卒業論文　　2000 年度 (平成 12 年度)

DV データの蓄積転送機構の実現

慶應義塾大学 環境情報学部

氏名：菅沢 延彦

指導教員

慶應義塾大学 環境情報学部

中村 修

卒業論文要旨 2000 年度 (平成 12 年度)

DV データの蓄積転送機構の実現

現在, DV は, 磁気テープ上のメディアとしてのみ利用されているため, 操作性が悪く, テープ上のデータを複数のユーザが共有できない. 本研究では, DV を映像ソースとした VoD 機構を構築することにより, DV の新しい利用環境の構築を行なった. 本機構の実現には, DV 送信に対するレート制御機能が必要である. 本研究では, DV 転送におけるレート制御の原理, 及びこの原理を応用したレート制御アルゴリズムを提案した. また, レート制御の原理に基づいた DV 送信機能を実装し, その機能の性能評価を行なった.

キーワード
1.DV, 2.VoD, 3.IEEE1394 , 4. レート制御

慶応義塾大学 環境情報学部

菅沢 延彦

Abstract of Bachelor's Thesis

Academic Year 2000

The implementation of mechanism of storing and transmitting DV data

This reseach proposed the design and implementation of a mechanism storing Digital Video data to computer HDD, and a mechanism to send DV data from storage to IP Network. Nowaday, DV data is been used only on magnetic tape devices. Therefore handling DV data is not convinient and there is no possibilities that we can share DV data on a same tape device.

Based on the design, I carried out the implementation of, not only the storing and sharing mechanism but playing the data through network, and executed a experiment running this mechanism on LAN.

Keywords :
1. DV, 2. VoD, 3. IEEE1394, 4. Rate Control

Keio University , Faculty of Environmental Information

Nobuhiko SUGASAWA

目 次

第 1 章　序論 **1**

1.1 背景 . . . 1

1.1.1 計算機環境の現状 . . . 1

1.1.2 DV/IEEE1394 の普及 . . . 1

1.2 本研究の目的 . . . 1

1.3 論文の構成 . . . 2

第 2 章　DV データ蓄積転送機構 **3**

2.1 実現するサービス . . . 3

2.1.1 想定する実行環境 . . . 3

2.1.2 アプリケーションモデル . . . 3

2.2 本機構による効用 . . . 4

第 3 章　本機構の実現における問題点及びその解決方法 **5**

3.1 DV 転送技術の現状 . . . 5

3.1.1 既存の DV 転送機構 . . . 5

3.1.2 DVTS で利用されている技術 . . . 5

3.2 本機構の実現における問題点 . . . 7

3.2.1 二次記憶装置の読み込み速度 . . . 7

3.2.2 バッファオーバーフロー . . . 8

3.3 バーストトラフィック . . . 9

3.3.1 問題点のまとめ . . . 10

3.4 問題解決の方法 . . . 10

3.4.1 DV ストリームのレート制御の原理 . . . 10

3.4.2 まとめ . . . 11

第 4 章　設計 **13**

4.1 設計概要 . . . 13

4.2 蓄積機構の設計 . . . 13

4.2.1 DV データの抽出 . . . 13

4.2.2 蓄積処理経過時間の算出 . . . 13

4.3 転送機構の設計 . . . 14

4.3.1 レート制御の精度 . . . 14

第 5 章　実装 **16**

5.1 実装概要 . . . 16

5.2 実装環境 . . . 16

5.3 蓄積機構 16
5.3.1 IEEE1394 インターフェースとの接続 16
5.3.2 蓄積時間のカウント 17
5.3.3 DV データの蓄積 17
5.4 転送機構 19
5.4.1 送信機能の実装 19

第 6 章 評価 **24**

第 7 章 結論 **26**
7.1 まとめ 26
7.2 今後の課題 26

図 目 次

2.1 全体像 4

3.1 DVTS 5
3.2 バッファリングの仕組み 6
3.3 フレームバッファ 7
3.4 二次記憶装置の読み込み速度テスト 8
3.5 バッファオーバーフロー 9
3.6 バーストトラフィックの例 9
3.7 インターバルを入れない場合の送信レートの推移 11
3.8 インターバルを入れた場合の送信レートの推移 11
3.9 インターバルをより多く入れた場合の送信レートの推移 12

4.1 DIFblock ヘッダフォーマット 14
4.2 レート制御の流れ図 15

5.1 IEEE1394 ソケットとの接続 17
5.2 蓄積処理の流れ 20
5.3 DIFblock のインターフェース 21
5.4 DV データのファイル化プログラム 22
5.5 RTP ヘッダ構造体 23

6.1 評価に用いた実験トポロジ 25
6.2 インターバルとパケットロスの関係を表すグラフ 25

表 目 次

3.1 テストに用いた計算機 7

5.1 ハードウェア環境 16
5.2 ソフトウェア環境 16

6.1 Dummynet 用計算機の仕様 24

第1章　序論

1.1　背景

1.1.1　計算機環境の現状

近年の半導体技術や通信技術の発展は, 計算機環境に対して, 大きな影響を与えている. 例として, 演算ユニット及びバスの高速化, 二次記憶装置やリムーバブルメディアの大容量化及び広帯域化, ネットワーク機器が持つ伝送速度の高速化がある. また, LAN 環境では, IEEE802.3u が広く用いられ, 各計算機は 100Mbps の帯域幅で接続されている. ネットワーク機器の性能向上は, インターネットへの接続性にも影響を与えた. 光多重や ATM, ギガビットイーサーネット等の技術によって, バックボーン・ネットワークは広帯域化している. これらにより, ネットワーク・アプリケーションの利用帯域に対する制約は軽減する傾向にある.

1.1.2　DV/IEEE1394 の普及

HD DIGITAL VCR CONFERENCE('94)[1] において, 磁気テープ上にデジタル高画質映像を記録するための VCR(Video Cassette Recorder) 規格として, DV は策定された. DV には民生用規格と, それを拡張したプロ用規格の 2 種類があり, 録画だけでなく, 編集素材としての利用も配慮した VCR 規格である.

また, DV は, シリアルバスの規格である IEEE1394 上での通信方式に関する定義も行っている. DV カメラやデッキは, このインターフェースを介して接続され, 他の IEEE1394 デバイスと DV データのデジタル転送を行える. また, IEEE1394 カードを PC に拡張することによって, DV 機器と PC 間における, DV データのやり取りが可能である. DV 機器は, DV データを外部転送する媒体として, この IEEE1394 規格を用いるため, 画質劣化を伴わない映像操作 (別媒体への保存, 複製, 編集等) ができる. DV は, 民生向きの映像フォーマットとして, 高い実用度を持った映像フォーマットであり, また VHS や 8mm をはじめとする既存の民生用 VCR 規格にかわって, 現在民間に広く普及している.

1.2　本研究の目的

現在 DV は, VCR 規格としてのステータスしか持たない. DV データへのアクセスには, DV 機器をセッティングして, さらにテープを操作する必要があり, DV はデジタル情報として利便性にかけている. 本研究の目的は, DV に関するこれらの欠点を解消するために, DV を映像ソースとした VoD 機構を構築し, DV の新しい利用環境を実現することである.

1.3 論文の構成

本論文は, 第二章において, DV データの蓄積転送機構の提案とその効用について述べる. 第三章では, 本機構を実現する上での問題点と, その解決手法について議論する. 第四章では, 本機構の設計を行なう. 第五章では, 蓄積機構, 及び転送機構の実装を行なう. 第六章では, 転送機構の送信機能に関する性能評価を行なう. 最後に第七章で, 結論を述べる.

第2章　DVデータ蓄積転送機構

2.1　実現するサービス

2.1.1　想定する実行環境

本機構は, 以下の条件を満たす実行環境を想定している.

- 通信技術に関する要求

  近年の広域分散環境, 及びLAN環境では, IPが広く用いられている. 本機構の通信基盤として, IPが適当だと考えられる. IP上のデータ転送プロトコルには, TCPとUDPがある. TCPは輻輳を検知して, 転送レートを制御することによりネットワーク親和性を高めている. また, 転送データが相手に到達することを保証するため, 再送機能を実装している. 本機構は, 時間的制約が高い連続メディアデータを転送するため, TCPのこれらの機能は不要である. したがって, 転送プロトコルには, UDPを用いる.

