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

2006－OS－101（13）
2006／2／17

分散環境での映像・音声入出力デバイスの管理手法に関する考察

芝 公仁[†]

映像や音声を入出力可能なデバイスを計算機に接続し使用することは珍しくなくなっている．また，高速なネットワークの普及に伴って，映像や音声をネットワークを介して計算機間で送受信することも簡単に行えるようになっている．一方，映像や音声に対する処理は，高度で複雑なものになってきており，複数の計算機を使用して処理を行いたいといった要求がある．しかし，既存のシステムには，複数の計算機から複数の映像・音声入出力デバイスを使用するようなアプリケーションを支援する機能が十分には備わっていない．本稿では，分散環境における映像・音声入出力デバイスの管理に要求される機能について考察し，これを実現するデバイス管理手法について述べる．

A Management Method of Audio and Video Devices in Distributed Environment

Masahito Shiba[†]

Audio devices and video devices which are connected to computers have become popular and easy to use. High-speed networks have made sending data got from these devices easy, too. On the other hand, analyzing these data has become complicated and it is required to use multiple computers to analyze these data. However, traditional systems do not have enough functions to support applications using multiple audio/video devices on multiple computers. In this paper, functions required by applications using audio/video devices in distributed environment and a device management method providing these functions are described.

1　はじめに

PC などの計算機に接続できる入出力デバイスには様々なものがあり，映像や音声を入出力するデバイスも広く使用されている．映像入力デバイスに関しても，近年，安価に入手することが可能になり，その使用方法も簡単になってきている．また，高速なネットワークの普及に伴って，計算機を用いた遠隔地とのコミュニケーションに，テキストデータだけではなく，映像・音声のデータが利用されるようになってきている．さらに，映像・音声のデータを単純に送受信するだけではなく，これらを計算機で解析することも行われている．データの解析は，近年，より高度で複雑なものになってきており，複数の計算機で処理したいといった要求もある．

映像・音声入出力デバイスの使用やその応用は広がってきているが，これらのデバイスを管理するデバイスドライバあるいはオペレーティングシステムは十分な機能を提供しているとは言えない．多くの場合，デバイスドライバの機能が，オペレーティングシステムによって共通のインタフェースにまとめられ，アプリケーションに提供されるだけである．このとき，デバイスの種類に応じたインタフェースは提供されるが，映像や音声を扱うデバイスに対しては，必ずしも十分なものとは言えない．そのため，デバイスを使用するアプリケーションが自身でデバイス管理を行うことも多い．しかし，これは，複数の計算機が使用されるような分散環境においては望ましくない．特に，複数のアプリケーションが動作するような環境では，デバイス管理は，個々のアプリケーションではなく，基盤となるシステムによって行われるべきである．

本稿では，このような問題を解決するデバイス管理手法について述べる．また，本手法を用いてデバイス管理行うシステムについても述べる．本システムは，ネットワークで接続された複数の計算機が動作する環境において，それらの計算機に接続された映像・音声入出力デバイスを管理する．本システムのデバイス管理は，次の 2 つの特徴を持つ．

- 映像や音声の特徴を考慮したデバイス管理
- 分散環境への対応

映像や音声のデータは時間制約を有するため，アプリケーションだけではなく，基盤となるシステムでもこれに対応することは重要である．また，映像や音声は，複数のデータの合成や，フィルタの適用が行われるデータである．本システムは，これらに

[†]龍谷大学理工学部
Faculty of Science and Technology, Ryukoku University

対応し，異なる計算機上で動作する複数のアプリケーションがデバイスを共有することや，複数のデバイスをまとめて単一のデバイスとして扱うことを可能としている．これによって，分散しているデバイスを一括して操作することや，デバイスから入力されるデータを多数の計算機で処理し，より高度な機能を実現するといったことが可能となる．

以下，本稿では，2 章，3 章で関連する研究とデバイス管理に要求される機能について述べる．続いて，4 章で本システムの概要，5 章，6 章でデバイス管理手法，7 章でシステムの動作例について述べる．また，8 章でシステムの評価を行い，本手法の有効性について議論する．

2 関連研究

分散環境において，システムがアプリケーションにどのような機能を提供するかは重要であり，CORBA[1] に代表されるような分散環境を対象としたミドルウェアは様々提案されてきている．また，ファイルの共有 [2, 3] のように，単純なデータの共有を実現する手法も様々ある．しかし，映像・音声入出力デバイスの管理では，データの特徴を考慮することが重要であり，これらの手法だけでは十分ではない．

