2018年度　卒業論文

和文題目

NTMobileをコミュニケーションロボットへ応用する手法の検討

英文題目

A study on a method of applying NTMobile to a communication robot

情報工学科　渡邊研究室

(学籍番号: 130441021)

岩田　誠

提出日: 平成31年2月8日

名城大学理工学部

概要

日本の少子高齢化が進みそれに伴い高齢者の一人暮らしが問題視されている。また、l 様々なコミュニケーションロボットの開発が進み、ローカルで会話ができたり遠隔地から操作してモニタリングが出来るものも市場に出始め、コミュニケーションロボットが一人暮らしの高齢者の問題に対する有力な解決策であると考える。一方、我々の研究室ではオリジナル技術として NTMobile(Network Traversal with Mobility)[1] を研究している。NTMobile はエンドツーエンドの通信を可能にする。そこで、本論文ではコミュニケーションロボットに NTMobile を搭載して何ができるかを検討した。ラズベリーパイに車輪を取り付けたラズパイ戦車をコミュニケーションロボットに見立て、NTMobile を用いてスマートフォンから遠隔操作できることを確認した。

目 次

第 1 章　はじめに **1**

第 2 章　現状のコミュニケーションロボット **3**

2.1 Pepper 3

2.2 タピア 3

2.3 現状のコミュニケーションロボットの課題 4

第 3 章　NTMobile について **6**

第 4 章　提案方式 **7**

第 5 章　実験 **8**

5.1 準備 8

5.2 動作検証 10

5.3 HTML、Python 13

5.4 考察 16

第 6 章　まとめ **17**

謝辞 **19**

第1章　はじめに

内閣府によると日本の高齢化は増加傾向にあり 2065 年には日本の人口における 38.4 %が 65 歳以上の者になると予想されている。[2] それに伴い高齢者の一人暮らしも増加しており、2015 年時点で男性約 192 万人、女性約 400 万人にもおよぶ。高齢者の一人暮らしの問題の一つに認知症がある。一人暮らしの高齢者が認知症にかかると、地域の約束事を守れなくなったり、悪いことと認識できずに大声で騒ぎ周りの人に迷惑をかけてしまう。また、孤独死も大きな問題の一つである。

現在コミュニケーションロボットの研究が進み、人間と会話が出来たり、遠隔地から操作してテレビ電話やモニタリングができるものも市場に出始めており、今後も研究が進むと思われる。コミュニケーションロボットは一人暮らしの高齢者に対して、話し相手になることができ、遠くにいる家族がモニタリングして高齢者の状態を把握することもできる。普段から会話をすることや、家族とコミュニケーションをとることは認知症の予防にもつながる。

一方、我々の研究室ではオリジナル技術として NTMobile を研究している。NTMobile を利用すると、あらゆる環境においてエンドツーエンドの通信が可能となる。NTMobile は動作検証を終えており、現在その応用を検討しているところである。そこでコミュニケーションロボットに NTMobile を搭載して何ができるかを検討した。

コミュニケーションロボットにはペット型のロボットや人間と会話ができるロボット等、様々な種類がある。ペット型ロボットである AIBO[3] は本物の動物のようなしぐさをし使用者に癒しを与える。また、使用者のふれあい方次第で育ち方が変わるといった成長を楽しむこともできる。会話ができるロボットである Pepper[4] やタピア [5] は、ロボット自ら話しかけてくれたり感情表現をすることによって、より人間らしいふるまいができる。両者はディスプレイを搭載しているので画面を使ったアプリでゲームをしたり、遠くにいる家族とロボットを介したテレビ電話でコミュニケーションをとることができ、遠隔操作してモニタリングすることも可能である。

コミュニケーションロボットに搭載されている機能のうち NTMobile を利用できるものに着目したところ、テレビ電話と遠隔操作であった。この機能を持っているコミュニケーションロボットの代表として Pepper とタピアがある。しかし、一般的にコミュニケーションロボットは家庭内に置くため、プライベートアドレス空間に存在する。従って、遠隔地から操作する場合は NAT の制約上必ずインターネット上のサーバを経由する必要がある。現状のコミュニケーションロボットの通信はこのクライアント・サーバシステムで行われている。