- スループットに関する要求

  約30MbpsのDVの転送は, 大量のパケット化処理やプロトコル処理が伴う. 本機構を構成する伝送媒体と計算機は, DV転送によって発生するトラフィックが, 十分なスループットを得られる処理能力を持っていることとする. 具体的には, 50Mbps以上の伝送媒体と, 計算機は, IntelのCPUの場合なら233Mhz以上のクロック周波数を持っていることを要求する.

  動画のデコード装置には, ソフトウェアによるものと, ハードウェアによるものがある. 本研究では, DVのデコードに要する処理を, DV機器に肩代わりさせることとする. これは, DVのデコードに要する計算を回避し, DVの再生処理のスループットを向上させるためである.

2.1.2　アプリケーションモデル

本機構は, DVの利便性, 及び共有度の向上を目標とした, ネットワークアプリケーションである. 本機能のサービスモデルは, サーバクライアントモデルである. DVデータをファイル化して蓄積, 及び送信する機能を持ったサーバと, サーバによって送信されたDVを, ストリーミング再生する機能を持ったクライアントによって構成される. 図2.1は, 本機構の全体像である.

サーバは, DV機器から受信したIEEE1394フレーム内のDVデータを抽出し, それを二次記憶装置に書き出してDVファイル化する. また, サーバは, そのDVファイルからDVストリームを生成し, クライアントへ送信する. クライアントは, サーバによって送信されたDVストリームを受信後, IEEE1394フレーム化し, DV機器へ逐次転送する. 映像は, DV機器によってデコードされて, モニターに表示される.

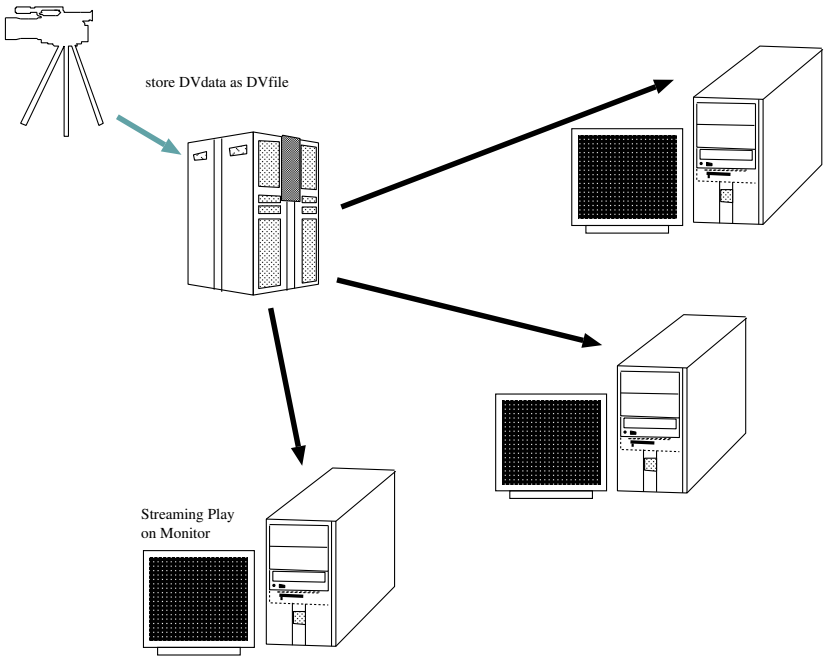

図 2.1: 全体像

2.2 本機構による効用

本機能の実現により, 新しい DV の利用環境を構築できる. 具体的には, 以下の 3 点である.

- DV データのランダムアクセス

  DV 機器は, 頭出し, 早送り, 巻き戻し等の機能によって, 任意のタイムコードまでジャンプすることができる. しかし, これらの制御には, テープの制御による大きなタイムラグが発生する. 本機構は, DV データの蓄積媒体として, 磁気テープではなく, 二次記憶装置を用いる. 二次記憶装置上のデータは, ランダムアクセスが可能である. これにより, 任意のタイムコードへジャンプする時に発生するタイムラグは, 大幅に軽減される.

- 複数地点における DV の再生

  DV 機器を DV データの蓄積媒体に用いる場合, 複数のユーザへの配信を行なうためには, ユーザ数の台数の DV 機器が必要である. 本機構は, ネットワークに接続された計算機の二次記憶装置に DV データを蓄積するため, 複数のユーザが, 一つの DV データを再生できる.

- 広域分散環境における DV の再生

  本機構は, 通信基盤として IP ネットワークを想定している. したがって, 広域分散環境における DV の遠隔利用が可能である.

第 3 章　本機構の実現における問題点及びその解決方法

3.1　DV 転送技術の現状

3.1.1　既存の DV 転送機構

既存の DV 転送機構として, DVTS(Digital Video Transport System)[2] がある. この機構は, 送信者が, DV カメラから IEEE1394 インターフェース経由で DV ストリームを取得し, 次にそれを IP データグラム化して, 受信者にリアルタイム転送するシステムである. DVTS によって, 既存のインターネット上において, DV 転送ができることが実証された.

- 1998 年, Super Computer Conference 98 において, 日本とアメリカ間の遠隔通信デモを行った.
- 1999 年, Networld+Interop Tokyo において, NOC(Network Operation Center) からの映像をマルチキャストで会場に放送 .
- 1999 年, IPv6 マルチキャストを利用し, 複数の地域で WIDE 研究会を行った.

DVTS は, 上記以外でも, 様々な場所で利用されている. 図 3.1 は, DVTS の全体像である.

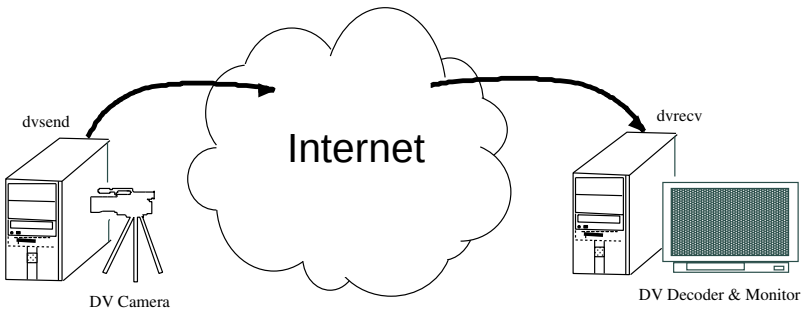

図 3.1: DVTS

3.1.2　DVTS で利用されている技術

DVTS は, インターネット上での DV 転送を想定した, いくつかの機能を実装している. 例えば, ストリーミング技術, バッファリング技術, マルチキャスト技術, 輻輳回避技術である. ここでは, 本機構と関係が深いストリーミング技術, 及びバッファリング技術について述べる.

ストリーミング技術

動画や音声といった連続メディアを, ネットワーク上で再生する手法は, 大別して, ダウンロード再生とストリーミング再生が考えられる.

ダウンロード再生は, 送信者が蓄積する連続メディアデータを, 受信者がダウンロードした後, 再生する手法である. この手法を用いた場合, 受信者は, 連続メディアデータを記憶するために, 二次記憶装置の空き領域を考慮する必要がある. 例えば, 1 時間の DV ファイルをダウンロードするためには, 約 13.5GB の空き記憶容量が必要である. また, ダウンロードに要する待機時間は, 連続メディアのファイルサイズに比例して長くなる.

ストリーミング再生は, 受信者が, 送信された DV ストリームの受信処理と並行して, 再生処理を逐次実行する技術である. この技術によって, 連続メディアデータのダウンロードに要する記憶領域や, ダウンロード待機時間に対する制限を緩和できる.

バッファリング技術

IP ネットワークでは, 到着パケット間の時間関係は保証されない. また, 到着するパケットサイズは, 途中経路の物理媒体に依存して可変長である. これらの特性を持ったネットワークにおいて, 連続メディアデータをストリーミング再生する場合, 到着したデータを, 元来送信者が保持していた状態に再構成する処理が必要である. バッファリング技術は, これらのネットワーク特性を吸収する手法である. 図 3.2 は, バッファリング技術により, 到着パケット間の時間関係を補正する仕組みを示す.