複数のアプリケーションから音声デバイスを共有できるようにするシステムとして，Jack[4] がある．逆に，一つのシステムが複数のデバイスを使用する例としては，音場再現 [5] やネットワークを介して演奏情報を送受信するシステム [6] などがある．これらは，時間制約について厳密に考慮する必要があり，これを支援する機能を基盤となるシステムが提供できることが望ましい．

映像入力デバイスも，多地点会議や遠隔講義など様々なシステムで使用されている．また，デバイス自体にも様々な種類があるが，これらを統一したインタフェースで使用できるようにする [7] ことで，アプリケーションの作成が容易になる．しかし，分散環境において複数のアプリケーションが複数のデバイスを使用するような場合には，基盤となるシステムにはより高度な機能が求められる．

3 デバイス管理の要件

デバイスの抽象化は，通常，デバイスを実際に制御するデバイスドライバやそれらを管理するオペレーティングシステムによって行われている．しかし，分散されたデバイスを複数のアプリケーションから使用できるようにするためには，分散環境に適したより高度な抽象化が必要になる．ここでは，特に，音声の出力，映像の入力を例に挙げ，分散環境でのデバイス管理に要求される機能について述べる．

3.1 音声出力

計算機に接続される出力装置には様々なものがあるが，ここでは，スピーカなどから音声出力を行うデバイスについて考える．ネットワークを利用した音声出力デバイスの応用としては，遠隔地のユーザとの音声によるコミュニケーションがある．また，一対一のコミュニケーションだけではなく，遠隔講義や多地点会議などの応用もある．これらを快適に行うためには，音声伝達の遅延が十分小さくなければならない．

また，広い範囲に音声を伝達することや，臨場感を高めることなどを目的に，多くのスピーカを使用することがある．PC で使用されるサウンドカード等は，通常，複数のスピーカを接続でき，それぞれ異なる音声を出力できる．しかし，より多くのスピーカを使用することが有効な場合もある．例えば，音場再現を行う場合，目的の空間を囲むように多数のスピーカが利用される．

多くのシステムでは，これらの応用において，アプリケーション自身が音声出力デバイスの管理を行っている．しかし，音声出力デバイス用いた様々なアプリケーションを動作させる場合，次のような機能は，基盤となるステムが提供すべきであると考えられる．

- 複数のアプリケーションから音声出力デバイスを共有
- 個々の音声出力デバイスの操作方法の違いを隠蔽
- 時間制約や位置を考慮した音声出力デバイスの制御

アプリケーション自身が音声出力デバイスを管理し制御や操作を行った場合，他のアプリケーションは当該デバイスを使用することができない．適当なシステムが音声出力デバイスを管理し，それを利用するアプリケーションに適切に資源を配分することで，アプリケーション間でのデバイスの共有を実現することができる．特に，音声の場合，同時に複数の出力要求があった場合，それらを合成して出力することも可能である．

多数の音声出力デバイスを使用する場合，種類の異なるデバイスが混在する可能性が高くなる．その場合，デバイスの種類ごとに制御方法が異なるが，アプリケーションそれぞれが，各デバイスに対応す

るのでは効率が悪い．また，音量の設定などは，すべての音声出力デバイスで統一された尺度で設定できると便利である．基盤となるシステムがデバイス間の差異を吸収し，統一されたインタフェースをアプリケーションに提供することによって，アプリケーションの作成が容易になる．

また，音声データは時間制約を有するため，基盤となるシステムでもこれを考慮したデバイス管理が必要になる．特に，音声はスピーカから出力された後，ユーザにそれが伝達するまでにミリ秒単位の時間がかかるため，スピーカの位置も考慮した制御が必要になる．音声は，ユーザへの情報伝達に重要な手段であり，生活空間にデバイスを組み込む試みが行われているが，このような場合にも，どの範囲に音声を伝えることができるのかなどの位置管理が必要になる．

3.2 映像入力

映像は，デバイスからの出力後，ユーザまでの伝達速度が十分速く，音声のように伝達速度を意識する必要はない．むしろ，入力に関して注意を払う必要がある．映像を入力可能なデバイスには，スキャナやデジタルスチルカメラなどがあるが，ここでは，特に，デジタルビデオカメラに着目する．デジタルビデオカメラは，映像データを扱うため，データやそれに対する処理が時間制約を持つ．