本論文では、コミュニケーションロボットにサーバを置き、Web プログラミングにより操作する方法を提案する。Web プログラミングはアプリ開発など、専門的な知識が必要なく開発ツールもそろっているので、今後の NTMobile の開発プラットフォームになりうる。ラズベリーパイに車輪を取り付けたラズパイ戦車をコミュニケーションロボットに見立て、NTMobile を用いて遠隔操

作できることを確認した。

以後、2 章では現状のコミュニケーションロボット、3 章では NTMobile について、4 章では提案方式、5 章では実験、最後に 6 章でまとめる。

第2章　現状のコミュニケーションロボット

現状のコミュニケーションロボットとして特に遠隔操作、会話可能なものについて着目した。本章ではその中で Pepper とタピア、またその課題について述べる。

2.1　Pepper

Pepper は人型のコミュニケーションロボットで店舗への導入も進んでいる。AI を搭載しているので人間と会話することができ、様々なコミュニケーションをとることができる。また、豊富なアプリケーションにより写真や歌やダンスなど幅広い用途に対応している。無料のプログラミングツール、コレグラフを用いて、自分の好きなように Pepper をカスタマイズすることができる。コレグラフはあらかじめ Pepper をしゃべらせたり動かしたりするためのモジュールが用意されているので、専門的な知識がなくともプログラミングが可能になる。専用のアプリである Pepper View を使用することにより、Pepper と連携した iPhone、iPad を利用して Pepper を遠隔操作することができる。Pepper が見ている景色が iPhone の画面に表示され、室内を自由に動きモニタリングしたり、入力した文字を Pepper にしゃべらせることも可能である。

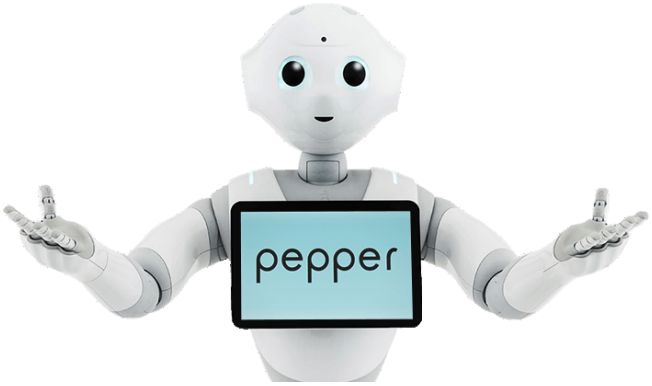


図 1　Pepper

2.2　タピア

タピアは球状の据え置き型のコミュニケーションロボットである。ディスプレイに目が表示されており、その表情で感情を表現し人間と会話ができる。またディスプレイにカメラがついておりスマートフォンからタピア目線でのモニタリングやテレビ電話が可能となる。スケジュールを

記憶させその時間になると呼びかけをしてくれる機能も搭載している。置き型のロボットのため Pepper のように動き回ることはできないが、専用のアプリであるタピアあぷりを使用して、連携したスマートフォンとテレビ電話やモニタリング、写真撮影が可能である。

図 2　tapia

2.3　現状のコミュニケーションロボットの課題

図 3 に現在のコミュニケーションロボットを遠隔操作する場合の通信を示す。コミュニケーションロボットは一般的に家庭内に置くため、プライベートアドレス空間に存在する。現在の通信はインターネットに接続されるホストが増えたためグローバル IP アドレスが枯渇するようになった。こうした状況に対応するため LAN 内のホストにはプライベート IP アドレスを割り当て、インターネットに接続する時だけグローバル IP アドレスを使用するようにするため、アドレスを変換する技術である NAT が用いられている。NAT の制約上、NAT の外側からは通信が開始できない。従って、スマートフォンからのモニタリングやテレビ電話をする場合必ずサーバを経由する。そのため、サーバへ接続するクライアントが増えるとその分サーバに負荷がかかり、クライアントが増えすぎるとサーバのリソース不足が全体のボトルネックになりやすい。また、スマートフォンで遠隔操作する場合、コミュニケーションロボット、サーバ、クライアント側のアプリケーションを別個開発する必要があり、専門的な知識も必要なため新規開発は容易ではない。

