra	0				12				24				36				48				60				72				84				96				108
±3	0	0	0	0	12	12	12	12	24	24	24	24	36	36	36	36	48	48	48	48	60	60	60	60	72	72	72	72	84	84	84	84	96	96	96	96	108
±9	0			0			0			0			36			36			36			36			72			72			72			72			108
±27	0									0									0									0									108
GOP = 15																																					
Intra	0					15					30					45					60					75					90					105	
±3	0	0	0	0	0	15	15	15	15	15	30	30	30	30	30	45	45	45	45	45	60	60	60	60	60	75	75	75	75	75	90	90	90	90	90	105	105
±9	0			0			0			0			0			45			45			45			45			45			90			90			90
±27	0									0									0									0									0

Fig. 13: Available frames during trick-play for different GOP structures.

Table 2: Trick play bit budgets vs. GOP.

GOP structure	9	12	15
B (bit budget)	443 Kb	591 Kb	739 Kb

Table 3: Trick-play picture average PSNR vs. GOP.

GOP structure	9	12	15
normal play	34.8 dB	34.8 dB	34.9 dB
trick play	23.1 dB	24.1 dB	24.1 dB

both forward and reverse directions have been included in our tests. An HDTV video sequence with 1920 x 1080 resolution is used in simulation at a compressed bit rate of 18.4 Mbps. This image sequence is coded using GOP=15; therefore, the average bit budget is 739 Kb/frame for storing the trick-play data. The recovered trick-play picture is shown in Figure 14. It has been suggested that only the DC components are retrieved for trick-play pictures. However, its picture quality, as shown in Figure 15, is clearly less favorite when compared to the one (Figure 14) including some runs of AC coefficients. Table 3 shows the average PSNR for several GOP structures using the proposed trick-play system. The PSNR of trick-play pictures is lower than that of the normal playback pictures for about 10 dB. Since the trick-play pictures are typically used for contents searching purpose, their quality observed on monitor in real time seems to be quite acceptable.

Fig. 14: Recovered trick-play picture of an HDTV sequence.

VII. Conclusions

In this paper, we propose three elements in a DVCR system for implementing trick-play functions. They are 1) an efficient data arrangement format for trick-play modes, 2) a trick-play tracking algorithm to precisely guide the

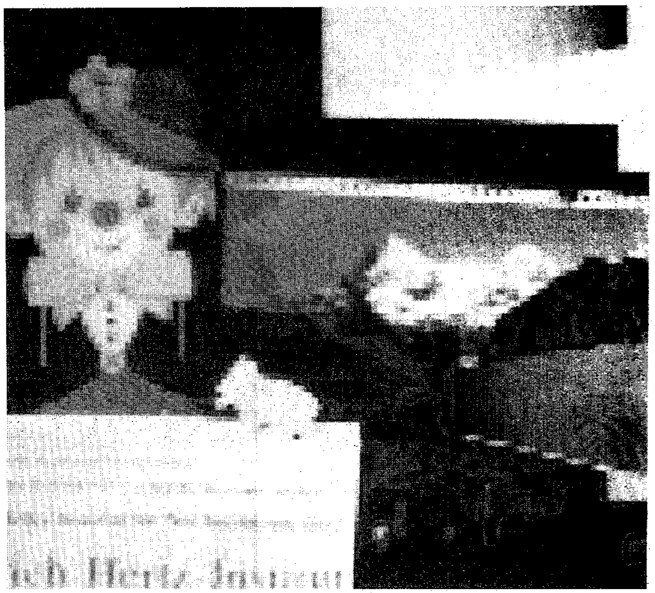


Fig. 15: Recovered picture with DC coefficients extracted only.

heads tracing the correct trajectories, and 3) a coefficient extraction algorithm to achieve rather good trick-play image quality. Particularly, the tape format for ATV recording with forward and reverse trick-play modes at 3x, 9x and 27x speeds is presented in detail. This tape format can accommodate two different two-head configurations. Extensive experiments show that the proposed zero-crossing tracking algorithm converges typically in less than 350ms. This algorithm is fairly simple and thus lends itself to easy hardware implementation. Also, we propose an adaptive coefficient extraction algorithm which selects the most important AC coefficients based on a picture-dependent threshold. Our simulations demonstrate that this algorithm produces acceptable picture quality within the designed bit budget. Limited by available tape space, bits assigned to a trick-play image frame is quite small. The compromise between spatial and temporal resolution has been carefully made in our design to produce a good overall visual quality.

References

[1] ISO-IEC MPEG CD 11172-2 (rev.1), "Coding of moving pictures and associated audio — for digital storage media at up to about 1.5 Mb/s, part 2: video," Nov. 1991.

[2] ISO-IEC/JTC1/SC29/WG11, "Test Model 2," Document MPEG92/No.245, July 1992.

[3] Grand Alliance Specialist Groups, "Grand Alliance HDTV System Specification Draft Document," Feb. 1994.

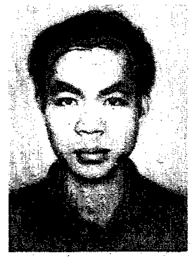
[4] HD-Digital VCR Conference, "Basic Specifications for Consumer-Use Digital VCR," Aug. 1993.

[5] J. Boyce and F. Lane, "Fast scan technology for digital video tape recorders," *IEEE Transactions on Consumer Electronics*, Vol.39, No.3, pp.199-204, Aug. 1993.

[6] F. Azadegan et. al., "Data-placement procedure for multi-speed digital VCR," *IEEE Transactions on Consumer Electronics*, Vol.40, No.3, pp.250-256, Aug. 1994.

[7] W. C. Lindsey and C. M. Chie, "A survey of digital phase-locked loops," *Proc. IEEE*, Vol. 69, pp. 410-431, Apr. 1981.

[8] L. F. Rocha, "Simulation of a discrete PLL with variable parameters," *Proc. IEEE*, Vol. 67, pp. 440-442, Mar. 1979.



Hou-Chun Ting was born in Changhua, Taiwan, R.O.C., in 1964. He received the BS, MS in Electrical Engineering from National Tsing Hua University, in 1986 and 1988 respectively. He is currently working toward the Ph.D. degree at National Chiao Tung University. His research interests include image processing and video coding.

Hsueh-Ming Hang received his B.S. and M.S. degrees from National Chiao Tung University, Hsinchu, Taiwan, and Ph.D. in Electrical Engineering from Rensselaer Polytechnic Institute, Troy, NY, in 1984. From 1984 to 1991, he was with AT&T Bell Laboratories, Holmdel, NJ, where he was engaged in digital image compression algorithm and architecture researches. He joined the Electronics Engineering Department of National Chiao Tung University, Hsinchu, Taiwan, in December 1991. His current research interests include digital video compression, image/signal processing algorithm and architecture, and digital communication theory.

Dr. Hang holds three US patents, with two pending, and has published over 40 technical papers. He was a conference co-chair of Conf. on Visual Communications and Image Processing (VCIP), 1993, and the Program Chair for the same conference in 1995. He was a co-editor of Optical Engineering special issues on Visual Communications and Image Processing in July 1991 and July 1993, and an associate editor of IEEE Transactions on Image Processing from 1992 to 1994. He is a co-editor and contributor of *Handbook of Visual Communications* to be published by Academic Press in 1995. He is a senior member of IEEE and a member of Sigma Xi.