高速なネットワークの普及によって，計算機を用いた遠隔地とのコミュニケーションにおいて，音声だけでなく，映像も用いることが容易になってきている．このような処理は，特に一対一の通信では，単純なアプリケーションで十分実現可能である．しかし，アプリケーションが独自に映像入力デバイスの制御を行った場合，一つのアプリケーションがデバイスを占有するため，他のアプリケーションは，当該デバイスを使用することができなくなる．

遠隔地間のコミュニケーションだけではなく，限られた範囲を複数の映像入力デバイスで撮影することも多い．特に，監視を目的とする場合などには，死角を減らすために多数のデバイスが必要になったり，広い範囲にデバイスを設置したりすることが必要になる．この場合，単一の計算機にすべての映像入力デバイスを接続することは困難である．

このように映像入力デバイスを使用する場合には，デバイスの使用を支援する機能を，基盤となるシステムで提供すべきであると考えられる．特に，次の3つの機能は重要である．

- 複数のアプリケーションから映像入力デバイスを共有
- 個々の映像入力デバイスの操作方法の違いを隠蔽
- 時間制約や位置を考慮した映像入力デバイスの制御

これらは，基本的に音声出力デバイスと同様のものである．例えば，音声出力デバイスと同様に，複数のアプリケーションから映像入力デバイスを共有できることが望ましい．アプリケーションが直接デバイスを操作するのではなく，基盤となるシステムがデバイスを多重化し，アプリケーションに提供することによって，デバイスの共有が可能になる．また，各アプリケーションは，他のアプリケーションの動作を意識することなく動作可能となる．

PC などの計算機に映像を取り込むためのデバイスには，USB 接続のものや，IEEE 1394 接続のものなどいくつかの種類がある．分散環境で複数の映像入力デバイスを扱う場合，異なる種類のデバイスが混在することも多くなると考えられるが，これらのデバイスの差異は基盤となるシステムで吸収できることが望ましい．

単一の計算機で管理できる映像入力デバイスの数には限界があるため，多数のデバイスを使用するさいには，複数の計算機を使用する必要がある．また，Web カメラなど，ネットワークに直接接続されるデバイスが使用されることもある．このとき，リモートのデバイスに対しても，ローカルのデバイスと同様のインタフェースで操作できることが望ましい．基盤となるシステムが，ネットワーク間の通信処理をアプリケーションの代りに行うことによって，アプリケーションはローカルとリモートのデバイスを区別することなく利用することが可能になる．また，音声出力デバイスの管理と同様に，映像入力デバイスの位置や撮影範囲などを一括して管理する機能も，アプリケーションごとではなく，基盤となるシステムで実現されるべきである．

4 システムの概要

前章で述べたようなデバイス管理を実現するためには，入出力デバイスを使用するアプリケーションのための基盤となるシステムが必要となる．本研究では，これを実現するためのプロトタイプシステムを Linux 上に構築した．本システムは，音声出力デバイスの操作に ALSA（Advanced Linux Sound Architecture）[8] を，映像入力デバイスの操作に Open Source Computer Vision Library [9] を使用している．Open Source Computer Vision Library は，Linux と Windows で利用可能であり，

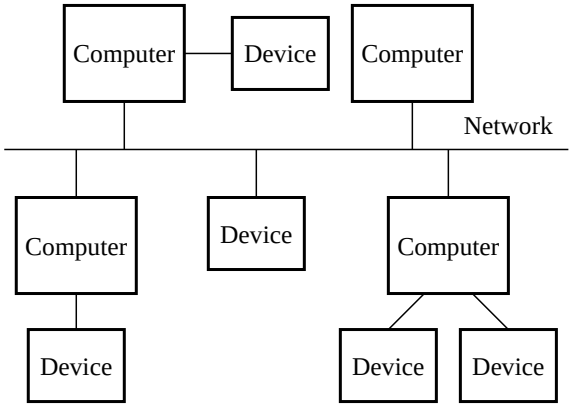

図 1　システムの動作環境

Linux では，Video4Linux を使用してデバイスからの映像入力を行う．

4.1　システム構成

図 1 は，本システムが動作する環境を示している．本システムは，ネットワークで接続された複数の計算機上で動作し，それらの計算機に接続された，あるいは，ネットワークに直接接続された音声・映像入出力デバイスを管理する．