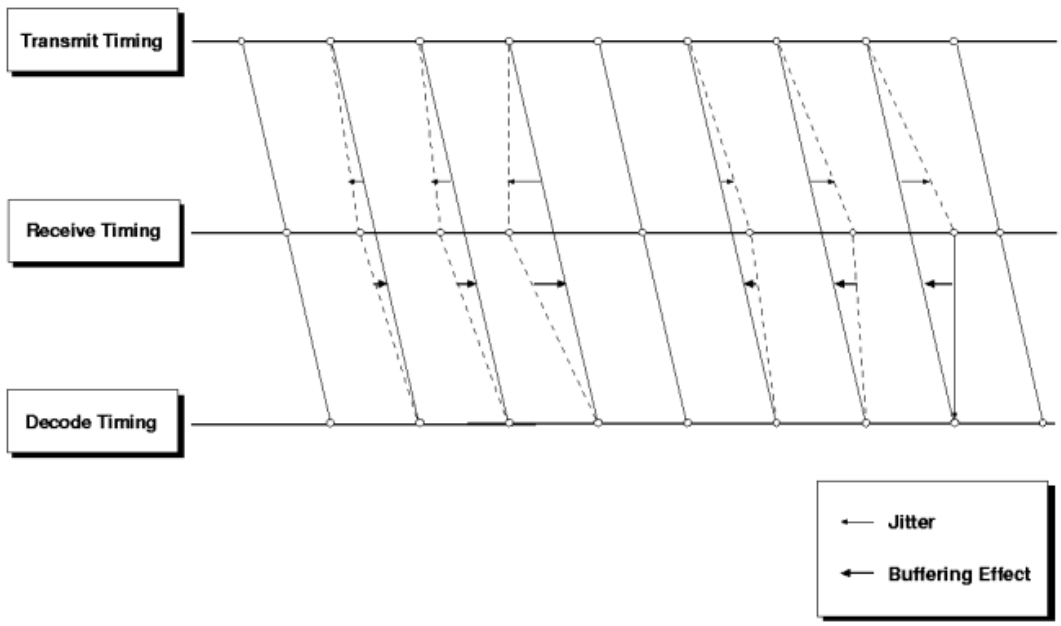

図 3.2: バッファリングの仕組み

DVTS の受信機構は, 次のようにバッファリング技術を利用している. 送信された DV は, 受信者に到着すると, 脱パケット化され, 主記憶領域に待機する. DVTS では, この領域をフレームバッファ (次ページ図 3.3) と呼んでいる. フレームバッファ内に 1 つのビデオフレーム分の DV データがバッファリングされた時, その 1 ビデオフレーム分の DV データが, IEEE1394 フレーム化され, DV 機器に転送される. 伝搬遅延のジッタや, パケットロスにより, データの不足が発生した場合, 1 つ前のフレームの

データを再利用することによって, ノイズやバッファアンダーフローを防いでいる.

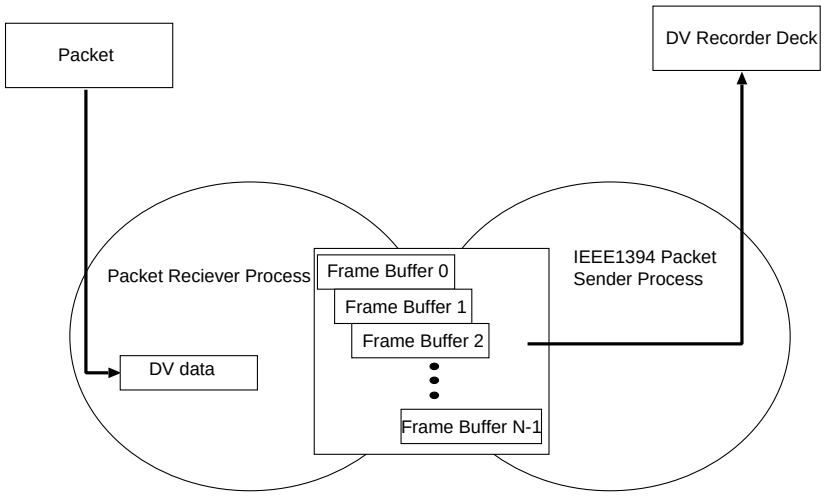

図 3.3: フレームバッファ

3.2 本機構の実現における問題点

本転送機構は, サーバの二次記憶装置上の DV データを, クライアントの DV 機器に転送する機能を持つ. しかし, それらの実現には, 2 つの問題がある. 以降では, それら問題と解決手法について述べる.

3.2.1 二次記憶装置の読み込み速度

本転送機構のサーバとして適当な性能の計算機上で, 二次記憶装置の読み込み速度に関する簡単なテストを行った. テストの目的は, サーバにおける二次記憶装置の読み込み速度の検証である.

表 3.1: テストに用いた計算機

CPU	PentiumIII 450Mhz
主記憶	SDRAM 128MB
二次記憶装置 (転送モード)	IBM-DTLA-307020(UDMA33)
IEEE1394 インターフェース	Texas Instrument PCILynx
OS	FreeBSD 4.1.1[5]

これは, 表 3.1 の計算機上で, 二次記憶装置の読み込み速度テストを行った結果である. テストでは, `dd` コマンドを用いて, *test* を/dev/null に書き出した. *test* は, テスト前にあらかじめ用意した約 1GB のファイルである. テスト結果では, この二次記憶装置に対して, 1024KB 単位で 1024 回の読み込みを実行した時, 約 240Mbps の読み込み速度が出たことを示す. これは, DV のデコードレートの 8 倍にあたる.

本機構の全ての構成要素は, オペレーティングシステムのユーザランドに構築される. また, 今回テストに用いた `dd` コマンドも, ユーザーランドのソフトウェアである. したがって, このテスト結果は, 転送機構が二次記憶装置からデータを読み込む速度の指標として適当であると考えられる.

```
% dd if=./test of=/dev/null bs=1024k count=1024
024+0 records in
1024+0 records out
1073741824 bytes transferred in 35.309488 secs (30409442 bytes/sec)
0.01s user 10.42s system 29% cpu 35.320 total
```

図 3.4: 二次記憶装置の読み込み速度テスト

3.2.2 バッファオーバーフロー

DVTS は, DV データの入力源, 及び DV のデコード装置の両方が DV 機器であるため, データ入出力の同期がとれている. 一方, 本機構では, DV の入力源が二次記憶装置であり, デコード装置は DV 機器であるため, データ入出力が非同期である. DV の送信レートが, DV のデコードレートを上回る場合, 超過して送信された分のデータは, クライアント機構内のいずれかの記憶領域に待避させる必要がある. ただし, 転送経過時間に比例して, 記憶領域を消費するため, いつか空き記憶領域は枯渇する. この状況下のクライアントは, 到着パケットを破棄し, パケットロスを生じさせる. この状態をバッファオーバーフローという. パケットロスは, 再生する映像にノイズを発生させる要因である.

サーバの送信レートを S, クライアントが DV をバッファリングできる最大容量を L, 転送経過時間を T, DV のデコードレートを D と定義した場合, 以下の式が成り立つ時, バッファオーバーフローが発生する.

$$S \times T - D \times T > L$$

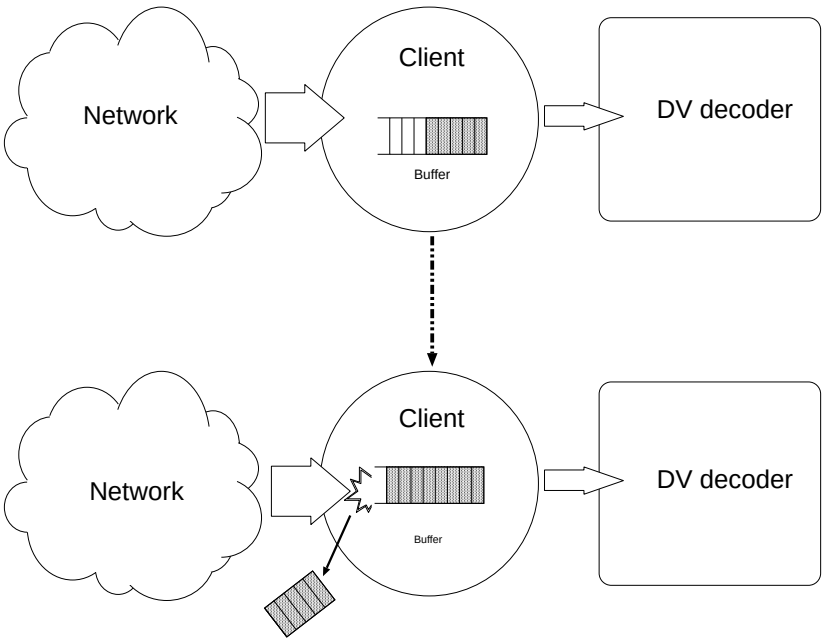

図 3.5: バッファオーバーフロー

3.3 バーストトラフィック

二次記憶装置上のデータを転送する場合, その装置の読み込み速度範囲内のレートで, データを読み込めるため, サーバは任意の送信スケジューリングを行なえる. これにより, 本機構のサーバは, バーストトラフィックを生成する可能性がある. バーストトラフィックとは, データの送信レートが, 大きく変動するトラフィックである. バーストトラフィックによって発生する問題は, 転送に必要な帯域幅の拡大や, データの伝送経路上における, 他のトラフィックとの帯域競合がある.

