社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2003－ＡＶＭ－43　（３）
2003／12／18

TCPマルチコネクションを用いた超高精細動画像の長距離・高速ストリーム伝送実験

白井 大介　山口 高弘　清水 敬司　野村 充　白川 千洋　藤井 哲郎

日本電信電話株式会社 NTT未来ねっと研究所
〒239-0847 神奈川県横須賀市光の丘1-1
E-mail: shirai.daisuke@lab.ntt.co.jp

あらまし　800万画素ディジタルシネマ配信システムを開発し，3000kmの距離を経た動画像のストリーム伝送・上映実験を行った．このとき，複数のTCPコネクションによる並列ストリーム伝送を用い，長RTT（Round Trip Time）環境下におけるスループット向上に効果があることを確認した．その後，ネットワークエミュレータを用いて長距離のネットワークを擬似的に構築し，TCPウィンドウサイズやコネクション数，伝送レート制御の有無などに対するスループットを測定した．その結果から，長RTT環境下での動画像ストリーム伝送における，スループット低下を回避する手法について考察を行う．

キーワード　ストリーム伝送，TCPマルチコネクション，長距離伝送，ディジタルシネマ

High-speed transmission of video stream over long distance using multiple TCP connections

Daisuke SHIRAI　Takahiro YAMAGUCHI　Takashi SHIMIZU　Mitsuru NOMURA　Kazuhiro SHIRAKAWA and Tetsuro FUJII

NTT Network Innovation laboratories, Nippon Telegraph And Telephone Corporation
1-1 Hikarinooka, Yokosuka-Shi, Kanagawa, 239-0847 Japan
E-mail: shirai.daisuke@lab.ntt.co.jp

Abstract We have developed a prototype 8-mega pixels digital cinema distribution system and experimented in video stream transmission over 3000km distance with the system. We used parallel stream transmission using multiple TCP connections in this experiment. As a result, it was recognized that this method for transmission was very effective to increase throughput in a long RTT (Round Trip Time) environment. Then, we measured throughput changing TCP window size, number of connections and using or not using traffic shaping in pseudo long distance network built with a network emulator. In this paper, we examine methods for preventing from decreasing throughput for video streaming in a long RTT environment.

Keyword stream transmission, multiple TCP connection, long distance transmission, digital cinema

1. はじめに

ブロードバンド環境の急速な普及に伴い，ネットワーク配信されるマルチメディアコンテンツに対する要求はより高品質のものへとシフトしている．映像メディアにおける高品質コンテンツの一つとして映画，すなわちディジタルシネマがあげられる．現在は，解像度1000本クラスのHDTV（High Definition TV）を用いた表示が主流となっているが，この解像度での表示ではフィルムが本来有している品質を充分に再現できないと指摘されている[1]．現在の映画撮影および上映に一般的に用いられている35mmフィルムは走査線1000本を凌ぐ解像度を有しており，特に撮影から上映のまでの工程の中で最も世代の若いオリジナルネガフィルムの持つ情報量はHDTVの解像度よりも大きく上回っていると言われる[2]．このため，ディジタルシネマの分野においては，HDTVよりも高品質な表示方式が求められている．

このような要求に対して，我々は解像度3840×2048（約800万）画素，RGB各10bit，毎秒24fps表示のSHD(Super High Definition)ディジタルシネマ配信システムを開発した[3]．

SHDディジタルシネマはHDTVの4倍の解像度を有し，35mmフィルムを超える画像品質の上映が可能である．

35mmオリジナルネガもしくはインターポジフィルムを1コマずつスキャンし，動画像データとして利用する．SHD動画像のビットレートは非圧縮で約5.7Gbpsとなるため，視覚的な画質劣化を起こさない程度に圧縮し，非圧縮PCM音声データと共にシネマ配信サーバに保存する．画像圧縮にはJPEG2000を利用する．動画像及び音声データは，シネマクライアント（デコーダ）の要求に応じて，ギガビットイーサネット（GbE）を用いたIPネットワークを通じてTCPスト

