RSNP 拡張によるロボット制御と音声通信の統合のための ロボットサービスプラットフォーム

大澤 秀也* 朝倉 健介* 小原 範子* 藤田 尚宏*
佐藤 健* 中川 幸子* 成田 雅彦*

Robot Service Platform for Integration of Voice Communication with Robot Control by RSNP Extended

Hideya Osawa*, Kensuke Asakura*, Noriko Obara*, Naohiro Fujita*,
Ken Sato*, Sachiko Nakagawa* and Masahiko Narita*

Robot common platform, web-robot and cloud robotics are becoming active area of research. On the other hand, although voice communication is key to service robots in the life space, existing platforms are insufficient to the problem. In this paper, we propose integration of voice communication and robot control into the robot service platform by extending RSNP (Robot Service Network Protocol). This voice communication realize conversation with a target via a robot. Voice data are continuously transmitted by bidirectional communication. Moreover, since this service is integrated into RSNP, robot operation, image transfer and voice communication are implemented on a single platform. The use of this platform enables to develop the services easily and efficiently, without the expertise of voice communication. Furthermore, system-integrators are able to construct a robot system, without peculiar equipment for voice communication. As a result, such effective robot services as monitoring/watching in the fields of welfare and care easily provided, which can be brought to market inexpensively and quickly. We also report the results of our implementation and verify the effectiveness of our voice communication service on the robot service platform.

Key Words: RSNP, RSi, Robot Services, Network, Internet, Voice

1. はじめに

ロボットは公共施設や商業施設などの公共空間や、家庭・職場などの生活空間への導入がはじまりつつあり、人と共存するロボットの発展や、福祉・介護サービス分野でのロボット技術の活用が期待されている。ロボットの実用化には、様々なサービスを多くの利用者に提供できるよう、サービスアプリケーションやコンテンツの充実が必須である。そのため、2004 年ごろから、ロボットの共通プラットフォームの開発・普及のための様々な取り組みが行われてきた。ロボットのソフトウェアプラットフォームの試みとして、RT ミドルウェア [1] や次世代ロボット知能化技術開発プロジェクト [2], RSi (Robot Service initiative) [3], ROS (Robot Operating System) [4], UNR-PF (Ubiquitous Network Robot Platform) [5] 等の研究成果があ

る。また、クラウドとの融合の視点では、Cloud Robotics [6] [7] や RoboEarth [8] などの研究が進められており、RSi-Cloud [9] も提案されている。

一方、音声通信技術では、Skype [10] や LINE [11] に代表されるように、デバイス端末上のインターネットサービスアプリケーションとしての利用が広く普及している。このような近年のインターネットの普及に伴い、相互接続、および、クラウド適用によるサービス提供の重要性は増している。ここで音声通信を定義するとすれば、音声メッセージをネットワークを介して送受信する技術である。既存のロボットプラットフォームでも音声の扱いに関連する様々な研究は行われているが、ネットワークを介した音声通信をロボット制御と統合するための考察が不足していた。

このような背景のもと、我々は、ロボットサービス基盤へのリアルタイム音声通信の統合を提案する。音声通信は、2004 年に設立されたロボットサービスイニシアチブ (Robot Service initiative: RSi) によって策定されたオープンなロボットサービス向けプロトコルである RSNP2.3 仕様 [12] の拡張により実現する。RSNP は、ロボットサービスとインターネットサービス

原稿受付 2013 年 12 月 9 日

* 公立大学首都大学東京産業技術大学院大学

* Graduate School of Industrial Technology, Advanced Institute of Industrial technology

■ 本論文は有用性で評価されました。

の融合のためのロボットソフトウェア基盤であり、インタフェースは標準の WSDL で定義されているため、インターネットと整合性の高いロボットサービス開発が可能である [9] [12]。さらに、インターネット環境へ適用可能なロボットのための音声通信技術を、サービスロボット分野で有望とされている見守りサービスや接客サービスなどへ適用した場合、生活空間に適応する実用的なサービスへの展開が期待できる。よって、プラットフォームへの音声通信技術の統合は、より付加価値の高いロボットサービス提供を可能にする。

以下、2 章で関連研究について述べ、3 章でロボットサービス基盤へリアルタイム音声通信技術を統合するための提案を行う。4 章で提案する音声通信技術を利用したプロトタイプシステムを実装する。5 章では、プロトタイプシステムを利用した評価実験とアンケート調査を実施し、本提案の有効性の評価を行う。6 章をまとめとする。

2. 関 連 研 究

近年のロボットサービスのためのプラットフォーム分野としては、複数の要素技術を集約し、統合プラットフォームによる開発環境を提供する動きが活発化している。本章では、ロボット視点の統合、ロボットへのインターネット技術の適用の関連研究について述べ、課題を明らかにする。

ロボット視点の統合においては、RT ミドルウェアプロジェクト [1]、RT ミドルウェアに対応した画像や音声認識、運動制御モジュール開発の次世代ロボット共通基盤開発プロジェクト、知能化技術部品である RT コンポーネント開発の次世代ロボット知能化プロジェクトがある [2]。また、WillowGarage 社が開発し、現在は OSRF (Open Source Robotics Foundation) によって運営される ROS (Robot Operating System) [4] では、豊富なライブラリ群と開発ツールが提供されている。両者とも、ハードウェアとしてのロボットやロボット部品、および、ロボット部品間の連携を対象としたプラットフォームである。ロボットにおける音声の取り扱いの視点では、ロボットと人の相互作用のためのサービスロボット研究が行われており、RoIS (Robotic Interaction Service Framework) [14] では、音検出、音源定位、音声認識、音声合成、発話の研究が進められている。また、特徴量検出によるロボット聴覚の HARK [15]、ロボットと人との対話システムの研究の対話認識行動プラットフォーム [16] や、OpenHRI [17] があり、ロボットによる人の声の聞き分けや、ジェスチャによる人とロボットの対話行動の研究が行われている。いずれも、インターネットを介したリアルタイム音声通信の統合は対象とはしていない。

一方、インターネットを介したロボットサービスとして、ロボット技術のクラウドへの適用がある。Cloud Robotics を推進する Google では、同社の認識エンジンと GraspIt! を利用したクラウドベースのロボット把持 [18]、Google/IO 2011 における ROS-java を搭載した Android-OS の携帯端末とロボットとの連携デモンストレーション [19] がある。RoboEarth では Cloud Robotics のための WebSocket プロトコルを利用したフレームワークとして rapyuta [20] を提供する。いずれも、ロボットのローカル環境では高負荷な複雑なタスク計算をクラ

ウド側で実行することに関心の対象がある。また、ROS では、WillowGarage 社によるインターネットを介したリモートロボットの Texai [21]、リモートロボットパッケージの rosbridge [22] や、JavaScript によるロボット用 Web アプリケーションのためのツール開発 [23] への取り組みがあり、インターネットを利用したロボットサービスの提供モデルの研究が進行している。