バーストトラフィックの発生例として, 転送する動画データの量子化スケールに起因して発生するものがある. 図 3.6 は, 総量, 内容, フォーマットが同一の動画データを, 異なる方式によって転送した時, 生成されたトラフィックのグラフである.

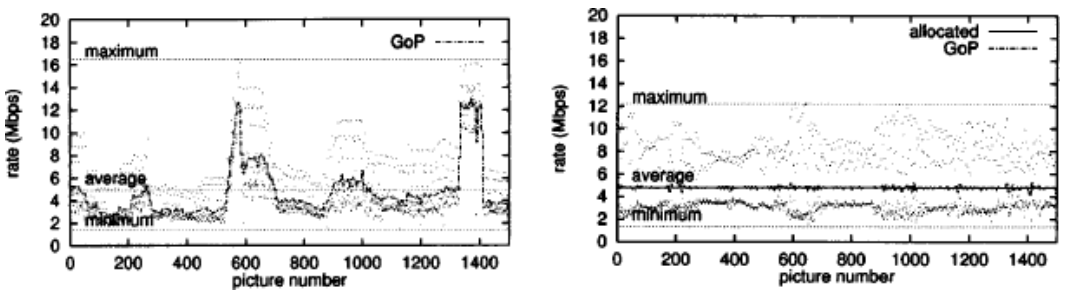

図 3.6: バーストトラフィックの例

図 3.6 中左表のトラフィックは, バーストトラフィックである. 図 3.6 中右表のトラフィックは, フレーム毎の量子化スケールを平均化することにより, バースト性の解消を行なった例である. この図において

注目すべき点は, 割り当て帯域幅の利用効率の相違である. 左表のトラフィックは, 右表のトラフィックより広い帯域を必要とすることが判る.

3.3.1 問題点のまとめ

本転送機構の実現には, サーバの二次記憶装置の読み込み速度が, クライアントの DV のデコードレートを上回るため, クライアントのバッファオーバーフローが発生する問題がある. この問題は, サーバの二次記憶装置と, クライアントの DV デコード装置の間に, データ入出力を同期する仕組みが無いために発生する. また, 二次記憶装置の読み込み速度が可変レートであるため, サーバはバーストトラフィックを生成する問題がある. IP ネットワークにおいて, バーストトラフィックは, 転送に必要な帯域幅の拡大や, 他のトラフィックとの帯域競合を起こす要因となる.

3.4 問題解決の方法

本機構の実現には, クライアントのバッファオーバーフローを回避する仕組み, 及びバーストトラフィックを解消する仕組みが必要である. IP ネットワークでは, 到着パケット間の時間関係は保証されない. したがって, 送信者と受信者間における, データ入出力の同期機構を実現することは困難である. また, IP ネットワークはパケット交換式の回線共有型ネットワークであり, 帯域幅の制約や, 帯域の利用効率に関して考慮すべきである. 本研究では, それらの問題の解決手法として, 独自のレート制御アルゴリズムを考案した. 次節では, 本機構で用いるレート制御の原理について議論する.

3.4.1 DV ストリームのレート制御の原理

レート制御に用いるパラメータには, 2 種類ある. 転送前に明らかな情報と, 転送時に検知する情報である. 前者の情報は, 転送する動画のデコードレート, 受信者のバッファサイズ, 接続帯域幅等がある. 後者の情報は, パケットロス発生率, 伝搬遅延, 伝搬遅延の揺らぎ等がある. 本研究は, まず DV の蓄積, 転送機構の実現に主眼をおくため, 転送する動画のデコードレートに焦点を当てることとする.

DV のフレームは, 同一の量子化スケールで圧縮されており, データをデコードするレートは一定である. これは, DV をネットワーク転送する時, 単純なバイト列として扱えることを示す. 別の動画フォーマットである MPEG は, フレーム毎に異なった量子化スケールを持つ. MPEG による動画転送は, 転送データ量の圧縮に効果があるが, フレーム毎が持っている情報量が変動するため, バーストトラフィックを起こし易い.

バイト列の転送は, 送信処理中の任意のタイミングにインターバルを挿入することにより実現できる. 図 3.7 は, インターバルを入れない場合の送信レートの推移と, デコードレートの推移を図式化したものである. 各グラフの面積は, 送信する DV ファイルのサイズで等しい. グラフの縦軸が送信データ量, 横軸がそのファイルの送信に要する時間である. 図 3.8 は, 送信インターバルを入れた場合の送信レートの推移を示す. 図 3.9 は, 送信インターバルをより多く入れた場合の送信レートの推移を示す.

図 3.7 は, クライアントがバッファオーバーフローを起こすが, 図 3.8 は起こさないことを示している. また, 図 3.8 は, 図 3.9 より広い帯域を必要とすることを示している. これらにより, 送信インターバルを増やし, 送信データを細分化することにより, クライアントのバッファ領域や帯域幅をより効果的に利用できると考えられる.

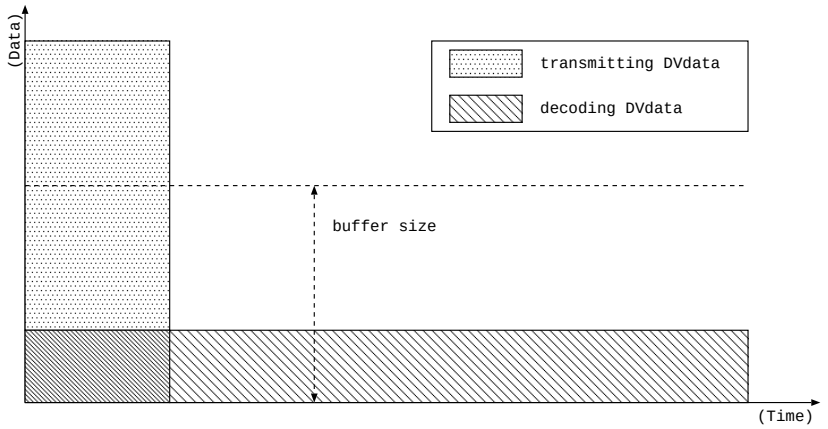

図 3.7: インターバルを入れない場合の送信レートの推移

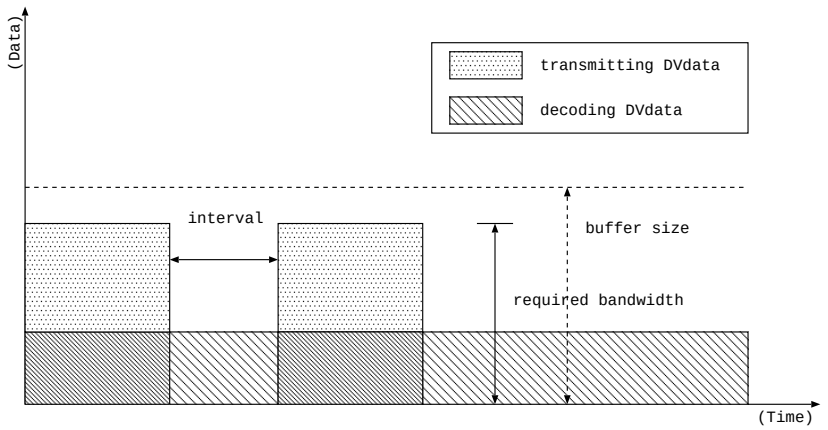

図 3.8: インターバルを入れた場合の送信レートの推移

3.4.2 まとめ

本機構の実現には, 2 つの問題がある. 1 つ目は, 二次記憶装置の読み込み速度が, DV のデコードレートを上回るため, クライアントのバッファオーバーフローが発生する問題である. この問題は, サーバの二次記憶装置と, クライアントの DV デコード装置の間に, データ入出力の同期機構が無いためである. 2 つ目の問題は, バーストトラフィックによる利用帯域幅の拡大や, 他のトラフィックとの帯域競合を起こす問題である. これらの問題の解決手法として, 送信するデータのレート制御を考案した. DV ストリームは, 単純なバイト列であるから, 送信処理中の任意のタイミングにインターバルを挿入し, 送信データを細分化することによるレート制御ができる. インターバルを挿入する回数を増やすことにより, より細かく送信データを細分化できる. この原理を用いて, クライアントのバッファオーバーフローの回避や, バーストトラフィックの解消を実現できる.