リーム伝送され，プロジェクタに上映される．

本システムのような大容量データの TCP ストリーム伝送を行う配信システムは，ネットワークの帯域幅だけでなく，伝送距離などに起因する伝送遅延がスループットに与える影響を充分に考慮して設計する必要がある[4]．そこで本研究では，本システムを用いた実 IP 網での長距離ストリーム伝送実験，及びネットワークエミュレータを用いた擬似的な長距離ネットワークを介したストリーム伝送実験を行い，長距離伝送におけるスループット向上のための方式について検討を行った．

2. SHD ディジタルシネマ配信システム

SHD ディジタルシネマ配信システムは，1)ストリームを受信し，データを実時間処理するデコーダ，2)SHD 表示可能なプロジェクタ，3)動画像及び音声データのストリームをネットワークに送出するシネマ配信サーバから構成されている[3]．ここでは 1)のデコーダ及び 3)のシネマ配信サーバについて説明する．

2.1. SHD リアルタイムデコーダ

IP ネットワークを介して受信した SHD 動画像データをリアルタイムでデコードして上映を行う，JPEG2000 ハードウェアデコーダである．30 個の JPEG2000 プロセッサ（Analog Devices ADV-JP2000）を搭載した PCI ボードを開発し，これを PC に 4 枚搭載している．1 枚あたり，1920×1024 画素，最大 48fps の処理能力を有しており，4 枚並列動作させることで SHD 動画像を表示可能となる．ベースとなる PC は IA-32CPU（PentiumIII 1.44GHz）を 2 個搭載した Linux PC である．また，ネットワークからのストリーム受信を行うための GbE ネットワークインタフェースと，音声出力のためのディジタル出力オーディオボードを搭載している．デコーダ制御プログラムは Linux 上で動作するアプリケーションであり，TCP ストリームの受信，デコーダボードへのデータの書き出し，音声の出力を行うスレッドから成り立っており，それぞれのスレッドは並列に動作する．

2.2. シネマ配信サーバ

シネマ配信サーバは IA-32CPU（PentiumIII 1.44GHz）×2，GbE ネットワークインタフェース，RAID システム（RAID0）を搭載した Linux PC である．動画像データは JPEG2000 を用いて，1 フレームずつ独立に符号化され RAID システムに保存される．サーバ制御プログラムは Linux 上で動作するアプリケーションであり，RAID システムからのデータの読み出し，ネットワークへ TCP ストリームの送信を行うスレッドから成り立っており，それぞれのスレッドは並列に動作する．

3. SHD 動画像の TCP ストリーム伝送

本システムの動画像 TCP ストリーム送受信におけるサーバ及びデコーダの基本的な振る舞いを以下に示す．

3.1. サーバプログラムの動作

読み出しスレッドは，動画像データを 1 フレーム単位で RAID システムから読み出し，ユーザ空間に確保された動画像リングバッファにデータを移す．リングバッファが満たされると RAID システムからの読み出しがブロックされる．ストリーム送信スレッドは，リングバッファの先頭の 1 フレーム分データを TCP ソケットに書き込む．書き込みが終わるまで動作はブロックされ，書き込みが終わるとリングバッファの空きカウンタを 1 増やし，次のフレームの送信を始める．リングバッファが空になると TCP ソケットへの書き込みがブロックされるが，RAID システムからの読み出し速度が充分速いため，通常はリングバッファが空になることはない．

3.2. デコーダプログラムの動作

ストリーム受信スレッドは，動画像データを 1 フレーム単位で TCP ソケットから読み出し，ユーザ空間に確保された動画像リングバッファにデータを移す．リングバッファが満たされると TCP ソケットからの読み出しがブロックされる．書き出しスレッドは，リングバッファの先頭の 1 フレーム分データをデコーダボードに DMA 転送する．デコーダボードが 1 フレーム表示を終えるまで DMA 転送はブロックされ，その後リングバッファの空きカウンタを 1 増やし，次のフレームの DMA 転送を始める．リングバッファが空になるとバッファアンダーランとなり，動画像のコマ落ちが発生する．デコーダの処理速度は動画像のフレームレートに依存し，24fps で動画像を表示する場合は，1/24 秒ごとに DMA 転送がブロックされる．従って，ストリーム伝送のスループットが充分に大きく，リングバッファが常に満たされている場合，ストリーム受信スレッドも 1/24 秒ごとにブロックされることとなる．同時に，TCP のフロー制御によりサーバのストリーム送信スレッドもブロックされるため，サーバの読み出しスレッドも連鎖的に影響を受け，見かけ上システム全体がデコーダボードの処理に同期し，ストリームの送受信が行われる．