しかし、従来、インターネットを介したロボットサービスにおいて、双方向のリアルタイム音声通信を伴うサービスを提供する場合には、ロボットシステムとは別法にて SIP/VoIP 等の他の音声通信システムを構築し、2 系統のシステムを用意することで対応されている。この場合の環境構築では、ロボット制御用と音声通信用の二つの通信路、およびサーバインフラなどを用意し、ロボットサービスアプリケーションと音声アプリケーションは個別に設置されることになる。さらに、この場合では、統合基盤としてロボット制御と音声通信が協調するようには定義されていないため、セッション管理や障害発生時の手続きは煩雑になり、通信回線のセキュリティ設計も別途に行う必要がある。よって、サービス開発や構築、運用におけるコストの増大を回避し難い。このような課題の解決のため、インターネット環境へ適用可能な音声通信のロボットサービスプラットフォームへの統合を提案する。

3. 設 計 計

本稿では、音声通信機能を備えた見守りサービスの実現を想定し、音声通信をロボットサービス基盤へ統合したサービスプラットフォームを実現する。

3.1 設計指針

本提案にあたっては、以下の設計指針を策定した。

- 通信プロトコルに RSNP を適用する。この RSNP 仕様を拡張することにより、インターネット経由のロボットサービス提供において、2 本の経路により実現されているロボット制御と音声通信の通信路を 1 本の経路にするプロトコルを実現する。これにより、ロボットの画像配信と遠隔操作制御に、音声通信を統合したプラットフォームを実現する。
- 音声通信は、会話が可能なレベルの双方向通信を目指す。さらに、多様な環境でのサービスへ容易に適合するため、サービスやネットワーク回線状況に応じた音声データ送受信の設定値を調整可能とする。

設計指針により実現されるプラットフォームの構成を Fig. 1

Fig. 1 A system architecture of the platform

Fig. 2 RSi robot service model Ver.2.3

に示す。ロボットは、RSNP サーバ経由で RSNP 通信を行う。ロボットで実行される制御は、RSNP クライアントとして実装され、サービスで実行される処理は RSNP サーバとして実装される。提案するプラットフォームは、インターネット経由でロボットとサービスが通信を行うための RSNP 通信層と、音声通信を実現するための機能群であるサービス層から構成される。プログラマは、プラットフォームが提供する機能群を利用し、ユーザアプリケーションを開発する。

以下、3.2 節に、本稿で利用する RSNP について詳述する。3.3 節では、RSNP 通信層の拡張について述べ、3.4 節では、サービス層の機能群となる音声通信モジュールについて述べる。

3.2 RSi と RSNP

本節では、RSi のロボットサービスモデルと、拡張対象である RSNP の特徴を述べる。

3.2.1 RSi のロボットサービスモデル

RSi では、ロボットサービスを、ネットワークを介してロボットが提供する情報サービス、もしくは物理的サービスと定義している。Fig. 2 にサービスモデル Version2.3 の概要を示す。このモデルは、ユーザ、ロボットや小型ロボットを含めたモバイル端末、ロボットサービス、サービスフロントなどから構成される。また、同期・非同期の通信による動作や動作パターンの指示や結果の取り出し、ロボットからサービスへの問い合わせ・通知、サービスの提供、ユーザを含む環境とのやり取りを行うことができる。また、クラウド間連携や、データ蓄積、クラウドに配備されるロボットサービス特有の仕組みについても考慮されており、Version3 として仕様策定が進められている。RSi では、ロボットをインターネットに接続する利点は、人と協調してサービスを行う「協調型」ロボットを実現できること、インターネット上のコンテンツを再利用できることであり、RSNP は、このモデルに従ってサービスのプロトコルを規定しているため、異なるベンダで独立して開発したロボット／サービスの間での相互運用が実現できる。

3.2.2 RSNP の特徴

RSNP の特徴は、下記のとおりである。

- インターネットとの整合性が高く、標準化された豊富な機能の上にサービスを構築でき、多様なロボットと各種サービスに適用できる。さらに、同一仕様で異なる実装間の相互接続が可能である。

Fig. 3 RSNP2.3 platform architecture

- ロボットらしいサービスを提供できる。例えば、ロボットを利用した継続的な見守りなどがある。
- 計算機資源の限られているロボットへ、情報提供や指示が迅速に行える。
- 種々のロボットプラットフォームとの連携が実現できる。

RSNP の現段階の通信層の実装は、インターネットやシステム構築向け通信基盤である Web サービス基盤を利用している。そのため、高信頼メッセージング機能、セキュリティ機能等、インターネットと整合性の高い標準化された機能を利用可能である。RSNP のシステムアーキテクチャを Fig. 3 に示す。

プロトコル自身は共通サービスとロボットサービスから成り、共通サービスは、各種サービスをサポートするために、Pull 型、Push 型、同期・非同期型の通信モデルを提供している。一方、ロボットサービスは基本プロファイルと応用プロファイルから成り、ロボット機能を提供する基本プロファイルとして、カメラ・音声入出力などのマルチメディア機能を実現する Multimedia プロファイルや、前後回転動作など単純な動作を実現する Motion プロファイル、パターン動作を提供する MotionPattern プロファイル、サーバ内の画像やテキストなどのコンテンツ情報をロボットに送信する Content プロファイルなどがある。サービス機能を提供する応用プロファイルとして、情報サービス・天気サービス・防災情報サービス・見守りサービス・リモート制御などのサービスを提供する。このように、ロボットの機能とロボットサービスをモジュール化して提供することで、機能の異なる多様なロボットや各種サービスへの適用を可能としている。

3.3 RSNP 通信プロトコルの拡張

本節では、3.1 節で述べた一つめの設計指針に従い、RSNP へ音声通信を統合する上で、RSNP 通信層のプロトコル拡張を行うための課題を抽出し設計を行う。

3.3.1 RSNP 通信プロトコルの拡張

一つめの設計指針を実現し、通信プロトコルとして RSNP を適用するにあたっては、RSNP 通信モデルを選択する必要がある。RSNP では、通信起点と応答パターンによる双方向通信のため、Fig. 4 に示す六つのメッセージパターンが規定されている [12]。本稿は、この中から、⑥のサーバ起点ロボット配信型と、③のロボット起点サーバ配信型を採用する。前者 ⑥のサーバ起点ロボット配信型は、サーバからの 1 要求メッセージに対してロボットから 1 回の要求受付結果応答メッセージと複

Fig. 4 RSNP message patterns

数回の実行結果応答メッセージを返却する。後者 (3) のロボット起点サーバ配信型は、ロボットからの 1 要求メッセージに対してサーバから 1 回の要求受付結果応答メッセージと複数回の実行結果応答メッセージを返却する。これにより、ロボットとサーバ間の通信回数を最小にし、通信負荷を低減することができる。これについては、3.3.2 項にも詳述する。