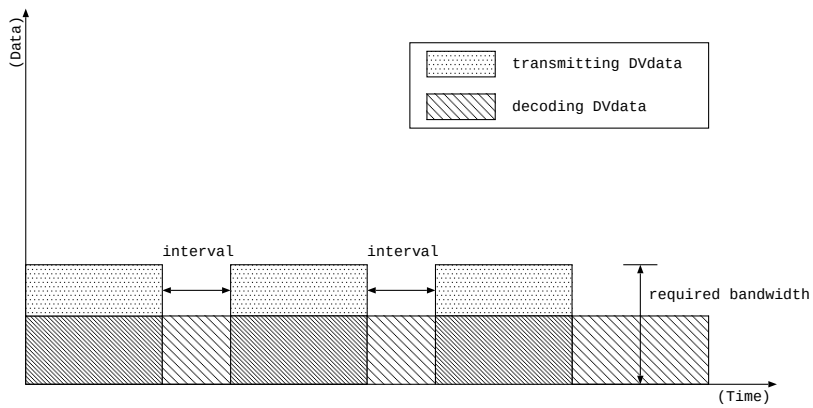

図 3.9: インターバルをより多く入れた場合の送信レートの推移

第4章　設計

4.1　設計概要

本機構は, 大きく分けて 2 つの要素から成る.

- 蓄積機構
  DV 機器からの出力を, IEEE1394 インターフェースで受信し, 受信した IEEE1394 フレームから DV データを抽出して, ファイル化する機構である.

- 転送機構
  転送機構は, 送信機能と受信機能から成る. 送信機能は, 二次記憶装置上の DV ファイルから, データを読み込み, それをネットワークへ送信する機能である. 受信機能は, DV を受信しながら, 逐次再生する機能である. ただし, 送信機構の実現には, レート制御機能が必要である.

まず, DV の仕様に基づいた蓄積機構の設計を行ない, その後, 本機構を実現する上で, 最も重要と考えられるレート制御機能について説明する.

4.2　蓄積機構の設計

4.2.1　DV データの抽出

IEEE1394 バス上の DV データは, 488 バイト単位で構成されている. 先頭の 8 バイトには, 送信元識別子, データフォーマット識別番号, NTSC または PAL の方式情報等が含まれる. DV データは, その 8 バイトを除いた残りの 480 バイトである. DV ファイルは, DV パケット内の DV データ部のみを抽出し, 単純に連結したものである.

IEEE1394 バス上を転送される DV パケット列には, 定期的に空 DV パケットが挿入されている. 空 DV パケットは, DV データを含まない. DV データを二次記憶装置に蓄積するにあたって, 空 DV パケットは不必要なので, 検出後, 無視する.

4.2.2　蓄積処理経過時間の算出

DV を構成する最小単位は DIF(Digital Interface Format)block と呼ばれる. 各 DIFblock は, DIFblock ヘッダを持ち, その中にビデオフレームのシーケンス番号を示す値を持っている. 図 4.1 は, DIFblock ヘッダフォーマットである. DV は毎秒 29.97 フレームであるため, この値の累計値を利用することにより, 蓄積処理の経過時間を算出できる. この値を SEQ という.

SEQ は, DIFblock の先頭 5 ビット目から 8 ビット目までのフィールドに記録されている. SEQ は, 0 ~ 11 もしくは 15 という値をとる. 15 は, DV の音声または映像以外のデータを含む DIFblock につけられる番号であり, ビデオフレームのシーケンス番号として用いない.

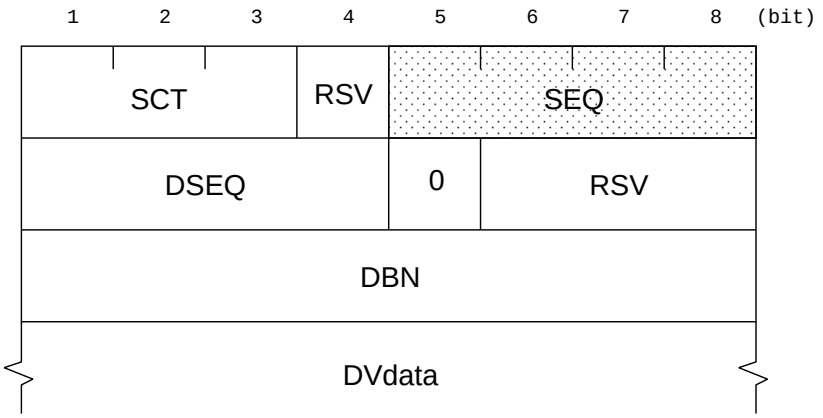

図 4.1: DIFblock ヘッダフォーマット

4.3 転送機構の設計

レート制御機能の仕組み

レート制御の目的は, クライアントのバッファオーバーフローの回避, 及びバーストトラフィックの解消である. 本レート制御機能は, 以下の 4 つの工程から成る.

1. 送信を完了したデータ量と, そのデータ量に対する再生時間
2. 再生時間の累積値が, ある設定値に達した時, その累積再生時間分のインターバルを発生
3. インターバル時に発生した誤差時間分のデータ量を算出
4. 次回の送信データに, その誤差時間分のデータを補充

本送信機能における DV 転送は, データ送信とインターバルの繰返しにより実行される. インターバルは, 送信を完了したデータ量から算出される再生時間の合計値に基づく. あらかじめインターバルを設定し, 再生時間の合計値が, そのインターバルの設定値に達した時, インターバルが実行される. ただし, 本転送機構の実行環境であるオペレーティングシステムでは, そのインターバルに対する時間的保証は無い. したがって, インターバルには誤差が生じることを考慮すべきである. インターバルは, 最低待機時間を保証するため, インターバルに超過時間が発生すると考えられる. これの解決のため, インターバル時に発生した誤差時間分のデータ量を算出し, 次に実行される送信に補充する.

図 4.2 は, 本レート制御の流れ図である. あらかじめ設定したインターバルを $I(\mathrm{sec})$, 送信したデータ量を $n(\mathrm{bit})$, 送信完了したデータ量の合計値を $t(\mathrm{bit})$, DV のデコードレートを $D(\mathrm{bps})$, 実際のインターバルを $s(\mathrm{sec})$, インターバル時に発生した誤差時間分のデータ量を $e(\mathrm{bit})$ とする.

4.3.1 レート制御の精度

本レート制御機能の目的は, クライアントのバッファオーバーフローの回避, 及びバーストトラフィックの解消である. 本レート制御機能に求められる精度は, これらの目的に基づく. したがって, 本レート制御機能の精度は, バッファオーバーフロー, 及びバッファーアンダーフローが起きない程度とする. バッファオーバーフローの発生条件は, ある期間に送信したデータ総量が, その期間にデコードされるデータ総量を, クライアントのバッファサイズより多く上回ることである. 一方, バッファアンダーフローの発