4. 実 IP 網による長距離 TCP ストリーム伝送実験

2002 年 10 月 28 日，29 日，アメリカ南カリフォルニア大学（USC）において Internet2 Member meeting が開催され，ここで SHD ディジタルシネマ配信システムを用いた長距離の配信上映実験を行った[5]．サーバをシカゴのイリノイ大学（UIC）に，デコーダをロサンゼルスの USC に配置し，それぞれスイッチ及びルータを用いて Internet2 の運用する

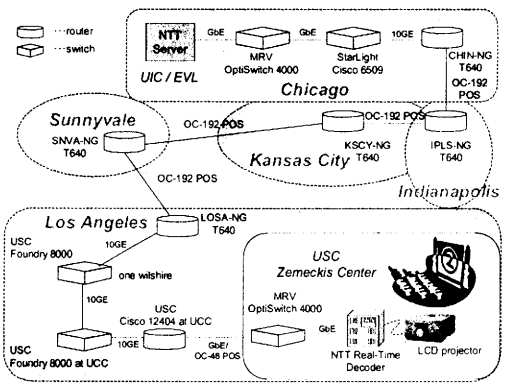

図 1 Internet2 伝送実験におけるネットワーク構成

Abilene ネットワークに接続した.

4.1. 実験環境について

実験ネットワークを図 1 に示す. アクセス回線は GbE で 1Gbps の帯域幅を有し, バックボーンとなる回線は 10GbE や OC-192 で構成されており約 10Gbps の帯域幅を有している. シカゴ—ロサンゼルス間の距離は約 3000km であり, RTT は 59msec, HOP 数は 7 であった. Internet2 には 200 以上の大学, 企業が参加しており, Abilene ネットワークのバックボーンには常時多量のトラフィックが存在する. 本実験の実施時にも, 他の実験トラフィックがロサンゼルスに向けて存在していた.

実験には, 平均ビットレートが異なる 24fps, 約 5 分間のエンコード済み動画像データを, 約 50Mbps から約 300Mbps まで 50Mbps おきに 6 種類用意して使用した. デコーダの有するリングバッファを 128 フレーム分とし, ストリーム伝送の開始後バッファが 50%以上満たされた時点で上映を開始するように設定した. それぞれのビットレートにおいて, バッファアンダーランが発生せずに最後までストリーム伝送が完了するかどうかを確かめた.

4.2. 長距離伝送時のスループット向上手法の適用

RTT が 59msec というネットワーク環境において, TCP ウィンドウサイズが 64KB である場合, 理論的なスループットの値は(64KB×8bit)／59msec≒9Mbps となる. Linux のデフォルトの TCP ウィンドウサイズは 64KB であるため, 通常の伝送では 50Mbps のストリームの伝送も不可能であることは明らかである. そのため本実験実施前にスループット向上の手法を検討し, 1)TCP ウィンドウサイズの拡大, 2)TCP マルチコネクションの利用, の 2 手法を適用して伝送実験を行った.

実験の結果, 目標としていた 300Mbps のストリーム伝送は成功しなかったため, 3)データ送出量の平滑化制御を導入し, 伝送実験を行った. 以下に各手法について説明し, 実験結果を示す.

1) TCP ウィンドウサイズの拡大

サーバ, デコーダそれぞれにおいて, デフォルトの TCP ウィンドウサイズを 64KB から 4MB に拡張する設定を行った. これによりスループットの理論値は約 570Mbps となる. しかしながら, 結果として 50Mbps のデータのみストリーム伝送が完了し, 100Mbps 以上のストリーム伝送はできなかった.