さらに、2 本の経路により実現されているロボット制御と音声通信の通信路を 1 本の経路にするプロトコルの提案にあたっては、RSNP で用意されている画像を連続して転送するプロトコルを拡張し、音声データのリアルタイム通信のオペレーションを追加する。従来、RSNP では、Multimedia プロファイルに音声入出力の仕組みが用意されているが、マイク、およびスピーカへの制御、サウンドの録音と再生・停止の指示を目的としており、音声通信を前提としたものではない。本稿は、遠隔操作可能なロボットによる見守りサービスを前提に、オペレータが、見守り対象の音声と画像情報を取得し、必要な時には見守り対象への声がけや、オペレータと見守り対象の簡単な会話を行うための通信方式の提供を最低要件とし、サービス実現のための必要最低限のデータを取り扱うのに適したプロファイル拡張の対象とする。

3.3.2 プロファイルの拡張

本稿では、データ送受信を前提としたモジュールである Multimedia プロファイルと Contents プロファイルを拡張し、音声データの取り扱いを可能とする。以降、各プロファイルをメソッド名にて記載する。Multimedia_profile では、3.3.1 項に記載のとおり、音声再生のため仕組みは定義されている。さらに、サーバからの要求を契機に、ロボットから画像を連続的に非同期にサービスに転送する仕組みが `distribute_camera_image` のオペレーションとして定義・実装されている。これは、3.3.1 項に述べたサーバ起点ロボット配信型の通信パターンを用いて、サーバからの到着応答を待たずに連続的に送信することができる。Contents_profile では、ロボット側で利用するテキストや画像などのファイルを、連続的にサービス側からロボットへ転送する仕組みが `distribute_contents` オペレーションとして定義されている。これは、同じく 3.3.1 項に述べたロボット起点サーバ配信型の通信パターンを用いて、ロボットからの到着応答を待たずに連続的にデータを送信することができる。よって、ロボットからサービスへの通信（上り通信）では画像と音声デー

Fig. 5 Voice data structures of RSNP extended

タの同時送付を可能とし、サービスからロボットへの通信（下り通信）では音声データのみを送付することが可能となる。

拡張にあたっては、上り通信では Multimedia_profile のオペレーションである `distribute_camera_image` を拡張し、ロボットのマイクから取得した音声を、音声バイナリデータに変換後、`attached_files` として添付する。このとき、複数のデータ添付を可能とすることで、画像と音声データの同時送付を可能とする。下り通信では、Contents_profile のオペレーションである `distribute_contents` を拡張し、同じく `attached_files` として添付することで音声データの送付を可能とする。音声データの格納構造を Fig. 5 に示す。このように、画像と音声の添付データを、一つのプロトコル上で扱うことにより、RSNP 上のロボット制御における音声情報と画像情報を、同時に取り扱うことを可能とし、1 本の経路でロボットサービスの構築を可能とする。

3.4 音声通信モジュール

本節では、3.1 節で述べた二つめの設計指針を実現し、RSNP で双方向音声通信を実現するための音声通信モジュールの設計を行う。

3.4.1 下位通信層

ロボットサービス上で音声通信を行うにあたっては、下位通信層に RSNP の実装で用いている TCP/HTTP を採用する。この場合、RSNP の疑似 Push 通信を利用することで、Firewall 環境下に設置されたロボットへのインターネット側からのロボット制御が可能である。従来、音声通信では UDP が利用されることが多いが、インターネットで実際に使用する上では、Firewall 上のポート解放や、NAT を超えるための処理を行う必要がある。この解消には、STUN/TURN サーバなどの別途の配備が必要となり、統合の利点が希薄となる。よって、本稿は TCP を採用する。

しかし、TCP では回線品質が低下した場合に再送が発生する。この TCP による再送の発生により、音声取得の実装方式によっては、遅れが蓄積する可能性がある。さらに、音声再生時の無音区間や音声の欠落が発生し、音声品質が低下する。よって、3.3.2 項に示した `attached_files` で送受信する処理のみでは、リアルタイムの連続的な音声出力を行うには困難がある。さらに、従来、RSNP 通信では、データ受信のタイミングでユーザアプリケーションを呼び出すコールバック制御の仕組みを持つが、このコールバック制御のみで、音声のような連続データ送受信のタイミング処理を行うのは、アプリケーション開発者に

Fig. 6 System architecture of a voice service

として難易度が高い。そこで、音声伝達の遅れへの対応として、ロボットとサーバのおおのにキューイング機構を設ける。このキューイング機構により、ロボット側 (RSNP クライアント) から音声データを送信するタイミングで最新のデータのみを送信し、サーバ側 (RSNP サーバ) も最新のデータのみを取得することができるため、遅れを減少することができる。さらに、これにより、データアクセスへのタイミング要件が緩和されるため、音声再生の実装が容易になる。また、音声品質の低下への対応として、サービスや回線状況に応じて、音声取得間隔、データ転送間隔、最大キュー数の三つの設定値について調整が可能な仕組みを導入する。これは、多様な環境へ適合するための調整を可能とするプラットフォーム形成に寄与するものである。

3.4.2 システム構成と機能

音声通信モジュールは、提案するプラットフォームを実現する上で必要なライブラリ群である。音声通信モジュールを配備したシステム構成を Fig. 6 に示す。以下にRSNP クライアントとRSNP サーバの、それぞれの音声モジュールの機能について詳述する。

- (1) **Voice Client Module:** RSNP クライアント側の音声モジュールである Voice Client Module には、送信キューとして SoundQueueRec、受信キューとして SoundQueuePlay が配備される。上り通信では、アプリケーションがロボット付属のマイクで取得した音声から、音声取得間隔に応じたバッファデータを連続して生成し、SoundQueueRec へ投入する。SoundQueueRec では、指定した音声間隔で取得したデータを、要素 (element) として管理しており、蓄積されるバッファが、最大キュー数を超えた場合、最も古い element を破棄する処理を行うことで、音声伝達の遅れを減少している。データ転送は、RSNP 通信により定期的に行われるが、このとき、SoundQueueRec に蓄積されたすべての element を結合した上で、RSNP パケットへ格納し、送出する。下り通信では、受信した RSNP パケットから音声データを取り出し、SoundQueuePlay へ投入する。ロボット上のユーザアプリケーションは SoundQueuePlay を呼び出し、デコードして音声を再生する。

```
//revd_ret_value : Objects are callback from RSNP
//0 : Image, 1 : Voice

public class ImageCallback extends AsyncReceiver implements IAsyncCallback{
    ...
    public void doEvent(Ret_value ret, boolean isLast){
        if(revd_ret_value==null ){
            revd_ret_value=ret;
            if(revd_ret_value.get_file_count() > 1){
                AttachedFile attfile =
                    revd_ret_value.get_attached_file(1);
                byte[] data = attfile.get_byte_array();
                SoundQueue queue = SoundQueue.getInstance();
                if(data != null && data.length != 0){
                    queue.add("tomcat", data);
                }
            }
        }
        else {
            try{Thread.sleep(5);catch(Exception e){};
        }
    }
}
```