本システムは，デバイスを管理するサーバと，アプリケーションにリンクされサーバとの通信を行うライブラリから構成される．いくつかの計算機上ではサーバが動作し，各サーバは自身が担当するデバイスの管理･操作を行う．アプリケーションは直接デバイスを操作することはなく，サーバを介してデータの入力などを行う．実際にサーバが行うデバイスの操作は，オペレーティングシステムが持つデバイスドライバを用いたり，HTTP を用いて直接デバイスと通信したりするなど，対象となるデバイスによって異なる方法をとる．しかし，サーバがアプリケーションに提供するデバイス操作のインタフェースは，デバイスによらず同じものである．

サーバには，マスタサーバとスレーブサーバの 2 つがあり，動作・役割が異なる．マスタサーバはシステム内に一つであり，他のサーバはすべてスレーブサーバとして動作する．スレーブサーバは，起動時に自身をマスタサーバに登録し，その後アプリケーションへのサービスを開始する．マスタサーバもスレーブサーバと同様にアプリケーションに対してサービスを提供するが，さらに，システム全体の情報を管理する役割を果たす．

4.2　デバイスの抽象化

サーバは，実際のデバイスを実デバイス，抽象化されアプリケーションに提供されるデバイスを仮想デバイスとして管理する．デバイスを直接アプリケーションに提供するのではなく，抽象化を行うことによって，次のような機能を実現することができる．

- デバイスの操作を位置透過に行う．
- 複数のアプリケーションからデバイスを共有する．
- 複数のデバイスを組み合わせる．
- 入出力されるデータを加工する．

通常，デバイスの操作はローカルの計算機に接続されたものに対してしか行うことができない．しかし，仮想デバイスを用いると，サーバがアプリケーションの代りに実際のデバイス操作を行うため，アプリケーションはリモートの計算機に接続されたデバイスも操作することができる．また，サーバとの通信はライブラリによって行われるため，アプリケーション自身は実際の通信を意識する必要はない．そのため，アプリケーションは，どの計算機に接続された音声出力デバイスからも，常に同一の方法で，音声を出力することができる．同様に，映像を入力するさいも，デバイスを操作する方法はデバイスの位置に依存しない．

本システムでは，複数のアプリケーションが一つの仮想デバイスを共有することが可能である．このとき，仮想デバイスの操作には位置透過性が実現されているため，異なる計算機上で動作するアプリケーション間でも共有可能である．特に，入力された映像を解析するような場合，高い計算能力が必要とされる．本システムを使用すると，映像入力デバイスを共有することができるため，複数の計算機を使用して映像解析の処理を行うことが可能となる．これを利用すると，単一の計算機ではできないような，より高度な処理を実現できる．

実デバイスを複数組み合わせて，一つの仮想デバイスとすることもできる．これを用いると，単一の仮想デバイスに出力するのみで，複数の実デバイスから音声を出力することが可能である．また，複数の映像入力デバイスを組み合わせることで，広い範囲を撮影できるデバイスや，冗長性を持つ障害に強いデバイスを実現することができる．

映像や音声は，入出力を行うデバイスや処理内容によって，加工すると便利な場合がある．例えば，異なる位置にある音声出力デバイスを用いる場合，デバイスの場所によって，音量や出力の時刻をずらした方がよりよい音声を出力できることがある．映像入力デバイスでは，サーバでノイズを除去し，その映像を仮想デバイスの出力とすることができる．また，ネットワーク帯域幅が十分でない場合，映像のサイズを小さくしてから出力する仮想デバイスは有用である．

5　実デバイス管理

実デバイスはサーバによって管理されるが，どのような実デバイスがあり，どのサーバがどの実デバイスを管理しているかなどの情報は，実デバイステーブルによって管理される．実デバイステーブルは，マスタサーバよって管理され，各デバイスについて次のような情報を持つ．

- ID: デバイスを識別するシステム全体で固有の整数値
- Type: デバイスの種類
- Name: 当該デバイスの名前
- Location: 当該デバイス，あるいは，入出力の対象となる場所
- Server: 当該デバイスを管理するサーバ

ID は，実デバイスを識別するための，デバイスごとに固有の整数値である．デバイスを検出したサーバが，それをマスタサーバに通知すると，マスタサーバは当該デバイスの ID を決め，実デバイステーブルに登録する．実デバイスが切り放された場合，マスタサーバは当該デバイスのエントリを実デバイステーブルから削除する．