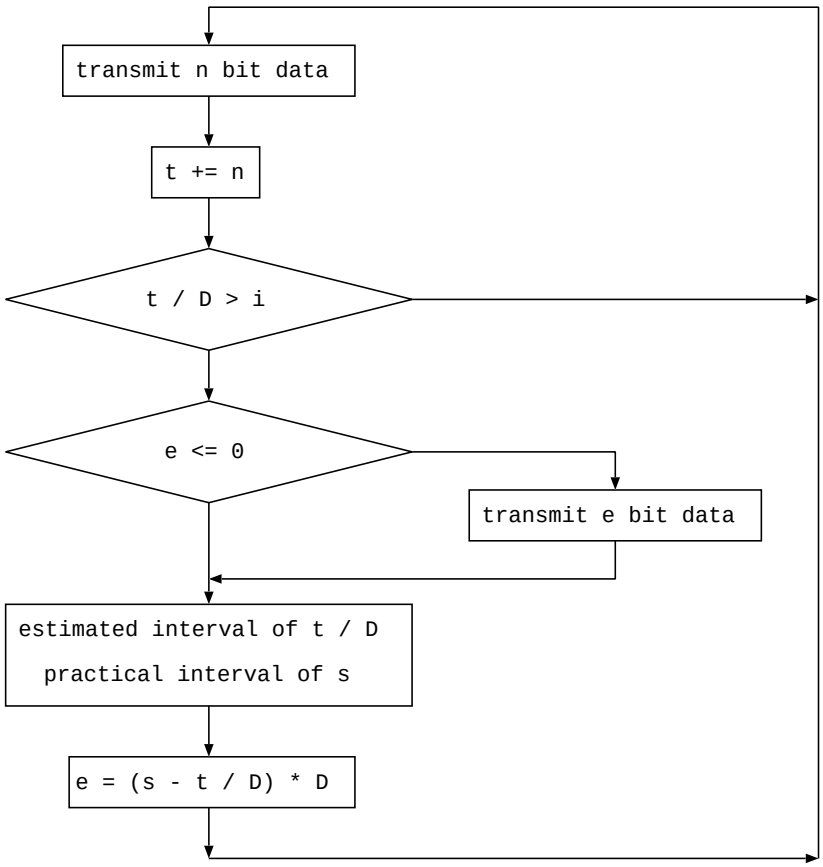

図 4.2: レート制御の流れ図

生条件は, ある期間に送信したデータ総量が, その期間にデコードされるデータ総量を下回ることである. DV は 1 秒当たり 29.97 フレーム, 1 フレーム 120KB である. また, DV のデコードは仕様上, 1 フレーム単位で行なう必要があるため, クライアントは IEEE1394 インターフェースに 1 フレームずつ DV データを出力する. したがって, クライアントは, 1/29.97 秒当たり 120KB のデータのバッファリングを最低限必要とする.

本レート制御の精度の決定因子は, バースト性の許容量である. 高いバースト性を許容できる伝送経路を使う場合, 高い精度のレート制御は必要ない. 一方, バースト性をほとんど許容できない伝送経路を使う場合, 高い精度のレート制御が必要である. レート制御の精度は, インターバルの数と長さに基づいて調整する. レート制御の精度の上限は, 実際はオペレーティングシステムのクロックレートに収束する. これは, レート制御の精度を上げるために, インターバルの数を増やし, 各インターバルを短かくした場合, インターバルは, オペレーティングシステムのクロックレートで限界に達するためである.

第5章　実装

5.1　実装概要

本研究では, 蓄積機構, 及び転送機構の送信機能を実装した. 蓄積機構は, IEEE1394 バスから IEEE1394 フレームを受信して, そこから抽出した DV データを二次記憶装置上に書き出し, 最終的に DV ファイルを作成する機構である. 転送機構の送信機能は, DV ファイルから DV データを抽出, それを RTP カプセル化し, DV ストリームを生成する機能である. クライアントは, DVTS の dvrecv を利用した.

5.2　実装環境

実装は, 下記の環境下で行った.

表 5.1: ハードウェア環境

CPU	PentiumIII 450Mhz
主記憶	SDRAM 128MB
二次記憶装置 (転送モード)	IBM-DTLA-307020(UDMA33)
IEEE1394 インターフェース	Texas Instrument PCILynx
OS	FreeBSD 4.2-RELEASE

表 5.2: ソフトウェア環境

OS	FreeBSD4.2-RELEASE
IEEE1394 パッチ	firewire-freebsd-4.0-20000710
DVTS	DVTS-0.3.5
プログラミング言語	C 言語
コンパイラ	gcc 2.95.2

5.3　蓄積機構

5.3.1　IEEE1394 インターフェースとの接続

サーバは, IEEE1394 インターフェースから DV データを受信する. IEEE1394 インターフェースは, ソケット抽象化されている. IEEE1394 インターフェースとの接続にはソケット API を利用する.

```
 #define  IEEE1394IF "lynx0"                          /* IEEE1394 インターフェース
名 */

 int ieee1394soc;                                      /* IEEE1394 ソケット */
 struct sockaddr_n1394 ieee1394soc_addr;               /* IEEE1394  ソケット構造
体 */

 ieee1394soc = socket(AF_N1394, SOCK_DGRAM, 1);
 memset((char *)&ieee1394soc_addr, '\0', sizeof(ieee1394soc_addr));
 ieee1394soc_addr.sn1394_family = AF_N1394;
 ieee1394soc_addr.sn1394_fch = 0x3f;
 ieee1394soc_addr.sn1394_ftag = 0x1;
 memcpy(ieee1394soc_addr.sn1394_if, IEEE1394IF, IFNAMSIZ);

 connect(ieee1394soc, (struct sockaddr *)&ieee1394soc_addr,
                                                  sizeof(ieee1394soc_addr));
```

図 5.1: IEEE1394 ソケットとの接続

5.3.2 蓄積時間のカウント

DV は固定長のデータフォーマットなので, 蓄積したデータの総量を用いた経過時間の算出が可能である. しかし, 蓄積するデータのバイト数を, `unsigned int` 型変数を利用してカウントする場合, 約 4GB を蓄積したとき, `unsinged int` 型変数がオーバーフローしてしまう. これは, `unsigned int` 型変数が 2 の 32 乗個の整数値しか持たないためである. この問題の解決のため, 本蓄積機構では, DIFblock ヘッダ中の SEQ の値を利用して, 録画時間をカウントする.

SEQ は, ビデオフレームをカウントする値である. 蓄積時間を T 秒, その T に換算される SEQ の合計値を S とする. 尚, 式最後の $+0.5$ は, 四捨五入の演算である. DV は 29.97fps のフレームレートであるから, 蓄積時間に対応した S は以下の式により算出できる.

$$S = 29.97 \times T + 0.5$$

5.3.3 DV データの蓄積

接続したソケットから受信する DV パケットは 488 バイトである. 先頭 8 バイトはヘッダであり, 本蓄積機構は利用しない. 抽出すべきデータは, その中の 9 バイト目以降の 480 バイトである. ただし, 定期的に空 DV パケットが受信されることを考慮すべきである. また, 蓄積時間をカウントするために, DIFblock ヘッダ内の SEQ を利用するため, DV パケット内の DIFblock ヘッダを覗き見る必要がある. ただし, SEQ の値が 15 の時は, DIFblock 内のデータが, 動画や音声ではないため, これをカウントしないことを考慮すべきである. DV パケットは, 6 個の DIFblock が連結されており, それらは全部同じ SEQ を持っている. したがって, 先頭の DIFblock の SEQ のみを覗き見ればよい. 図 5.3 は, DIFblock ヘッダを格納するための構造体 `difblock_hdr`, 及び, DV パケットの中を参照し, DIFblock ヘッダの情報を,

構造体 `difblock_hdr` に格納する関数 `get_difblock_hdr` の宣言である. 図 5.2 が, 蓄積処理全体のフローチャートであり, 図 5.4 は, DV のファイル化プログラム本体である.

5.4 転送機構

本転送機構の DV 転送におけるレート制御の原理は, 送信処理中の任意のタイミングにインターバルを挿入し, 送信データを細分化することである. この原理の検証のため, 送信機能の実装を行なった.

5.4.1 送信機能の実装

本転送機構では, DV をパケット化する際, RTP を利用する. RTP(Real Time Protocol)[3] は, 連続メディアデータをパケット化する時, そのパケットに付加する情報群の枠組みを規定したプロトコルである. [3] では, ペイロード [4] が DV であるパケットに対する付加情報を, RTP に基づいて規定している. 本転送機構では, 他の DV 転送機構の実装との相互接続性を保たせるために, この標準を利用する. 図 5.5 は, RTP ヘッダ構造体である.

実装した送信機能は, 以下の 6 つの工程から成る.

1. DV データの読み込み
   DV データを読み込む単位は, 480 バイトで行った. これは, IEEE1394 インターフェースから受信する DV パケット内のデータ含有量と等しい. DV は, 480 バイト単位で構造化されており, この読み込み方法が合理的である.

2. SEQ の抽出
   480 バイト単位で読み込んだ DV データは, SEQ が同じである. したがって, 先頭の DIFblock についてのみ, SEQ を覗き見すればよい.

3. SEQ 値に基づいた RTP タイムスタンプの決定
   SEQ の値が換わった時, RTP タイムスタンプを増加させる必要がある. RTP タイムスタンプの増加値は, NTSC 形式の場合 3003 である [4].

4. DV データの RTP カプセル化
   1 つのパケットには, 任意の量のデータを載せてよい. ただし, フレームが異なるデータを, 1 つのパケット内に共存させてはならない. フレームが異なるデータは, RTP タイムスタンプを増加させ, 別のパケットを生成する.