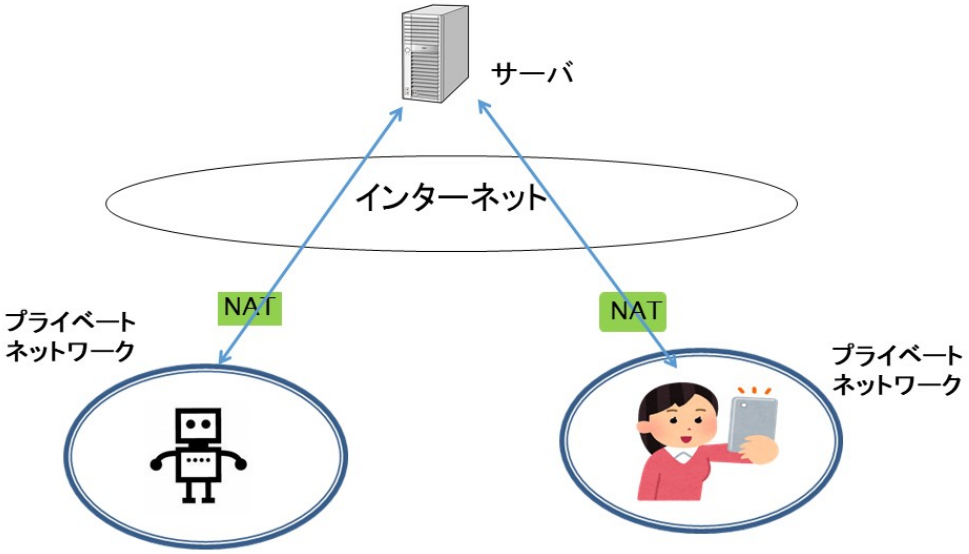


図 3　ロボットを遠隔操作する場合

第3章　NTMobileについて

本稿では、提案方式をオリジナル技術であるNTMobileを利用して実装する。そのため本章では移動透過性とNAT越え通信の両者を同時に実現するNTMobileについて、概要を述べる。

NTMobileは、通信中にネットワークが切り替わっても通信を継続することができる移動透過性と、NATの外側から通信を開始することができるNAT越えの両者を同時に実現する技術である。通信相手がどこにいてもエンドツーエンドの通信を確立できる。通信相手がプライベートアドレス空間であっても通信を開始できるという特徴がある。NTMobileをアプリケーションとしてインストールすれば、これまで使用していた既存のアプリケーションに変更を加えずそのまま使用することができる。すなわちLAN内での通信を確認することができたら、後からNTMobileを追加インストールすることによりそのままリモート通信に転用できる。

さらに、NTMobileと一般通信は併存させることができる。相手をNTMobile特有の名前で指定することによって、NTMobile上で動作する。

図 4にNTMobileの構成を示す。NTMobileを支える装置群としてAC、DC、RSが必要であるが、ユーザはこれらを意識する必要はない。ユーザはNTMobileをインストールするだけでエンドツーエンドの通信を確立することができる。