Type は，音声出力デバイスや映像入力デバイスなど，実デバイスの種類を表す．解像度や色数など，デバイスの種類に固有の情報も，実デバイステーブルによって管理される．

Name は当該デバイスの名前であり，通常，製造元などが付ける製品名である．これはデバイス自身が持つ文字列であり，サーバは実デバイスの初期化時にこれを取得する．

Location は，実デバイスの設置された場所や，実デバイスが入出力の対象とする範囲を表す文字列である．アプリケーションは，任意の文字列を Location に設定することができる．Location は，アプリケーションが入力あるいは出力に利用する実デバイスを検索するさいに使用される．

Server は，当該デバイスを管理するサーバを表す．通常，実デバイスは接続されている計算機上で動作するサーバによって管理されるが，ネットワークに直接接続される実デバイスも，いずれかのサーバによって管理される．実デバイスの入出力はすべての Server で示されるサーバによって行われ，アプリケーションが直接実デバイスを操作することはない．

実デバイステーブルは，マスタサーバに管理され，そのエントリの全部または一部は，スレーブサーバにキャッシュされる．アプリケーションも実デバイステーブルを参照することが可能であり，これを用いることで，分散されたデバイスのなかから目的のものを容易に見つけることができる．

6　仮想デバイス管理

6.1　仮想デバイス

仮想デバイスは，アプリケーションが分散環境において容易にデバイスを利用できるよう，実デバイスを抽象化したものである．サーバは，アプリケーションからの要求に応じて，動的に仮想デバイスを作成する．

図 2 は，実デバイスと仮想デバイスの対応を示している．仮想デバイスは，1 つ以上の実デバイスから実現される．異なる計算機に接続された実デバイスを組み合わせて仮想デバイスを作ることもでき，これによって複数の計算機のデバイスを一括して操作することが可能になる．また，単一の実デバイスから，複数の仮想デバイスを構築することもできる．

実デバイスは，通常，カメラなど実際のデバイスであるが，動画ファイルなどを用いることもできる．実デバイスに動画ファイルを用いた場合，仮想デバイスからは，その動画のフレームが順に出力される．また，実際のデバイスを用いずに，無音のデータ入力されるマイクや，どこにも出力されない音声出力デバイスを実現することもできる．

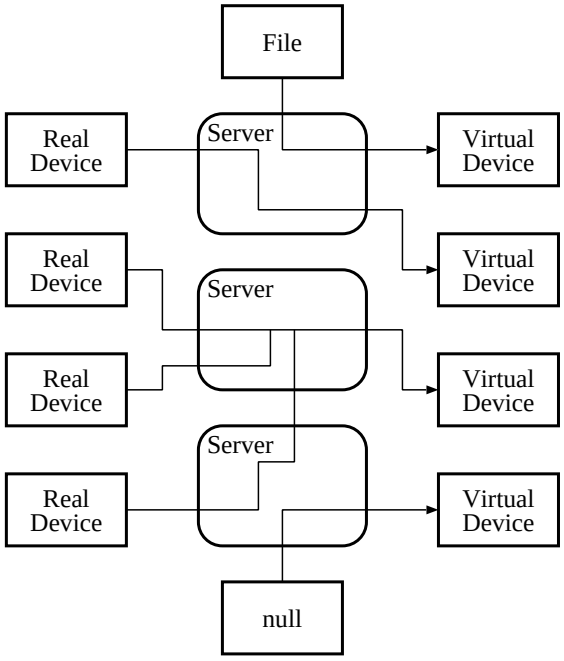

図 2　実デバイスと仮想デバイス

6.2 仮想デバイステーブル

仮想デバイスは，次のような情報を持つ仮想デバイステーブルによって管理される．

- ID: デバイスを識別するシステム全体で固有の整数値
- Type: デバイスの種類
- Name: 当該デバイスの名前
- Server: 当該デバイスを管理するサーバ
- Real Device: 当該デバイスを実現する実デバイス

アプリケーションからの要求に応じて，マスタサーバは仮想デバイスを作成する．このとき，マスタサーバは，当該デバイスの ID とする整数値を決定し，当該デバイスを仮想デバイステーブルに登録する．