5. 送信
   送信は, `sendto` システムコールによって行う. 出力バッファは, `setsockopt` システムコールにより拡大させる. これは, 出力バッファのオーバーフローを防ぐためである.

6. インターバル
   インターバルは, インターバルの発生頻度に基づいて実行される. これにより, 定期的なインターバルを生成できる. 前の 5 つの工程を繰返し, インターバルを取るべきタイミングになった時, そのインターバルを実行する. インターバルは, FreeBSD のカーネルコンフィグレーションにおける `options HZ` を利用し, 1msec の時間粒度で指定できるようにした.

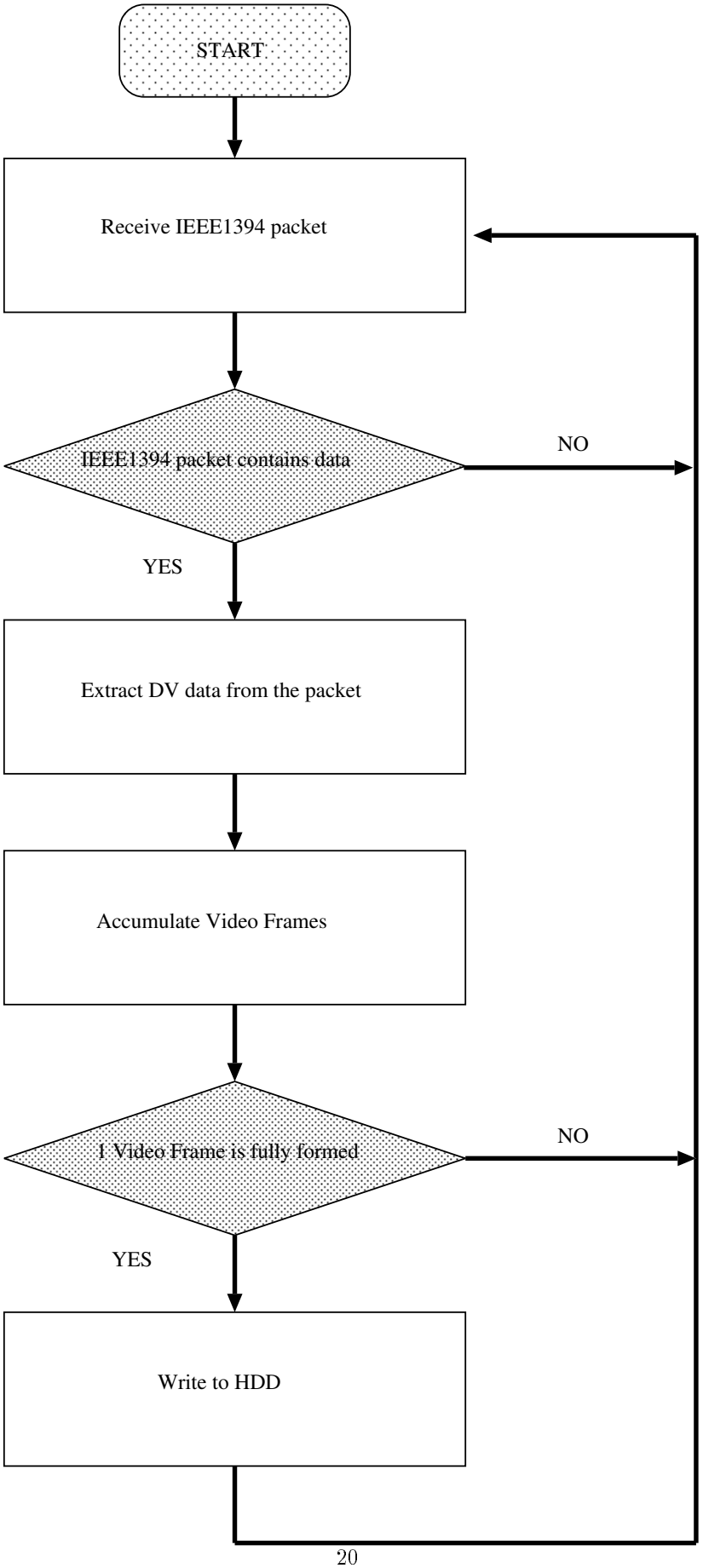
START
Receive IEEE1394 packet
IEEE1394 packet contains data
NO
YES
Extract DV data from the packet
Accumulate Video Frames
1 Video Frame is fully formed
NO
YES
Write to HDD

図 5.2: 蓄積処理の流れ

```
struct difblock_hdr{
        unsigned char sct; unsigned char seq; /* 本機構で用いる変数 */
        unsigned char dseq; unsigned char dbn;
};

void
get_difblock_hdr(struct difblock_hdr *dblkhdr, const unsigned char *dvpkt){
        dvpkt->sct  = (ntohl(*dblk) >> 29) & 0x7;
        dvpkt->seq  = (ntohl(*dblk) >> 24) & 0xf;
        dvpkt->dseq = (ntohl(*dblk) >> 20) & 0xf;
        dvpkt->dbn  = (ntohl(*dblk) >> 8)  & 0xff;
}
```

図 5.3: DIFblock のインターフェース

```
#define VIDEOFRAME_SZ           12000      /* ビデオフレームのサイズ*/
#define DVPAKCET_SZ             488        /* DV パケットのサイズ */
#define IEEE1394FRAME_SZ        512        /* IEEE1394 フレームのサイズ */
#define CIPHEADER_OFFSET        8          /* ヘッダのサイズ */

int prev_seq;                              /* 一つ前の SEQ */
int seq_cnt;                               /* SEQ のカウンタ */
int nbyte;                                 /* 受信した DV パケットのバイト数 */
struct difblock_hdr *dblock_hdr;           /* DIFblock 構造体へのポインタ */
int dvpkt_num;                             /* DV パケット数のカウンタ */
unsigned char recv_buf[IEEE1394FRAME_SZ];  /* IEEE1394 フレームの受信バッファ */
int fd;                                    /* DV ファイルのファイルディスクリプタ */
unsigned char out_buf[VIDEOFRAME_SZ];      /* ビデオフレームの書き込みバッファ */

(中略)

for( prev_seq = 0, seq_cnt = 0; ; ){
        for( dvpkt_num = 0;
             dvpkt_num < VIDEOFRAME_SZ / (DVPACKET_SZ - CIPHEADER_OFFSET); ){
                nbyte = recv(ieee1394soc, recv_buf, IEEE1394FRAME_SZ, 0);
                if( nbyte == CIPHEADER_OFFSET )        continue;

                get_difblock_hdr(dblock_hdr, &recv_buf[CIPHEADER_OFFSET]);

                if( (dblock_hdr->seq != 15) && (dblock_hdr->seq != prev_seq) ){
                        seq_cnt++;
                        prev_seq = dblock_hdr->seq;
                }

                memcpy(&out_buf[VIDEOFRAME_SZ * dvpkt_num],
                                &recv_buf[CIPHEADER_OFFSET], DVPACKET_SZ);

                dvpkt_num++;
        }
        write(fd, out_buf, VIDEOFRAME_SZ);
}
```

図 5.4: DV データのファイル化プログラム

```
/*
 * RTP data header
 */
typedef struct {
    unsigned int version:2;   /* protocol version */
    unsigned int p:1;         /* padding flag */
    unsigned int x:1;         /* header extension flag */
    unsigned int cc:4;        /* CSRC count */
    unsigned int m:1;         /* marker bit */
    unsigned int pt:7;        /* payload type */
    unsigned int seq:16;      /* sequence number */
    u_int32 ts;               /* timestamp */
    u_int32 ssrc;             /* synchronization source */
 /* u_int32 csrc[1]; */       /* optional CSRC list */
} rtp_hdr_t;
```

図 5.5: RTP ヘッダ構造体

第6章　評価

バッファオーバーフロー回避に対する評価

クライアントのバッファオーバーフローの問題に対する評価は, 送信処理中にインターバルをいれない転送と, 適当なインターバルをいれる転送を比較して行った. 評価には, dvrecv の-n を用い, パケットロスをカウントした. 結果は, インターバルをいれない転送は, 常に 12 万パケットがロスした. 一方, インターバルをいれた転送では, 任意の回数のインターバルを挿入することにより, バッファオーバーフローによるパケットロスを回避できた.