2) TCP マルチコネクションの利用

デフォルト TCP ウィンドウサイズは 4MB のまま, サーバ, デコーダ間に複数の TCP ソケットを接続し, 1 フレーム分データを各ソケットに等分して割り当て, 各ソケットに順次書き込むという形式で並列ストリーム伝送を行った. 各ソケットにおける書き込みは, 直前のソケットの書き込みが終了するまでブロックされる. コネクション数を変化させ, 各ビットレートに対する伝送実験を行った. 結果を表 1 に示す.

コネクション数 64 において 200Mbps のストリーム伝送が完了したが, 平均ビットレート 200Mbps のデータは原画像に対する圧縮率が高いため, 35mm フィルムの品質を再現するには不十分であった.

表 1 TCP マルチコネクションによる実験結果

TCPコネクション数 ＼ ビットレート(Mbps)	50	100	150	200	250	300
1	○	×	-	-	-	-
4	○	○	×	-	-	-
16	-	○	○	×	-	-
64	-	-	○	○	×	×
80	-	-	-	○	×	-

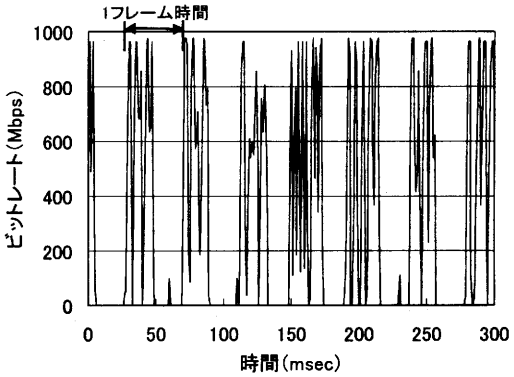

図 2 トラフィックパターン（平滑化制御なし）

3) データ送出量の平滑化制御

上述の実験において, 1msec の解像度でスループットの推移を表示可能なトラフィックモニタ[6]をサーバ側に設置し, トラフィックパターンを観察した結果, 図 2 に示すようにピークレートが 800Mbps を超えるような, バースト性の非常に高いデータ伝送が行われていることが確認された. 本実験に先立って行った, netperf を用いた UDP のスループット計測により, 実験で用いたネットワークは 800Mbps を超えるビットレートでパケット損失が発生することが認められていた. そのため, バースト性を抑える目的で, 図 3 に示すようなデータ送出の平滑化制御をサーバのソケット書き込みプロセスに実装した. 1 フレームの表示時間あたり, 確実に 1 フレーム分のデータが送出される範囲で, ソケットへの書き込み 1 回あたりのデータ量が充分小さくなるように分割し, 各書き込み処理間にウェイト（待ち時間）を入れる制御を行った. ただし, 一時的にスループットが低下した場合のバッファアンダーランを回避する目的で, 1 フレーム時間より若干短い時間で 1 フレーム分のデータが送出されるように調整した. 1 フレームあたりの書き込み回数は 4096 とし, 例えばコネクション数が 64 のとき, 各ソケットにおける書き込み総数を 64 回とした. このときの書き込み一回あたりの間隔は, (1sec／24)／4096≒10.17μsec と求められる. ただし, 前述のように書き込みの合計時間が 1 フレーム時間より短い必要があるため, 間隔を 10μsec とした. 送出間隔の制御はビジーウェイトを用いて行った. この結果, トラフィックパターンは図 4 のようになり, 300Mbps のストリーム伝送が完了した.