Type は，音声出力デバイスや映像入力デバイスなど，デバイスの種類を表す．仮想デバイスは実デバイスを組み合わせて実現されるが，組み合わされる実デバイスは同一の種類のもののみである．仮想デバイステーブルの Type は，組み合わされる実デバイスの種類をもとに設定される．また，デバイスの種類に応じた固有の情報も仮想デバイステーブルによって管理される．

Name は，当該デバイスの名前であり，アプリケーションが自由に設定することができる．これは，目的の仮想デバイスを検索するさいに使用され，特に，アプリケーション間で仮想デバイスを共有するときに有用である．

Server は，当該仮想デバイスを管理するサーバを示す．通常，Server で示されるサーバは，当該仮想デバイスを実現する実デバイスの一つが接続された計算機で動作しているものである．Server で示されるサーバは，アプリケーションからの当該仮想デバイスに対する要求を受け，それを処理する．このとき，他のサーバに管理される実デバイスを操作する必要があれば，そのサーバを使用して実デバイスの操作を行う．

Real Device は，当該仮想デバイスを実現する実デバイスの ID を保持する．すなわち，仮想デバイステーブルの Real Device は，実デバイステーブルへのリンクに相当する．

仮想デバイステーブルも，実デバイステーブルと同様に，アプリケーションから参照することができる．仮想デバイスを使用する場合，仮想デバイステーブルから目的のデバイスを管理するサーバを調べ，そのサーバへ要求を送る．この手順はデバイスの位置に依存しないため，どのデバイスに対する操作でも同一の方法で行うことができる．また，実際の通信はライブラリによって行われるため，通信処理を意識することなく，アプリケーションを作成することができる．

7 動作例

本システム上で動作するアプリケーションは，サーバと通信することでデバイスを使用する．図 3 は，アプリケーションがライブラリを使用して，2 つのサーバと通信しデバイスを操作している様子を表している．仮想デバイスの操作は，当該デバイスを管理するサーバに要求を出す必要がある．そのため，異なるサーバが管理する仮想デバイスを使用する場合，それぞれのデバイスに応じて適切なサーバと通信する必要がある．

映像入力デバイスからフレームを入力する場合，アプリケーションとサーバは次のように動作する．

(1) アプリケーションが目的の仮想デバイスを管理するサーバにフレームの取得を要求する．
(2) 要求を受けたサーバは，対象の仮想デバイスを実現する実デバイスからフレームを取得する．また，他のサーバが管理する実デバイスからの入力も必要な場合，それらのサーバにフレーム取得の要求を送る．
(3) 要求を受けたサーバは，実デバイスからフレームを取得し，要求元のサーバに送る．
(4) 仮想デバイスを実現する実デバイスからのフレームを集めたサーバは，それらを元に仮想デバイスの出力となるフレームを作成する．
(5) 完成したフレームを要求元のアプリケーションに送る．

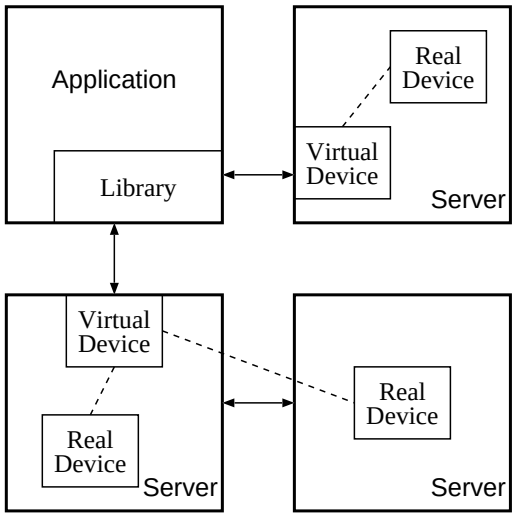

図 3 ライブラリを使用した仮想デバイスの操作

仮想デバイスからフレームを取得する手順は，本システムを使用せずにローカルのデバイスからフレームを直接取得する手順と同様である．すなわち，要求を出し，フレームを取得する．これは，リモートの計算機に接続されたデバイスからのフレーム取得でも同様であり，アプリケーションはデバイスを位置透過に使用することができる．また，本システムが実現する仮想デバイスにより，複数のデバイスから一括してデータを入力することも可能である．