Fig. 7 Voice Service Module is picking up a received voice data in RSNP packet from ROBOT and enqueues it to Sound Queue during a callback process

- (2) **Voice Service Module:** RSNP サーバの音声モジュールである Voice Service Module には、上り通信用の音声キューとしての SoundQueue と、下り通信用のキューである ContentsQueue が配備される。上り通信では、ロボットから送出された RSNP パケットを受信した Voice Service Module は、RSNP のコールバック処理の中で、取り出された音声データを SoundQueue へ格納する (Fig. 7)。このとき、到着した RSNP パケットの 2 番目の添付ファイルを音声データとみなし、SoundQueue へ格納することで、音声データのみの取り扱いを可能としている。なお、1 番目のデータは、画像データとして ImageQueue へ格納される。SoundQueue へ格納された音声データは、すべての element を結合する。サーバ上のユーザアプリケーションは、poll インタフェースを用いて、SoundQueue から音声データを取り出す。Fig. 8 の例では、ユーザアプリケーションは、ブラウザへ音声データを転送している。下り通信では、Web アプリケーションから取得した音声データを、Contents Queue に蓄積し、RSNP 通信によりロボットへ送出する。

```

class ToClientTimerTask extends TimerTask {
    @Override
    public void run() {
        AudioToClient(queue.poll(robot_id), false);
    }
}
private void AudioToClient(byte[] voiceData, boolean filterOn) {
    try {
        ByteBuffer voiceDataBufferDst;
        if (voiceData.length > 0) {
            ByteBuffer voiceDataBufferSrc =
                ByteBuffer.wrap(voiceData);
            voiceDataBufferSrc.order(ByteOrder.LITTLE_ENDIAN);
            talkBinary(voiceDataBufferSrc);
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}
}

```

Fig. 8 An example of user application uses SoundQueue's poll interface to obtain voice data

Table 1 SoundQueue class methods (in Java language)

Method	Function
static SoundQueue getInstance()	Return an instance of a singleton.
void add(String robotId, byte[] sound)	Add a voice data in the queue. Key is the robot id.
byte[] poll(String robotId)	Get voice data from the queue based on the robot ID.
int getSize(String robotId)	Return the size of the queue based on the robot ID

(3) *SoundQueue*: 音声通信機能のための音声通信モジュール内の *SoundQueue* では、音声データ格納共通クラスとして *SoundQueue* を配備した。 *SoundQueue* クラスの実装には、Singleton パターンを利用する。Singleton パターンを利用することで、ブラウザからサーバへのアクセスと iPhone へのアクセスが、異なるセッションで接続され、それらから同一の音声データにアクセスすることが可能となる。 *SoundQueue* クラスのメソッドを Table 1 に示す。 *SoundQueue* では、 *getInstance()* メソッドを呼び出すことによって単一インスタンスを取得する。この *getInstance()* メソッドにより取得されたインスタンスを変数へ格納し、各メソッドを呼び出す。 RSNP プロトコルでは、複数のロボットを制御するために、個別にロボット ID が割り当てられており、サーバ側では、接続した複数のロボットをロボット ID で区別する。よって、Singleton のキューの中で、ロボット ID により個別にロボットの音声データが管理されているため、ロボットの複数接続の利用にも対応できる。

4. プロトタイプシステムの実装

本稿の音声通信モジュールにより実現するプロトタイプシステムとして、音声通信機能をもつ見守りサービスを構築した。本章では、その結果を報告する。

4.1 プロトタイプシステムの概要

プロトタイプシステムでは、ロボット制御と音声通信を統合したサービスを実装する必要がある。そのため本稿では、外出先から自宅の状況を確認し、在宅の高齢者や子供やペットなどの見守りと、見守り対象とのコミュニケーションを可能とする

Fig. 9 A prototype system

ロボットによる見守りサービスを実装した。サービスのイメージを Fig. 9 に示す。

4.2 システムの実装

システムは、RSNP クライアントとなるロボットとそのソフトウェア、RSNP サーバ上のロボットサービスアプリである見守りサービスと、見守りを行うオペレータとその前の端末から構成される。以下に、各構成要素について述べる。

4.2.1 ロボットとソフトウェア

ロボットは、カメラによる連続画像取得と転送を行う。さらに、オペレータの遠隔操作により見守り対象空間を移動し、見守り対象（ユーザ）や周辺の音声を取得し、ユーザとオペレータで会話ができる音声通信機能を持つ。本実装では、ロボット本体には、スマートペットとそれに接続した iPhone を用いた。ソフトウェアは、RSNP プラットフォームを搭載した iPhone のアプリケーションである RSiCameraRobot を拡張し、Voice Client Module と、これと連携して音声の入出力を行うアプリケーションを実装した。

以下に、実装したロボットとソフトウェアについて詳述する。

(1) *RSiCameraRobot*: ここで、本稿の実装に用いた RSiCameraRobotkiyou12 について詳述する。 RSiCameraRobot サービスロボットを持たない多くの開発者が容易にロボットサービスを開発できることを目的とした iPhone アプリケーションであり、AppStore で配布されている。スマートフォンは、カメラ、加速度センサ、磁気センサ、明暗センサ、マイク、Wi-Fi、Bluetooth 通信機能を持ち、物理動作を除けばロボットと共通点が多い。さらに、 RSiCameraRobot は、 RSNP の通信基盤と共通サービスである Multimedia_profile の distribute_camera_image で実装されているため、サーバ上の見守りサービスアプリケーションと接続して、カメラの前の様子を遠隔からモニタリングすることができる。

本システムでは、スマートペットを制御するため、この RSiCameraRobot に、 RSNP のロボットの動作制御のためのプロファイルである Motion_profile と、3.3.2 項に詳述した Contents_profile を追加で実装した。我々は、これを RSiCameraRobot Ver.2 と呼んでいる。本サービスの

画像取得の性能は、携帯電話の動画通話に一般に用いられる QVGA (320×240 ピクセル) やその半分 (240×160 ピクセル) の解像度、フレームレートは、iPhone5 で 23 [fps] 程度である。また、スマートペットの制御には、スマートペット SDK [25] を利用した。

- (2) *Voice Client Module* と音声処理: 音声通信の実装にあたっては、RSiCameraRobot Ver.2 の上に、Voice Client Module を 3.4.2.1 の設計に基づき Objective-C で実装した。また、これと連携して音声の入出力を行うアプリケーションを iOS の Audio Unit Service を用いて実装した。指定した間隔で音声取得されると AudioInputCallback が呼ばれるので、その中で、Voice Client Module とのインタフェースである SoundQueueRec に音声データをキューイングする。音声出力は、再生中のデータを再生し終わると、AudioOutputCallback が呼ばれ、その中で、Voice Client Module である SoundQueuePlay からデータを取得して、音声の再生を続ける。音声の取得間隔、Voice Client Module の最大キュー数は、このアプリケーションにて設定する。