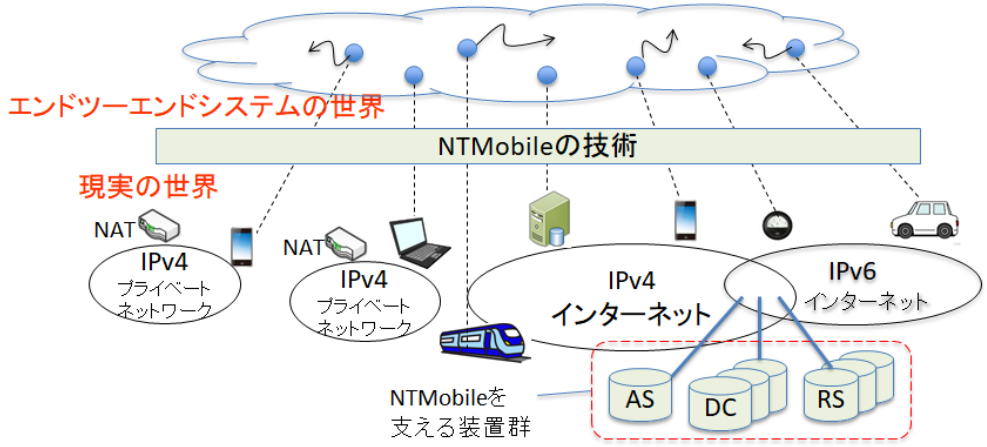


図 4　NTMobileの構成図

第4章　提案方式

本章では、コミュニケーションロボットに Web サーバと NTMobile を実装する提案方式について述べる。

提案方式を図 5 に示す。コミュニケーションロボットに Web サーバと NTMobile を実装する。また、操作するスマートフォンにも NTMobile を実装する。この状態において Web プログラミングによりサーバ機能を開発する方式を提案する。

Web プログラミングは一般に開発ツールがそろっているため、誰でも容易に開発することができる。また、サーバ側だけの開発で済み、スマートフォン側はブラウザさえ用意すればよい。試験は LAN 内の通信で確認できるため、アプリケーションの開発が容易で新規開発が極めて簡単である。また、NTMobile によりエンドツーエンドの通信が可能なので、サーバを経由させるような無駄なパケットが不要で通信ディレイも少ないという利点がある。

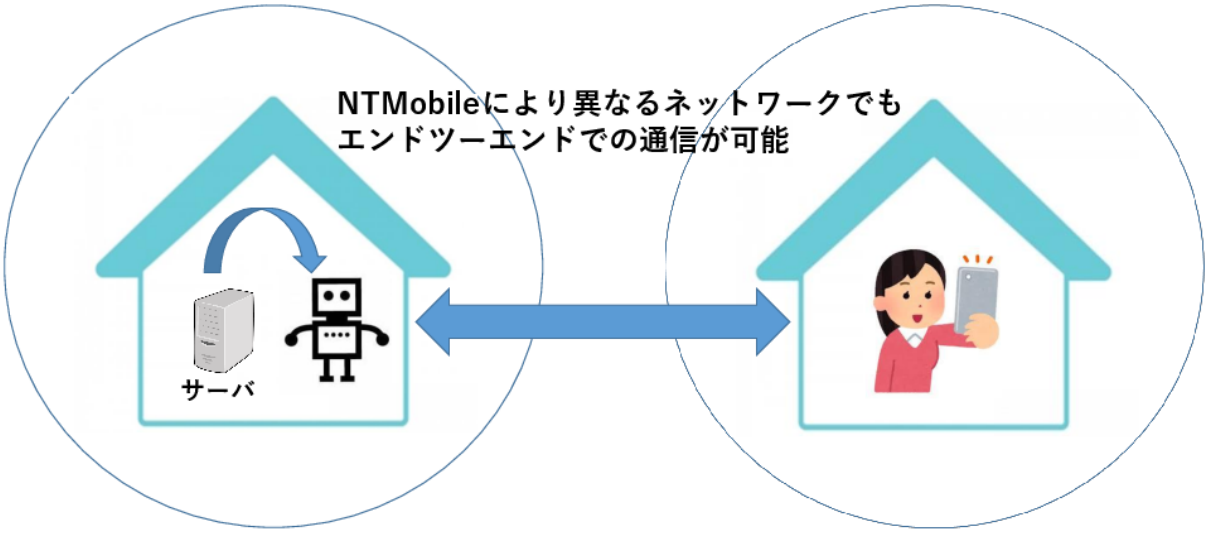


図 5　提案方式

第5章　実験

本章では、提案方式の実験および動作の確認について述べる。実験はラズベリーパイに車輪を取り付けたラズパイ戦車をコミュニケーションロボットに見立てて行った。

5.1　準備

ラズベリーパイにサーバ機能を持たせるため WebIOPi[6] を用いた。WebIOPi は Web サーバを用いてラズベリーパイをリモート操作することができるフレームワークである。