また，入力と同様の手順で，仮想デバイスからの出力処理も行われる．すなわち，対象となる仮想デバイスを管理するサーバは，アプリケーションから与えられたデータを，当該仮想デバイスを実現する実デバイスから出力する．このとき，他のサーバが管理する実デバイスからの出力が必要な場合，出力データを当該サーバに転送し，出力を依頼する．

8 評価

8.1 音声出力デバイス

複数の異なる場所に設置されている音声出力デバイス使用する場合，各デバイスからの音声出力時刻には注意が必要である．これは，音の伝達する速度が，計算機に処理速度と比較して遅いためである．

実際に，どの程度音声にずれが生じるかを確認する実験を行った．本実験では，スピーカから音声を出力可能な 2 台の計算機上で，それぞれサーバプログラムを実行させている．また，これらとは別の計算機上で音声出力要求を出すアプリケーションを動作させる．ただし，これらの計算機は，Pentium 4 (2.8 GHz) を持つ PC で，1000 Mbps のイーサネットで接続されている．サーバは，アプリケーションからの音声出力要求を受けると，自身が動作する計算機のデバイスを使用して，スピーカから音声を出力する．

2 台のスピーカとマイクをこの順で一直線上に配置し，2 台のスピーカから出力される音声をマイクで取得した．図 4 は，このときの得られた音声の波形データである．ただし，2 つのサーバに同時に要求を送れるよう，アプリケーションからのサーバへの要求にはブロードキャストを用いている．

マイクに近い方のスピーカからは 400 Hz の，マイクから遠い方のスピーカからは 3200 Hz のサイン音を出力している．図から分かるように，マイクで取得している音声は，始めはノイズのみであるが，続いて 400 Hz の波形が現れる．また，その後，3200 Hz の波形も加わっている．2 台のスピーカ間の距離は 2.6 m であり，音の伝達する速度が秒速 340 m であるとすると，2.6 m 伝達するのに要する時間は 7.7 ms である．また，400 Hz の波形と 3200 Hz の波形が観測されはじめる時刻の差も，これに近い時間である．ユーザが聞く音声のずれは，アプリケーションとサーバ間の通信の時間やサーバプログラムの動作する時間に比べ，スピーカの配置される位置により大きく影響されることが分かる．

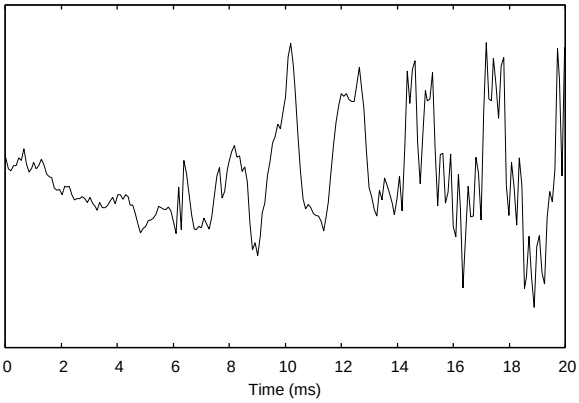

図 4　2 つのスピーカからの音声出力

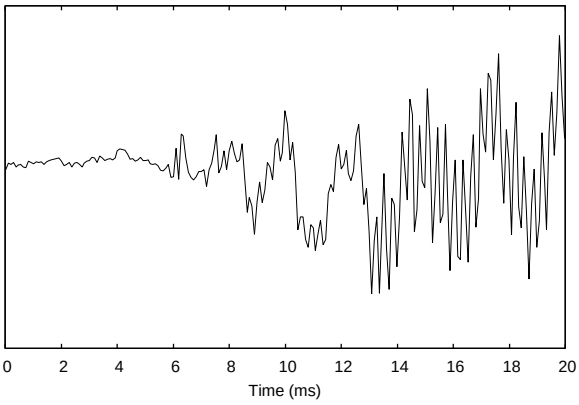

図 5　音声出力の時刻を調整した場合

最近の計算機の処理能力を考えると，音声を出力する時刻を調整し，ある程度，ずれを軽減させることは可能であると考えられる．図 5 は，同様の実験を，マイクに近いスピーカからの出力を 8 ms 遅らせて行ったときの結果である．400 Hz の波形あるいは 3200 Hz の波形のみが観測されるところはほとんどなく，ずれが軽減されていることが分かる．このように，音声の出力においては，デバイスの位置を管理することは重要で，これを適切に行うことによって，ユーザに届く音声のずれを軽減させることも可能である．