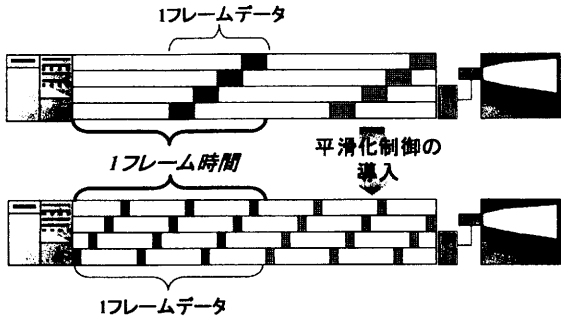

図 3 データ送出量の平滑化制御

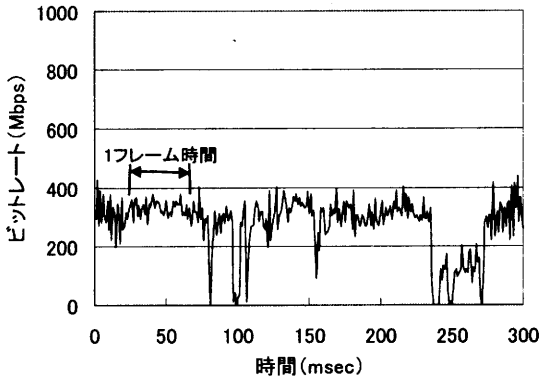

図 4 トラフィックパターン（平滑化制御あり）

平均ビットレート 300Mbps のデータは SHD 解像度での鑑賞には充分堪えうる品質であった．なお，コネクション数を 64 以上にしても伝送可能なビットレートは増加しなかった．

5. 疑似的長距離ネットワークにおける伝送実験

Internet2 実験の結果から，長距離 TCP ストリーム伝送において，TCP ウィンドウサイズ，マルチコネクション数，平滑化制御の有無がスループットに大きく寄与することが確認できた．これらのパラメータに対するスループットの変化について，より詳細な評価を行うことと，さらなるスループット向上の検討を行うため，GbE における遅延やパケット損失などを発生させることが可能なネットワークエミュレータ（PacketSphere）を用い，擬似的な長距離ネットワークを構成し，ストリーム伝送実験を行った．

5.1. 実験環境について

実験に用いたネットワークの構成を図 5 に示す．ネットワークエミュレータ内での遅延は，片方向で 30msec，往復で 60msec に設定した．Internet2 における伝送実験でのネットワークは，バックボーンの帯域幅が 10Gbps でアクセス回線の帯域幅が 1Gbps という構成であったが，本実験におけるネットワークは帯域幅が全て 1Gbps であり，HOP 数は 1 という構成になっている．

Internet2 における伝送実験の際は，フレームごとにビットレートが変動する実際の JPEG2000 動画像データをストリーム伝送に用いたが，本実験ではビットレートを固定したダミーデータを用いた．フレームレートは 24fps で，7200 フレーム（5 分間）のデータをストリーム伝送し，バッファアンダーランが起きることなく完了するかを，50Mbps から 450Mbps まで 50Mbps ごとに確かめた．

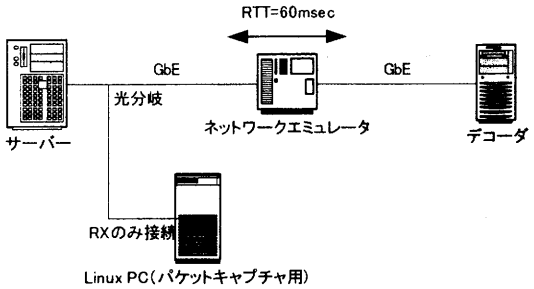

図 5 擬似的長距離ネットワークの構成

表 2 エミュレータを用いた実験結果

TCPコネクション数	ストリーム完了ビットレート(Mbps)			
	Window Size = 64KB		Window Size = 4MB	
	平滑化なし	平滑化あり	平滑化なし	平滑化あり
2	0	0	0	50
4	0	0	50	50
8	0	50	50	50
16	50	50	50	50
32	100	100	50	50
64	100	100	100	100
120	100	150	150	100

5.2. TCP マルチコネクション，平滑化制御の適用

1)TCP ウィンドウサイズの拡大，2)TCP マルチコネクションの利用，3)データ送出量の平滑化制御について，パラメータを変化させてストリーム伝送を行った．平滑化制御における 1 フレームあたりの書き込み回数は，4.2 で述べた通り 4096 とした．結果を表 2 に示す．

この実験において，TCP のコネクション数の増加がスループットに寄与することは確認できた．ただし，平滑化制御に関しては，導入することによってバッファアンダーランまでの時間は長くなったが，ストリーム完了まで至らないというケースが多く見られた．TCP ウィンドウサイズに関しては，拡張してもスループットの向上はほとんど見られなかった．