バースト性の解消に対する評価

本実装によるバースト性の解消度を明らかにするため, 帯域の利用効率に焦点を当てて性能評価を行なった. 評価実験に用いたノードは, サーバ, クライアント, 及び仮想的な帯域制限を設けるための中間ノードである. 中間ノードには, 帯域制御アプリケーションの Dummynet[7] を用いた. Dummynet 用計算機の仕様を, 表 6.1 に示す.

表 6.1: Dummynet 用計算機の仕様

CPU	PentiumIII 450Mhz
主記憶	SDRAM 128MB
ネットワークインターフェース (output)	Intel Pro 10/100B/100+ Ethernet
ネットワークインターフェース (input)	VIA VT3043 Rhine I 10/100BaseTX
OS	FreeBSD 4.2-RELEASE

Dummynet により, 仮想的な帯域制限を設け, 本転送機構により生成される DV ストリームの帯域利用効率に関する評価を行った. 評価を行なった実験トポロジは, 図 6.1 である. 実験は, インターバルの長さ, 及びインターバルの回数の値のペアを変化させて, DV 転送を行った. 送信レートは, DV のデコードレートで一定である.

表 6.2 が, 実験の結果である. 表 6.2 は, 本サーバによる送信のインターバルを 10msec から 60msec まで, 5msec きざみに増加させた時のパケットロス増加量を示している. パケットロスの測定には, dvrecv の-n オプションを用いた.

図 6.2 は, インターバル 45msec 以上において, パケットロスが多く発生することを示している. これは, インターバルが長くなることにより, 生成されるトラフィックのバースト性が高まり, 帯域の上限を超えるためだと考えられる. 本論では, 送信インターバルを増やし, 送信データの細分化を図ることにより, クライアントのバッファ領域や帯域幅をより効果的に利用できることを, 第 3 章 4 節で述べた. これらの評価により, レート制御の原理は検証された. また, 本転送機構は, バーストトラフィックの問題を解決した.

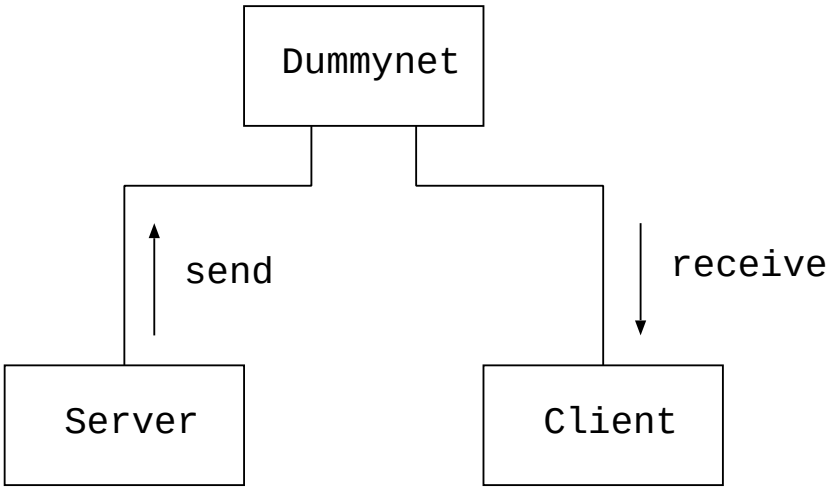

図 6.1: 評価に用いた実験トポロジ

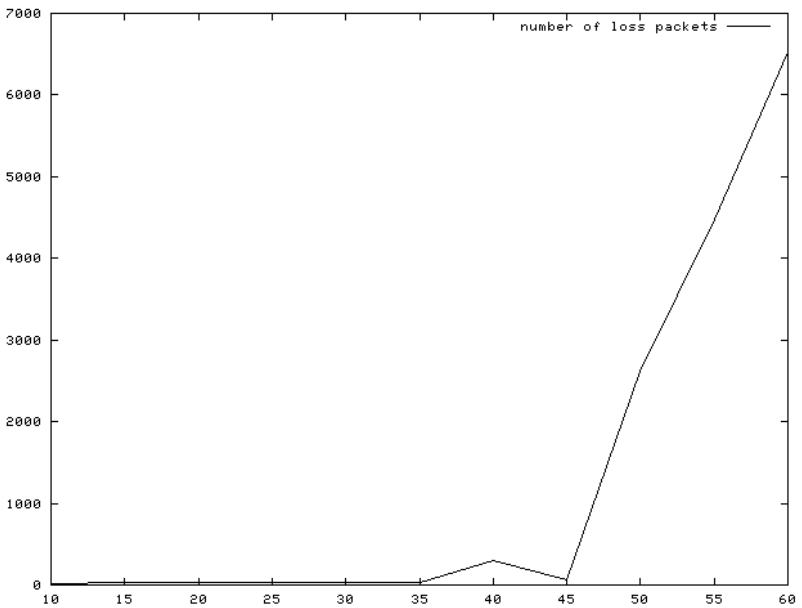

図 6.2: インターバルとパケットロスの関係を表すグラフ

第7章　結論

7.1　まとめ

現在 DV は, 民生の VCR 規格として広く用いられているが, DV の記憶媒体としてテープを用いるため, データの操作性や共有度に欠る. DV 機器を用いた場合, ある一つの映像ソースは 1 ユーザからしか参照できないことや, データに対するランダムアクセスができないという問題がある.

近年の半導体技術や通信技術の発展により, CPU の高速化や記憶媒体の大容量化, ネットワークの広帯域化が進み, 計算機環境は高性能化している. DV データは, 比較的大きな情報を持ったデータだが, 現在の計算機環境を活用することにより, DV の新しい利用環境を構築できる.

本研究は, DV データを計算機上の二次記憶装置に蓄積する機構, 及び IP ネットワーク上に転送する機構を構築することにより, DV の利便性や共有度の向上を狙った. ただし, 二次記憶装置上の DV データをストリーミング再生する際, 二次記憶装置からのデータ読み込みと, DV のデコード装置間に同期機構がないためクライアントのバッファオーバーフローを招く可能性がある. また, 二次記憶装置の読み込み速度は可変レートであるため, バーストトラフィックを生成する可能性がある. 本研究では, 送信インターバルに基づいたレート制御機能を用い, これらの問題を解決した. 本研究は, DV の新しい利用環境を実現した.

7.2　今後の課題

インターネットにおける連続メディアデータ転送は, 伝搬遅延の揺らぎや, 輻輳状態などが要因となって, 様々な問題が起きる. 本研究では, これらに対する問題解決を行っていない. 今後の課題は, 本機構がインターネット上で利用できるように改良を加えることである. また今回, DV を二次記憶装置上に蓄積することによって, DV の利便性向上を図ったが, 現時点では再生しかできない. 映像を制御するためのユーザインターフェースを実装することも考えている.

謝辞

本研究を進めるにあたり, 御指導をいただきました, 慶應義塾大学環境情報学部教授の徳田英幸博士と村井純博士に感謝いたします. また, 絶えず御助言と御指導をいただきました, 同学部の中村修博士と楠本博之博士に深い感謝の意を表します.

杉浦一徳氏, 重近範之氏, 土本康生氏をはじめとする慶應義塾大学政策メディア研究科後期博士課程の方々, 並びに同大学環境情報学部の徳田・村井・中村・楠本研究室の諸氏には, 本研究を進めていく上で様々な議論をして頂き, また耐えざる励ましとご援助を頂きました. ここに, 深い感謝の念を表します.

最後に, 慶応義塾大学湘南藤沢キャンパス オミクロン館 308 号室に属していた方々には, 研究開始当初から数々のご助言と励ましを頂きました. 深い感謝の念を表します.

以上を持って謝辞といたします.

2001.1.31

菅沢 延彦

参考文献

[1] HD Digital VCR Conference: "Specifications Consumer-Use Digital VCR's using 6.3mm magnetic tape", HD Digital VCR Conference,1994

[2] DVTS http://www.sfc.wide.ad.jp/DVTS/

[3] H.Schulzrinne S.Casner R.Frederick V.jacobson,"RTP:A Transport Protocol for Real-Time Applicatio ns",RFC1889,1996

[4] K.Kobayashi A.Ogawa S.Casner C.Bormann,"RTP Payload Format for DV Format Video",Internet Draft,1999

[5] "FreeBSD Home-page",http://www.freebsd.org/ as of September 2000

[6] RealNetworks, http://www.realnetworks.com/

[7] Luigi Rizzo, "Dummynet", http://www.iet.unipi.it/ luigi/ip_dummynet/