8.2 映像入力デバイス

デバイスからの入力処理に関しては，入力されたデータがそれを処理するアプリケーションに到達す

表 1　映像入力の処理に要する時間

入力方法	処理時間
(1)	3.18 ms
(2)	3.93 ms
(3)	6.54 ms
(4)	4.00 ms

るまでにかかる時間を短く保つことが重要である．特に，本システムのようなミドルウェアを使用する場合，そのオーバヘッドには注意を払う必要がある．

本システムのオーバヘッドを調べるために，USB 接続のデジタルビデオカメラからの入力処理に関して，次の 4 つの場合について処理時間を測定した．

(1) 本システム使用せずに直接入力する．
(2) ローカルの計算機上で動作するサーバを使用して入力する．
(3) リモートの計算機上で動作するサーバを使用して入力する．
(4) リモートの計算機上で動作するサーバのバッファから入力する．

これらの処理時間の表 1 に示す．ただし，実験には前節で用いたものと同様の環境を使用している．また，入力されるフレームのサイズは，230,400 バイトである．

(1) は，カメラからフレームを取得するのに要する時間そのものである．一方，(2) は，同様の処理を本システムを介して行ったものであり，(1) と (2) の差 0.75 ms がサーバとライブラリのオーバヘッドであると考えられる．

(3) では，リモートの計算機に接続されたカメラからの入力を行っている．アプリケーションの動作は，(2) と同様であるが，ライブラリとサーバ間でネットワークを介した通信が発生する．(2) と (3) の処理時間の差 2.61 ms がこの通信に要する時間であると考えられる．

(2) と (3) の処理時間の差，すなわち，フレームをローカルのカメラから取得する場合とリモートのカメラから取得する場合の処理時間の差が比較的大きい．この時間差は，カメラから取得したフレームを記憶するバッファをサーバ内に用意することで，軽減することができる．すなわち，サーバは逐次カメラからフレームを取得しこれをバッファ内に保持し，アプリケーションからフレーム取得要求があったときは，このバッファの内容を返す．(4) は，この方法によって入力を行った場合である．バッファへのアクセスには排他制御が必要になるため，完全には独立して処理することはできないが，多くの場合，処理時間にはカメラからフレームを取得する時間は含まれない．このような方法を用いることによって，ローカルのカメラから取得する (2) に近い処理時間で，リモートのカメラからフレームを取得することが可能になる．

9　おわりに

本稿では，分散環境における映像・音声入出力デバイスの管理に要求される機能について考察し，また，それを実現するデバイス管理手法について述べた．本手法では，実際のデバイスを抽象化し，仮想デバイスを実現する．仮想デバイスを用いることによって，デバイスを共有することや，デバイス操作を位置透過に行うことが可能になる．また，複数のデバイスを組み合わせ一括して操作するができるため，アプリケーションは多数のデバイスを簡単に操作することができる．

参考文献

[1] Object Management Group: *The Common Object Request Broker: Architecture and Specification — Version 2.6*, Framingham, Massachusetts (2001).

[2] Levy, E. and Silberschatz, A.: Distributed File Systems: Concepts and Examples, *Computing Surveys*, Vol. 22, No. 4, pp. 321–374 (1990).

[3] Callaghan, B., Pawlowski, B. and Staubach, P.: RFC 1813: NFS Version 3 Protocol Specification (1995).

[4] Davis, P., Gunther, R., Olofson, D., Sennoner, B. and Vehmanen, K.: Jack audio connection kit, http://jackit.sourceforge.net/.

[5] 永田悠, 立蔵洋介, 猿渡洋, 鹿野清宏: 音場再現システムにおける環境変化に適応的な逆フィルタの逐次的緩和アルゴリズム, 電子情報通信学会論文誌, Vol. J86–A, No. 8, pp. 824–834 (2003).

[6] 徳元大輔, 池戸丈太朗, 金子孝夫, 片岡章俊: IP ネットワーク経由で配信される音響信号のための同期再生技術, 電子情報通信学会論文誌, Vol. J87–D–II, No. 9, pp. 1870–1883 (2004).

[7] Video for Linux resources, http://www.exploits.org/v4l/.

[8] Advanced Linux Sound Architecture, http://www.alsa-project.org/.

[9] Open Source Computer Vision Library, http://www.intel.com/technology/computing/opencv/index.htm.