TCP ウィンドウサイズを 4MB に設定し，サーバにおいて tcpdump を用いてパケットをキャプチャしてみたところ，デコーダからは常に約 3MB のウィンドウサイズ通知が送られていた．設定値より低い値であるが，理論的には約 400Mbps の伝送が可能である．また，パケット損失は発生していなかったため，再送制御に伴う伝送レートの低下は無かった．従って，Linux における TCP 制御アルゴリズムは，大きいウィンドウサイズが通知されてもそれを満たすだけのパケットを送出しないという特性があるものと考えられる．

5.3. 空きバッファ量に応じた送信レート制御の導入

上述の実験において平滑化制御を適用した際，デコーダの動画像リングバッファが満たされた後にスループットが急激に低下し，バッファアンダーランになるというケースが多く見られた．

3.2 で述べたように，デコーダプログラムにおけるストリーム受信スレッドの動作は，リングバッファが満たされるとブロックされる．同時に，受信ソケットバッファの空き容量が減少するため，デコーダ側からは定常時より少ないウィンドウサイズが通知される．デコーダからの ACK パケットを調べたところ，定常時では 64KB のウィンドウサイズが通知されていたのに対し，リングバッファが満たされた後では一時的に 15KB 程度まで減少していた．

また，平滑化制御における各書き込みのタイミングは，誤差の蓄積によるスループットの変動を避けるために，直前の書き込みから一定の間隔をおいて行われるのではなく，1 フレームの表示時間で完結するように決まった時刻で行われる．ある時刻での書き込みのタイミングが遅れた場合は，次の書き込みまでの時間間隔は短くなり，次の書き込み時刻を越えて遅れた場合は，次の書き込みは待ち時間無しで直ちに行われる．このため，書き込み処理のブロッキングが発生すると，バースト性の高いデータ送出になる可能性がある．従って，ウィンドウサイズの減少に伴うデータ送信レートの低下と，それによって引き起こされるバースト性の増加により，スループットが回復しないままバッファアンダーランが発生すると考えられる．

そこで，デコーダのリングバッファの空き容量が少なくなるとサーバ側に書き込みの間隔を広げるように指示を出し，空き容量が回復したところで元の間隔に戻すように指示を出すことで送信レートの制御を行った．具体的には，空きバッファ量が 20%以下になった時点で，書き込み間隔を 10μsec から 11μsec に広げ，空きバッファ量が 50%以上になった時点で書き込み間隔を 11μsec から 10μsec に戻すように制御を行った．この制御を導入し，ウィンドウサイズをデフォルトの 64KB に設定し，TCP コネクション数を変化させてストリーム伝送実験を行った．結果を表 3 に示す．

結果として，TCP コネクション数 120 において，300Mbps のストリーム伝送が可能となった．

5.4. ノンブロッキング処理による伝送効率の向上

これまでの実験における TCP マルチコネクションや平滑化制御では，複数のソケットへのデータの書き込み及び読み出し処理は全て単一のスレッド内で行われ，それぞれブロッキングモードで動作していた．そのため，あるソケットにおいてパケットの到着が遅れたり，一時的にスループットが低下した場合などに，他のソケットにおける処理が遅れ，送受信の効率が低下するという問題がある．これに対し，各ソケットの処理をそれぞれ別のスレッドで行い，並列処理するという手法が考えられるが，スレッド切り替えのオーバーヘッドは無視できないほど大きい．また，Linux のスレッド制御の時間的粒度はカーネル 2.4 系列で 10msec，カーネル 2.6 系列でも 1msec であり，10μsec という単位での平滑化制御は不可能である．

そこで，ソケットをノンブロッキングモードに設定した上で TCP マルチコネクションによる伝送を導入した．例えば送信側でデータ送出が遅れ，バッファに空きが無いソケットが発生した際は，バッファの空き容量分だけデータを書き込んで，次のソケットへの書き込みを試みるという動作になる．各ソケットへの書き込みは，1 フレーム分のデータ送信が完了するまで繰り返される（割り当てられたデータを全て書き込んだソケットに関してはスキップされる）．これは受信側においても同様の動作となる．