4.2.2 サーバ上のロボットサービス

以下に、実装したサーバ上のロボットサービスについて詳述する。

- (1) 見守りサービス: 本システムには、連続画像取得、遠隔操作、音声通信の機能がある。4.2.1 項に述べたようにロボット側には RSNP の Multimedia_profile の distribute_camera_image と、Contents_profile の distribute_contents と、Motion_profile が実装されている。これに対応するサービスを、品川区の本学内に有線 LAN 接続にて設置された Mac mini 上に構築した。これには、RSNP プラットフォームのサーバ機能実装を提供する RSNP の実装ライブラリである FJLIB を利用した。

連続画像取得には、Multimedia_profile の distribute_camera_image と連携して画像表示を行う見守りサービスアプリケーションを利用した。遠隔操作では、Motion_profile と連携して Web ブラウザからの遠隔操作を可能とした遠隔操作サービスを利用した。ここでは、Web ブラウザ上の「前進」「後退」「右」「左」「停止」のボタンにより Motion_profile の各オペレーションを呼び出し、RSNP サーバ経由でロボットの前進、後進、左右回転、停止の動作を行うことができる。音声通信では、Contents_profile と連携して、連続データの送信を行った。これらのサービスアプリケーションの利用、および拡張を行い、本稿の見守りサービスを実装した。

- (2) *Voice Service Module* と音声通信アプリケーション: サーバ上の音声通信の実装には、前節の RSNP ライブラリの上に、Voice Service Module を 3.4.2 (2) の設計に基づき Java で実装した。また、これと連携してサーバと端末とで通信を行いさらにブラウザから音声の入出力を行うため、ブラウザ側とサーバ側のおおのにアプリケーションを実装した。

ブラウザからサーバの通信（下り通信）では、ブラウザ側アプリケーションは、ブラウザのマイクから音声を取得

Fig. 10 An image of browsing screen of operator side

し、ダウンサンプリングのためのビットレート変換を行い、変換後の音声データを RSNP サーバへ送信する。この時の音声取得間隔は、ブラウザの音声用のバッファの長さにより決まる。サーバ側アプリケーションでは、そのデータを Voice Service Module の poll インタフェースで、Contents_queue に投入する。サーバからブラウザの通信（上り通信）では、サーバ側のアプリケーションは、SoundQueue を定期的に呼び出し、ブラウザへ音声を送出する。ブラウザ側アプリケーションは、再生キューに音声データを一定量まで蓄積し、再生形式に変換して出力する。なお、サーバとブラウザ間の通信は WebSocket を用いた。ブラウザの音声機能には、WebAudioAPI を利用し、ブラウザ側のアプリケーションは、JavaScript で実装した。Web ブラウザには、GoogleChrome28.0.1500.95 を利用した。

4.2.3 音声設定インタフェース: 音声設定値の調整

音声通信モジュールに対し、サービスや回線状況に応じた容易なパラメータ調整を可能とするため、音声取得間隔とデータ転送間隔を、オペレータ側ブラウザから調整できるようにした。オペレータ側のブラウジング画面を Fig. 10 に示す。Browser Buffer ボタンは音声取得間隔 (ms) の変更に対応している。Span ボタンはデータ転送間隔 (fps) の変更に対応している。なお、最大キュー数 (個) については、現実装では、プログラムの中で設定変更を行っている。

5. 評価

本章では、前章までに述べたプロトタイプシステム実装を用い、5.1 節で、評価実験を行う。5.2 節では、評価実験結果を用いて、複数台のロボット制御通信を行うにあたってのアプリケーション開発について考察を行う。

5.1 評価実験

本検証は実現する音声通信とロボットサービスとしての性能について 3 段階で検証する。音声の評価方法には、客観評価として音声伝達の遅れと、音切れなどのパケット欠損、および、揺らぎなどの指標がある。ロボットサービスで行うロボットを介

した会話の実現では、特に遅れが重要なので、検証1（音声の遅延）では、音声伝達の遅れを中心に評価する。その際、双方向の対話が重視される通信（IP 電話など）で採用されている ITU-T に G.114 [26] として勧告されている 400 [ms] の遅れを目標値とした。検証2（ロボットとの統合）では、RSNP へ音声通信を統合することにより、音声通信の、画像転送とロボット制御通信への影響を検証する。さらに、画像サイズを変更した場合の、音声通信の遅れへの影響を評価する。検証3（主観的評価）では、第三者に音声システムを体感してもらうことによる主観的評価を行う。主観品質は、音声や映像といった視聴覚メディアを用いた通信サービスにおいて、品質指標として用いられている。この評価の最大の特徴は、利用者の感じる品質を直接測定する点にあり、音声サービスの主観品質を評価するうえで信頼できる手法といえる。

以下に、各検証について詳述する。

5.1.1 検証1（音声の遅延）

本検証では、上り通信（Uplink）、下り通信（Downlink）のそれぞれについて、音声の遅れを計測する。検証環境は、東京都品川区の本学内（上り 100 [Mbps] / 下り 100 [Mbps] の光回線で契約）に設置されたサーバと、東京都中野区某所（上り 100 [Mbps] / 下り 200 [Mbps] の光回線で契約）に配備されたロボットおよび、ブラウザ閲覧端末で構成される。ロボット設置地点からサーバまでの直線距離は約 13 [km]、サーバからブラウザ端末設置地点までの直接距離は同じく約 13 [km] である。また、ロボットから無線アクセスポイントまでは、IEEE802.11a で接続される。遅れの計測には、音声の発話と再生を録音し、ツールを利用して音声データを波形として表示し、ロボットとブラウザの入力波形と出力波形の間隔のずれを取得する。ここでは、双方向の音声通信が行われており、上り通信には 3.5 [KB] 程度の連続画像転送、下り通信では遠隔操作指示であるロボット制御通信が、1 回/秒で行われる。音声のサンプリングレートは 8,000 サンプル/秒、16 [bit] の PCM を指定した。検証環境イメージを Fig. 11 に示す。

計測にあたっては、音声取得間隔、最大キュー数、データ転送間隔の設定値を変更する。音声取得間隔は、ロボット、および、ブラウザがマイクから取得する要素 element の大きさであり、音声取得間隔が大きい場合は、ロボットの送信キューに蓄積されるデータ量が大きくなるため遅れが増大する。一方、最大キュー数は、element の蓄積数の最大値であり、この値が小さいほど遅れは小さくなる。ただし、本提案では、蓄積数の最大値を超え

た場合は、element を破棄することで遅れを減少しているため、最大キュー数が小さい場合は破棄が増大する。よって音切れが増大する。なお、データ転送間隔は、distribute_camera_image のデータパケットの送信間隔である。

