TV 会議議事録システムに対応したメディアデータ蓄積方式

1X－1

田中 賢一郎　高木 常好　佐藤 宏明　坂内 祐一*

キヤノン情報メディア研究所

1　はじめに

我々は、遠隔地間でより効果的な協調作業を実現するため、TV 会議システムの新たな可能性について研究を重ねている．その中でも特に、「以前行なった作業内容を効果的に参照するシステム (議事録システム)」について着目し、その実現に取り組んでいる．我々の議事録システムでは、以下の要求を満たすことを目標としている．

1. 参加者の途中参加・退出があった場合においても、希望する時刻のデータを速やかに取得できること
2. データ取得に要する時間がメディアの種類に依存せず、同期が容易に実現できること

本発表では、上記の目標を実現するデータ蓄積手法を紹介する．そして、それを実現するモジュールを実装したので、実験を通して有効性について考察する．

2　従来の蓄積方法における問題点

同時刻に生成された音声データ（固定長）と映像データ（固定長）を交互にファイルに追加してゆく蓄積方法を考えてみる (図 1)．この蓄積方法では、「データ生成時刻とその格納位置」の間に、対応関係があるため、希望する時刻のデータを簡単に取得できる（例：時刻 5 のデータはファイルの 500 の位置に格納されている）．

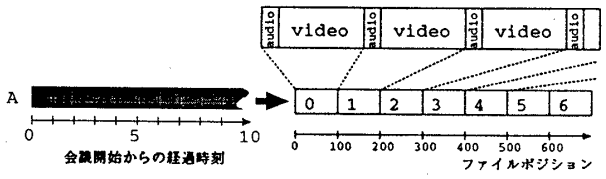

図1 従来のデータ蓄積方法

しかし、図 1 の蓄積方法では、

問題 1 － データが時間的に不連続になる

問題 2 － 可変長フレームの使用

などの問題により「データ生成時刻と格納位置」の関連がなくなった場合に、希望する時刻のデータを速やかに取得できなくなるという問題がある．

(問題 1) の場合にどのような状況が発生するか、ある参加者が途中退席した場合を考えてみる．

図 2 は、参加者 A が途中退席した時に、データがどの様に蓄積されるかを示している．A は時刻 2～5 の期間退席したため、その区間のデータが欠落している．この状態において、時刻 6 で発生した A のデータを取得しようとして上述の手法を用いると、時刻 9 で発生したデータを取得してしまう．

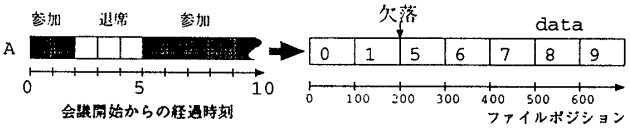

図2　データが欠落した場合

次に、(問題 2) の場合の例として、ドローツールのイベントを扱う場合を考えてみる (図 3)．この場合も、データ生成時刻と格納位置を関連付けるものが何も無いため、ある時刻にどんなイベントが起きていたか把握することは困難である．

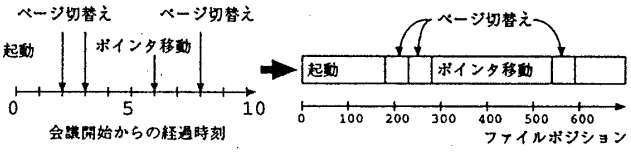

図3　可変長データを扱う場合

3　データ蓄積方法

我々の方式では、上述の問題を解決するため「データ生成時刻とその格納位置」を関連付ける index file(図 4) を導入した．

- data file — 実際にデータを格納するファイル
- index file — 「各時間で発生したデータをデータファイルのどこに格納したか」という情報を保持するファイル

*Kenichiro Tanaka Tuneyoshi Takagi Hiroaki Sato Yuuichi Bannai

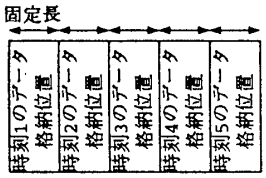

図4 index file

index file を用いてデータを格納する場合、先ずデータが data file に追加される. そして、そのデータを追加した位置を index file に格納するのだが、index file に格納される位置は次式により計算される.

$$POS = rec\_time * seg\_size * freq$$

rec_time:会議開始からの経過時間 (秒)

seg_size:index file の記録要素の大きさ

freq :一秒間におけるデータ記録回数

つまり、データ実体は単に data file に追加されてゆくだけだが、データを追加した位置は、会議開始からの経過時間に応じた場所に格納される.