また，ノンブロッキングモードでの伝送時においても，平滑化処理を導入して実験を行った．各書き込みにおけるデータ量は不定であるため，一定量（約 3KB）以上のデータが書き込まれた後に一定時間（約 10μsec）のウェイトを入れるようにしてデータ送出量の平滑化を行った．ウィンドウサイズを 64KB に設定し，平滑化制御の有無それぞれについて TCP コネクション数を変化させてストリーム伝送実験を行った．結果を表 4 に示す．

ノンブロッキングモードによる伝送と，平滑化制御を導入することで，TCP コネクション数 120 において 400Mbps のストリーム伝送が可能であることが確認できた．

5.5. 各手法におけるトラフィックパターンの測定

各手法を適用した際のトラフィックパターンの違いを調査するため，サーバから送出されたパケットをキャプチャし，単位時間を 1msec として各時刻におけるスループットを計算し，グラフを作成した．230Mbps のデータをストリーム伝送し，平滑化制御を適用しなかったときの結果の一部を切り出して図 6 に示す．同様に，平滑化制御を適用した結果（空きバッファ量に応じた送信レート制御は適用せず）を図 7 に，ノンブロッキングモードによる伝送と平滑化制御を適用したときの結果を図 8 に示す．図 6 ではバッファアンダーランが発生し，図 7,8 においてはストリーム伝送が完了した．

平滑化処理の有無によってトラフィックパターンが大きく変わることが確認できる．また，データのビットレートが

表 3 空きバッファ量に応じた送信レート制御の導入

TCPコネクション数	ビットレート (Mbps)
2	0
4	0
8	50
16	50
32	100
64	200
120	300

Window Size = 64KB

表 4 ノンブロッキングモードを利用した伝送

TCPコネクション数	ビットレート(Mbps)	
	平滑化なし	平滑化あり
2	0	0
4	0	0
8	50	50
16	100	100
32	50	150
64	150	250
120	100	400

Window Size = 64KB

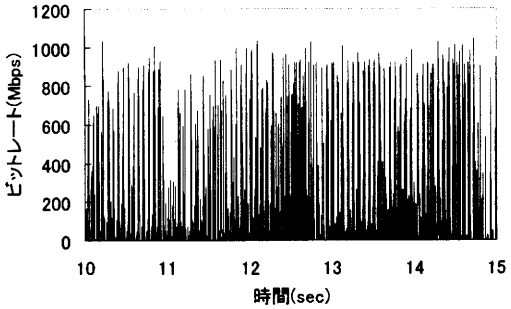

図 6 トラフィックパターン（平滑化制御なし）

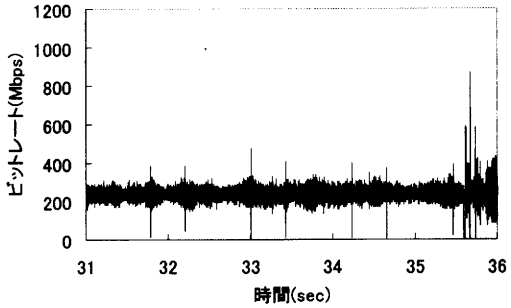

図 7 トラフィックパターン（平滑化制御あり）

高すぎてバッファアンダーランが起きた場合などは，ほぼ例外なく図6のような針山状のトラフィックパターンが確認された．ウィンドウサイズを 4MB にした場合にも，平滑化制御の有無に関わらず図6のようなパターンが見られた．ウィンドウサイズが 4MB の時は，GbE インタフェースからパケットが送出される段階で，平滑化制御が何らかの原因で機能しなかったと見られる．

6. 考察

TCP ウィンドウサイズを拡張することにより，理論上のスループットは向上するはずであるが，本システムにおいては有効な手段とはならなかった．Linux の TCP プロトコルスタックには，輻輳ウィンドウを用いた独自の輻輳回避アルゴリズムが実装されている．受信側から充分な大きさの受信ウィンドウサイズが通知されているにもかかわらず，送信側でその大きさに応じたデータ送信を行わないのは，輻輳ウィンドウサイズが充分に広がらないためだと考えられる．

一方，TCP マルチコネクションを使ったデータ伝送はスループット向上に大きく寄与した．輻輳回避アルゴリズムによる制限は TCP ソケット毎に行われるため，実質的な輻輳ウィンドウサイズが拡大するからだと考えられる．複数のトラフィックが混在するネットワーク上においても，TCP マルチコネクションによる並列伝送は，公平性を保ちつつ高い伝送レートを望むことができるとされている[7][8].