計測結果を Table 2 (A) (B) に示す。Case-1 では、音声の遅れは上り通信は 248 [ms]、下り通信は 203 [ms] の結果が得られた。Case-2 では、遅れは上り通信は 342 [ms]、下り通信は 297 [ms] の結果が得られた。Case-1 と Case-2 のいずれも 400 [ms] を下回っており、遅れは小さいといえる。ただし、Case-1 では、上りで 1 秒間に 0.57 回、下りで 1 秒間に 0.082 回の音切れが観測され、Case-2 では音切れは観測されなかった。また、Case-3 では、上り通信は 374 [ms]、下り通信は 325 [ms] の結果が得られた。Case-4 では、上り通信は 501 [ms]、下り通信は 420 [ms] の結果が得られた。Case-3 では遅れは小さいが、Case-4 ではやや遅れが大きくなった。ただし、Case-3 では遅れは小さいが、上りで 1 秒間に 0.06 回の音切れが観測され、Case-4 では遅れはやや大きい、上り下りとも音切れは観測されなかった。本検証により、適切なパラメータを設定することで、会話に適した 400 [ms] の遅れを下回る結果を得られることを確認した。

次に、転送画像サイズを変更し、音声の遅れへの影響が小さいことを確認する。ここでの設定値は、Case-2 を利用し、画像サイズを増加した。測定結果を Table 3 に示す。画像サイズを 50 [KB] にしても遅れは比較的小さく 400 [ms] を下回る結果を得ることができた。これにより Skype などの一般的な音声シス

Fig. 11 Left side of this photo is a pet-robot connected to RSNP server on the internet. Right side PC is showing our monitoring-service, connected to the server

Table 2 Measurement of delays of voice, images and robot-control signals transfer rate

	Settings			Measured value			
	Voice Sampling interval (ms)	Max Queue Number	Image transfer rate (fps)	(A) Voice transfer Delay Time (ms)/ Uplink	(B) Voice transfer Delay Time (ms)/ Downlink	(C) Robot control Delay Time (ms)/ (Downlink only)	(D) Image transfer rate (fps)
Case-1	46	2	21	248	203	128	17.4
Case-2	46	5	21	342	297	124	17.4
Case-3	92	2	10	374	325	150	10.6
Case-4	92	3	10	501	420	160	10.2

* RSNP client- iPhone5/ iOS7/ CPU:Apple A6 1.3 [GHz],

RSNP server- Mac mini/ OS X 10.9/ 2.3 [GHz] intel Core i7/ 8 [GB] memory

Table 3 Experiment of image data size change

Image Size (KB)	Voice transfer Delay Time (ms)/ Uplink
3.5	342
10.0	342
20.0	376
50.0	389

テムと同程度の画像サイズをプラットフォーム上で扱うことができることを確認した。

5.1.2 検証 2 (ロボットとの統合)

本検証は、検証 1 の環境を用いて、音声通信がロボット制御通信、および画像転送へ与える影響、すなわち、音声通信のプラットフォーム統合への影響を検証する。

ロボット動作制御通信は、Motion_profile のメソッドにより送出され、ブラウザのユーザインターフェースからのロボットへの動作制御の指示のタイミングと、iPhone 側からの制御信号の発行のタイミングのずれが動作制御の遅延時間に相当するため、おのこの制御信号の送信時間をログへ出力して計測し、その差分から遅延時間を測定する。設定値は、前節の検証 1 の環境に準じる。計測結果を Table 2 (C) に示す。Case-1 と Case-2 での遅れは 125 [ms] 程度、Case-3 と Case-4 での遅れは 150 [ms] から 160 [ms] 程度であり、ロボット制御への影響は小さいといえる。

画像転送については、distribute_camera_image により送出される。システムへの設定値としては、データ転送間隔を与えているが、ここでは実測値を取得する。計測結果を Table 2 (D) に示す。データ転送間隔、Case-1 と Case-2 では 17.4 [fps]、Case-3 と Case-4 では 10 [fps] の結果を得られ、見守りサービスでの連続画像としての違和感は小さい。なお、データ転送間隔を 21 [fps] に設定したにもかかわらず、17.4 [fps] となるのは、iPhone のハードウェア性能の限界のためと思われる。

よって、音声通信をプラットフォームへ統合しても、ロボット制御通信、および、画像転送を伴う見守りサービスなどで十分に使用できるといえる。

5.1.3 検証 3 (主観的評価)

2014 年 4 月に芝浦工業大学豊洲キャンパスで開催されたネットワークを利用したロボットサービス研究専門委員会 2014 年度 第 1 回研究会において、本稿についての実演デモを行い、主観的評価を実施した。実験参加者は研究者・教員 8 名、技術職 3 名、大学生 2 名の合計 13 名で、ロボット工学、ロボットソフトウェア等のロボット分野や、ネットワーク、IT、メカトロニクス等の専門家となる。

実験では、設定値の異なる以下の 2 種類の音声の評価を行った。

- 音声 1：音声取得間隔は、上り 46 [ms]、下り 92 [ms] とする。データ転送間隔は、上りに対し 21 [fps]、最大キュー数は 2 とする
- 音声 2：音声取得間隔は、上り 46 [ms]、下り 184 [ms] とする。データ転送間隔は上りに対し 4 [fps]、最大キュー数は 2 とする。

事前のローカル接続による検証により、音声 1 では遅延

Table 4 Evaluation results of the demonstration

	Voice1	Voice2
Delay of voice	3.4	1.9
Voice chopping	3.0	3.9
Quality for conversation	3.0	3.9

360 [ms] であるが音切れが 1 秒間に 0.1 回程度あること、音声 2 では遅延 613 [ms] であるが音切れは 1 秒間に 0.06 回であり、音声 2 は音声 1 に比べて、遅れは大きい音切れがないことを確認後、上りの音声を評価した。評価項目は、それぞれの音声の (1) 遅れ、(2) 音切れについて、段階 (5: まったく気にならない, 4: ほとんど気にならない, 3: どちらでもない, 2: 少し気になる, 3: かなり気になる), ③ 会話ができる品質として (5: かなり使える, 4: まあまあ使える, 3: どちらでもない, 2: ほとんど使えない, 1: まったく使えない) で評点を付けてもらい、平均値を求めた。

実験の結果を Table 4 に示す。この評価により、音声品質について会話が可能なレベルであることが確認された。さらに、会話のしやすさの向上を目指した遅れの少ない設定である音声 1 よりも、遅れがあっても音切れがないクリアに聞こえる音声 2 に、被験者の好評価を得た。これは、音声 2 では、会話に限らず適用範囲を拡大することが可能となるためである。音声 2 に対しても有用であるとの主観的評価により、プラットフォームにあたっては、設定値を可変とする柔軟な実装によって、サービスアプリケーションに合わせた通信路を提供することの有用性を確認した。

5.2 ネットワーク負荷の見積り

本検証では、5.1 節に基づいて音声通信を統合したロボットサービスの提供にあたり、通信に使用するデータ量から利用を想定される通信帯域占有率や、トランザクション量を見積もるための基礎データを提示する。また、公共のインターネットを使ってロボットサービスを提供する際には、ネットワークへ与える影響を軽減する必要がある。よって、測定結果から、アプリケーションが行うべき工夫について述べる。