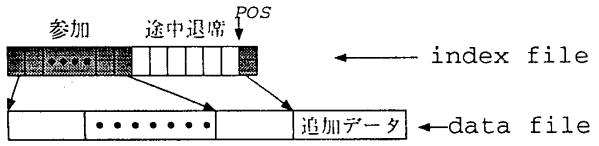

図5 途中退出が起きた時の蓄積状態

例えば、ある参加者が途中退席後、会議に復帰した状況を考えてみる (図 5). 参加者のデータはデータファ イルに追加される. そして、「何処にデータを蓄積したか」という情報を index file に格納する. 格納する場所は上式により計算されるため、途中退席していた期間はスキップされる (退席中も会議開始からの経過時間は増加している). データを取得する場合は、「再生希望時刻の情報が data file の何処に格納されているか」という情報を index file から取得する. そのため、以下の式を用いて index file のファ イルポインタを設定し、位置情報を獲得する.

$$POS = play\_time * seg\_size * freq$$

play_time:再生希望時刻

位置情報が獲得できたら、data file からデータを取得する. 正しい位置情報が獲得できない場合は、その時刻でデータが発生していない (or 記録されていない) 判断する.

データ発生頻度は各メディ ア毎に異なるが、これは、メディ ア毎に index file と datafile を用意し、freq の値を調節することで対応可能である. 例えば、映像フレームが 30/秒、音声が 20/秒で発生する場合、映像データ用の index file の freq を 30, 音声データ用の freq を 20 にすれば良い.

このように「データ生成時刻と格納位置」に関する情報を管理する index file を導入することで、はじめに述べた目標を達成できる.

1. 参加者の途中参加・退出のサポート – 図 6 に示されるよう、参加者毎に index file を用意することで、複数の参加者の途中参加・退出がある場合にも、速やかに希望する時刻のデータを取得できる.

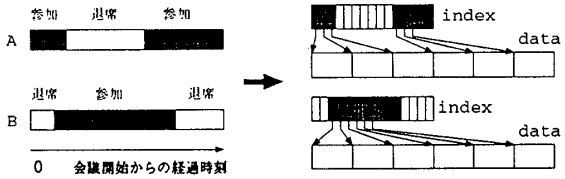

図6 複数参加者の途中参加・退出のサポート

2. 異なるメディ アの同期の実現 – 図 7 に示されるように、各々のメディ アについて、時刻 t におけるデータを、同様の手法で容易に取得できる.

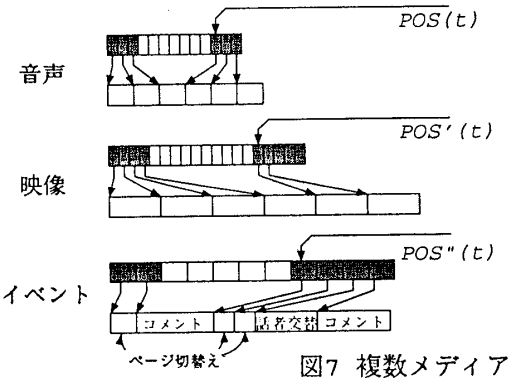

図7 複数メディアの同期

4 実装と実験結果

今回は、音声データに関するデータ蓄積・再生部分のみを実装した. 複数メディ アの同期に関しては今後の課題である. また、大きさ 640x480 の JPEG 圧縮した映像フレーム取得に要する時間を測定したところ、再生希望時刻によらず、15〜20msec であることが確認された. この値は、毎秒 50 枚以上の動画像を扱えることを意味し、議事録システムを構築する上では十分である. このとき、ハードディ スクからのデータ読み込みにかかるコストが 99%以上であり、我々の方式によるオーバーヘッドはほとんど無いことも確認した. これらのことから、本蓄積方式の性能は十分満足できる結果を得た.

5 終りに

実験結果から、上述の手法を用いると、参加者の途中参加・退出をサポートし、かつ効果的に蓄積データへのランダムアクセスができることが明らかになった. 今後は、効果的なタグの生成、タグのブラウジング等についても、取り組んで行く予定である. 最後に、実装に協力して頂いた画像メディ ア第 2,3 研究室の方々に感謝します.