また，Linux の TCP の実装において，ソケットバッファからネットワークデバイス用のバッファへのデータ送出量が大きい場合，このバッファがあふれ，それを輻輳とみなす機構がある[9]. 平滑化制御によりスループットが大きく向上したのは，バースト性を抑えることによるネットワークへの負荷

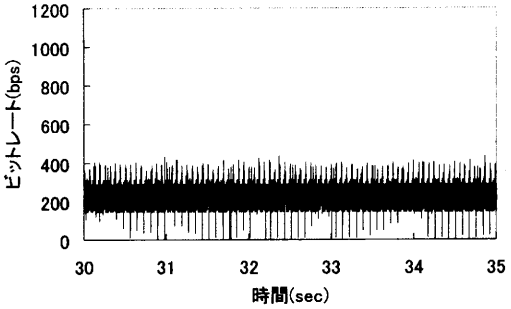

図 8 トラフィックパターン（ノンブロッキング＋平滑化）

の低減に加え，ソケットバッファの送出量を抑えることで，ネットワークデバイス用バッファがあふれるのを抑え，輻輳回避制御が行われないようになるためと考えられる．

7. まとめ

本稿では，SHD ディジタルシネマ配信システムを用いた，長距離の動画像ストリーム伝送実験において，各種のスループット向上のアプローチを適用した結果について述べた．本システムを用いた長距離ストリーム伝送において，TCP マルチコネクション数，平滑化制御の適用と，受信バッファ量に応じた送信レート制御，ノンブロッキングモードの利用がスループット向上に大きく関係することが確認された．結果として，TCP マルチコネクションによる並列伝送に平滑化制御を導入し，ノンブロッキングモードによる伝送の効率化を図ることで，スループットを最も大きく向上できることを確認した．

文　献

[1] http://www.screendigest.com/digitalcinema.htm

[2] R.R.A.Morton, M.A.Maure, and C.L.DuMout, "Assessing the Quality of Motion Picture Systems from Scene to Digital Data," SMPTEJ, Vol. 111, No.2, pp.85-96, 2002.

[3] T. Fujii, M. Nomura, D. Shirai, T. Yamaguchi, T. Fujii, and S. Ono, "Digital cinema system using JPEG2000 movie of 8 million pixel resolution," IS&T/SPIE Electronic Imaging VICP2003, no.5022-07, pp. 50-57, Santa Clara, California, U.S.A., Jan. 2003.

[4] V. Jacobson, R. Braden, "TCP Extensions for Long-Delay Paths," RFC1072, Oct. 1988.

[5] 白井, 清水, 室岡, 山口, 藤井, 藤井, 高原, 小口, "TCP マルチコネクションによる 800 万画素ディジタルシネマの 3000km・300Mbps ストリーム伝送実験," 2003 信学全大, no. B-7-68, pp. 328, Mar. 2003.

[6] 室岡, 橋本, 清水, 白井, "高時間分解能リアルタイムアプリケーショントラフィックモニタ," 2003 信学全大, no.SB-4-1, pp. S.1-S.2, Mar 2003.

[7] T. J. Hacker, B. D. Athey, J. Sommerfield, "Experiences Using Web100 for Host Tuning," The Web100 Project, http://www.web100.org/docs/

[8] H. Sivakumar, S. Bailey, R. L. Grossman, "PSockets: The Case for Application-level Network Striping for Data Intensive Applications using High Speed Wide Area Networks," Proceedings of Supercomputing, pp.38-43, Dallas, Texas, U.S.A., Oct, 2000.

[9] 高野, 石川, 工藤, 松田, 児玉, 手塚, "並列アプリケーション実行における TCP/IP 通信挙動の解析," インターネットコンファレンス 2003, No.26, pp. 11-18, Oct. 2003.