5.2.1 ロボットとサーバ間の通信量

ロボットから無線 LAN によりアクセスポイントまで通信し、ロボットを無線 LAN (IEEE802.11a/b/g/n) でサービスへ接続するネットワークについて、5.1 節に述べたプロトタイプシステムを用いて通信データの見積もりのための基礎データを提示する。RSNP の要求パケットと応答パケットは、SOAP エンベロップとして 1 組の HTTP 要求と HTTP 応答として送受信される。音声データや画像データ等の添付がない場合の RSNP パケットの大きさは、SOAP の制御情報、HTTP ヘッダなどの制御情報など含めて 1.4 [KB] である。音声、および画像を添付した場合は、これに添付ファイルのサイズが付加される。RSNP のメッセージパターンは、Fig. 4 に示したとおりである。すなわち、上りの音声・画像データの送信は、1 HTTP 要求と 1 HTTP 応答から構成される。下りの音声データの送信は、1 HTTP 要求に対して、音声データが 1 HTTP 応答として送信される。ここで、Table 2 より得た結果より、通信を単純化し

Table 5 Transmission data per. Second per a robot

Image Data/ data packet (KB)	Voice Data (KB/s)	Image Data (KB/s)	Robot Con- trol Data (KB/s)	Packet head- er (KB/s)	Total data trans- fer (KB/s)	HTTP Request/ Response (Pairs/s)
3.5	32.0	35.0	1.4	57.4	125.8	21
10.0	32.0	100.0	1.4	57.4	190.8	21
20.0	32.0	200.0	1.4	57.4	290.8	21
50.0	32.0	500.0	1.4	57.4	590.8	21

た例を用いて、通信データ量と HTTP 要求と HTTP 応答の組数を求める。

例えば、音声データは16ビットPCM、8,000サンプル/秒とし、画像サイズを10[KB]、上下の音声取得時間を92[ms]、上下のデータ転送間隔を100[ms]とし、ロボット動作制御通信が1秒に1回行われるとする。このとき、ロボット制御パケットのHTTP応答・HTTP要求はおのの1.4[KB]、音声データ1.6[KB]と制御データ1.4[KB]の合計3[KB]、HTTP要求における上りのRSNPの音声・画像データパケット(HTTP要求)は、制御データ1.4[KB]、画像データ10[KB]、音声データ1.6[KB]で、合計13[KB]である。HTTP応答パケットは制御データのみで1.4[KB]である。下りの音声データパケット(HTTP応答)は、音声データ1.6[KB]と制御データ1.4[KB]の合計3[KB]、HTTP要求パケットは1.4[KB]である。これらが1秒に10回の通信が行われるので、上り145.4[KB]/秒、下り45.4[KB]/秒、合計190.8[KB]/秒で、HTTP要求とHTTP応答は、21組/秒となる。Table 5に画像サイズを変化させた場合の上下合わせたデータ量、およびHTTP要求とHTTP応答の組数を示す。また、Table 5の画像サイズは、それぞれ192×144ピクセル、480×360ピクセル、480×360ピクセル、1280×720ピクセルのjpegデータに対応している。尚、RSNPパケットに関する、制御データ、HTTPとの対応はRSNP仕様書[12]を参照されたい。

ここでHTTPの要求と応答の組数を減らすには1回の通信データ量を大きくすることになるが、音声の遅延とのトレードオフになる。また、通信のデータ量については、8,000サンプル/秒程度の音声データの場合は、音声データが占める割合は比較的小さい。

5.2.2 ロボットとサーバ間の通信量

対象とする無線LAN規格において最も速度の遅いIEEE802.11bの最大速度11[Mbps]を利用した場合、5.2.1項で述べた通信帯域占有率は約15%になり、通信可能な範囲と言える。

5.2.3 無線アクセスポイントからロボットサービスサーバまでの通信量

無線アクセスポイントからサーバまでは公共のインターネット使うことになる。光ファイバーを用いたインターネット接続のデータ転送速度は、最大100[Mbps]から1[Gbps]であり、1ロボットを接続した本システムの利用の場合は帯域の問題はない。しかしながら、1ロボットであっても、24時間接続して見守りサービスを利用した場合は、5.2.1項の例でも、15.89[GB]/日の転送量になる。数台のロボットをサービスに接続しただけでも、50[GB]/日のデータ転送を行うことになる。現在、イン

ターネットの利用については30[GB]/日の帯域制限をはじめとした様々な帯域利用制限が検討されている。したがって、インターネットを利用したロボットサービスを検討する際には、こうした帯域利用の制限に配慮し、ネットワークへの負荷の軽減を検討する必要がある。

5.2.4 ネットワーク負荷への考察

これらの見積り結果より、本稿で述べている見守りサービスアプリケーションが行うべき工夫について述べる。見守りサービスでは、平常時は、画像の頻繁な更新は必要なく、また、音声の会話は異常時などに限定的に用いるとした場合は、長時間である必要はない。これらを考慮した場合、アプリケーションは、以下の工夫を行うことができる：①平常時は、画像の更新頻度を下げ、1画像当たりのデータ量を減らす、②音声は、無送信でも済む場合はできるだけ送信せず、また、一方向に送信するのみとする。この場合は会話を伴わないため、音声の遅れに配慮する必要はないため、音声データのパケットの送信間隔、およびパケットを長くする。これによりパケット数を減らすことができる。本試作では、4.2.3項で述べたように音声の取得間隔を変更し、音声と画像のパケットの送信間隔を変更する仕組みを持つため、これに対応できる。プラットフォームとして音声および画像通信機能を提供する際は、こうしたパラメータを容易に設定、および変更する機能をアプリケーションに提供する必要がある。

6. まとめ

本稿では、音声通信機能を備えた見守りサービスの実現を目的とし、音声通信をロボットサービス基盤へ統合したサービスプラットフォームを提案した。3章ではRSNP上に音声通信を統合したプラットフォームを提案し、4章ではプロトタイプシステムを構築することで統合が可能であることを示した、5章では、プロトタイプシステムによる評価実験を行い、インターネットを介して遠隔地にいる見守り対象との会話ができ、提案した音声通信をロボットサービス基盤へ統合したサービスプラットフォームを構築可能であることを示した。さらに、得られた結果から、見守りサービスアプリケーションが行うべき工夫と、利用環境に合わせて設定値を調整することの有用性を述べた。

今後は、複数ロボットの同時接続も視野にいれインターネット経由でのネットワーク高負荷環境における検証を行い、実サービスとして展開できるよう実証実験を行っていく。さらに、音声利用のサービスを効率的に利用できるように、コーデックの見直しなどによる、帯域の圧縮にも取り組んでいく予定である。

さらに、本提案をもとにRSNP仕様へフィードバックを行っていくことを検討している。音声通信品質の向上については、

RSNP が利用する下位の通信層の見直しも検討課題といえる。現状の RSNP ライブラリは HTTP を通信路として構築されているが、RSNP は下位の通信層に依存しない設計であるため、より軽量の通信路に切り替えることで、さらなる音声通信品質の向上の可能性がある。通信層の実装も含め、引き続き検討をすすめていく。

謝 辞 本研究の一部は、JSPS 科研費 26330299 の助成を受けたものです。

参 考 文 献

- [1] N. Ando, T. Suehiro, K. Kitagaki, T. Kotoku and W.-K. Yoon: "RT-Middleware: Distributed Component Middleware for RT (Robot Technology)," Proc. 2005 IEEE/RSJ international Conference on Intelligent Robots and Systems (IROS2005), pp.3555–3560, 2005.
- [2] 佐藤知正, 岡野克弥: "RT ミドルウェアと知能モジュール構築プロジェクト", 日本ロボット学会誌, vol.28, no.5, pp.546–549, 2010.
- [3] RSi-Robot Service initiative, <http://robotsservices.org/> [Online]
- [4] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler and A. Ng: "ROS: an open-source Robot Operating System," Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA2009) Workshop on Open Source Robotics, 2009.
- [5] K. Kamei, S. Nishio and N. Hagita: "Cloud networked Robotics," IEEE Network, vol.26, no.3, pp.28–34, 2012.
- [6] R. Arumugam, R. Enti, L. Bingbing, W. Xiaojun, K. Baskaran, F.F. Kong, A.S. Kumar, K.D. Meng and G.W. KitVikas: "DAVINCI: A Cloud Computing Framework for Service Robots," IEEE International Conference on Robotics and Automation (ICRA2010), 2010.
- [7] J.J. Kuffner: "Cloud-Enabled Robots," IEEE-RAS International Conference on Humanoid Robots 2010, 2010.
- [8] M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Galvez-Lopez, Kai-Haussermann, R. Janssen, J.M.M. Montiel, A. Perzylo, B. Schiessle, M. Tenorth, O. Zweigle and Rene van de Molengraft: "Robo Earth —World Wide Web for Robots—", Robotics & Automation Magazine, IEEE, Jun. 2011, pp.69–82, 2011.
- [9] 加藤由花, 岡部泉, 村川賀彦, 岡林桂樹, 植木美和, 土屋陽介, 成田雅彦: "インターネットサービスプラットフォームにおけるロボットサービス提供手法", 情報処理学会論文誌, vol.54, no.2, pp.1234–1246, 2013.
- [10] Skype, <http://www.skype.com/ja/> [Online]
- [11] LINE, <http://line.naver.jp/ja/> [Online]
- [12] ロボットサービスイニシアチブ, Robot Service Network Protocol 2.3 仕様書 1.0 版, 2010.
- [13] 成田雅彦, 村川賀彦, 植木美和, 岡林桂樹, 秋口忠三, 日浦亮太, 蔵田英之, 加藤由花: "インターネットを活用したロボットサービスの実現と開発を支援する RSi (Robot Service Initiative) の取り組み", 日本ロボット学会誌, vol.28, no.7, pp.829–840, 2010.
- [14] 堀俊夫, 佐藤幹: "Robotic Interaction Service (RoIS) Framework: サービスロボットシステム用インタフェースの標準化", 日本ロボット学会誌, vol.29, no.4, pp.341–344, 2011.
- [15] 奥乃博, 中臺一博: "ロボット聴覚オープンソースソフトウェア HARK", 日本ロボット学会誌, vol.28, no.1, pp.6–9, 2010.
- [16] 塩見昌裕, 岩井儀雄, 角康之, 中臺一博, 萩田紀博: "対話行動認識プラットフォーム", 日本ロボット学会誌, vol.29, no.10, pp.883–886, 2011.
- [17] Y. Matsusaka, H. Asoh, I. Hara and F. Asano: "Specification and Implementation of Open Source Software Suite for Realizing Communication Intelligence," Journal ref: Journal of Robotics and Mechatronics, vol.24, no.1 pp.86–94, 2012.
- [18] B. Kehoe, A. Matsukawa, S. Candido, J. Kuffner and K. Goldberg: "Cloud-Based Robot Grasping with the Google Object Recognition Engine," IEEE International Conference on Robotics and Automation (ICRA2013), 2013.
- [19] R. Hickman, D. Kohler, K. Conley and B. Gerkey: Cloud Robotics, Google I/O 2011 Sessions, 2011.
- [20] D. Hunziker, M. Gajamohan, M. Waibel and R. D'Andrea: "Rapyuta: The RoboEarth Cloud Engine," IEEE the International Conference on Robotics and Automation (ICRA2013), 2013.
- [21] T. Mizumoto, K. Nakadai, T. Yoshida, R. Takeda, T. Otsuka, T. Takahashi and H.G. Okuno: "Design and Implementation of Selectable Sound Separation on the Texai Telepresence System using HARK," IEEE International Conference on Robotics and Automation (ICRA2011), 2011.
- [22] rosbridge.org, <http://rosbridge.org/doku.php> [Online]
- [23] B. Alexander, K. Hsiao, C. Jenkins, B. Suay and R. Toris: "Robot Web Tools," IEEE Robotics & Automation Magazine, Dec. 2012, pp.20–23, 2012.
- [24] 成田雅彦, 加藤由花, 土屋陽介: "小型ロボットを利用した簡易型 RSNP ロボットサービス開発環境の構築", 産業技術大学院大学紀要, vol.6, pp.53–59, 2012.
- [25] BANDAI, Smartpet Control library—API Reference Manual ver. 1.00, 2012.
- [26] ITU-T Recommendation G.114: One-way transmission time, 2003.

大澤秀也 (Hideya Osawa)

現在産業技術大学院大学産業技術研究科修士課程に
在学中。 (日本ロボット学会学生会員)

朝倉健介 (Kensuke Asakura)

現在産業技術大学院大学産業技術研究科修士課程に
在学中。

小原範子 (Noriko Obara)

現在産業技術大学院大学産業技術研究科修士課程に
在学中。

藤田尚宏 (Naohiro Fujita)

現在産業技術大学院大学産業技術研究科修士課程に
在学中。

佐藤 健 (Ken Sato)

現在産業技術大学院大学産業技術研究科修士課程に
在学中。

中川幸子 (Sachiko Nakagawa)

2012 年産業技術大学院大学産業技術研究科修士課
程修了 (情報システム学修士)。同年より産業技術
大学院大学客員研究員, 2013 年より同大学認定講
師としてロボットサービス基盤研究に従事。情報処
理学会会員。 (日本ロボット学会正会員)

成田雅彦 (Masahiko Narita)

1980 年早稲田大学大学院数学科修士課程修了。同
年富士通 (株) 入社。以来、ソフトウェアプラッ
トフォームの企画・研究開発と関連の国際標準活動に
従事。2008 年より産業技術大学院大学産業技術研
究科教授。2010 年首都大学東京大学院システムデ
ザイン研究科博士課程修了, 博士 (工学) 取得。電
子情報通信学会, 精密機械工学会会員。 (日本ロボット学会正会員)