HTML で操作画面となる Web ページを作成し、Python によるプログラムでラズベリーパイのモーター制御を行った。[7] それぞれのファイルはラズベリーパイの/home/pi/WebIOPi-0.7.1/htdocs に bb というフォルダを作り置いておく。

ラズベリーパイはモータドライバ DRV8835 を用いて DC モーターとピンで接続した。モーターの電源は電池、ラズベリーパイの電源はモバイルバッテリーから供給される。

最後に NTMobile をラズベリーパイと Linux に実装した。

製作したラズパイ戦車、操作画面をそれぞれ図 6、図 7 に示す。また、ラズベリーパイとモータドライバとの接続を図 8 に示す。ラズベリーパイ側のモジュール構成、クライアント側のモジュール構成をそれぞれ図 9、図 10 に示す。

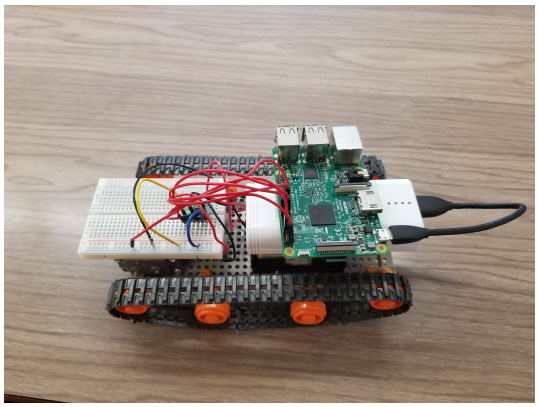


図 6　ラズパイ戦車

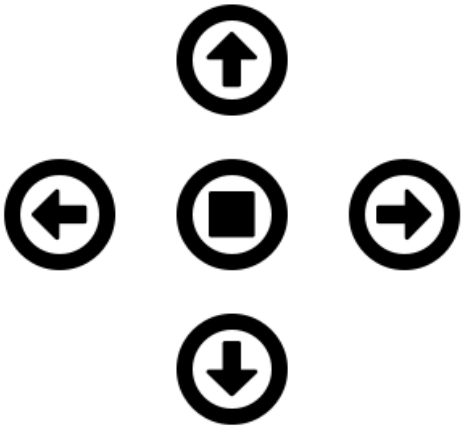


図 7　操作画面

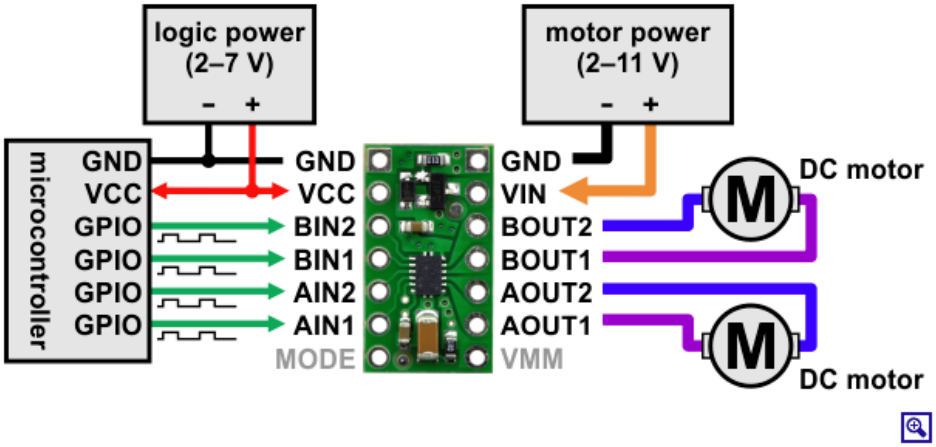


図 8　ピン配線

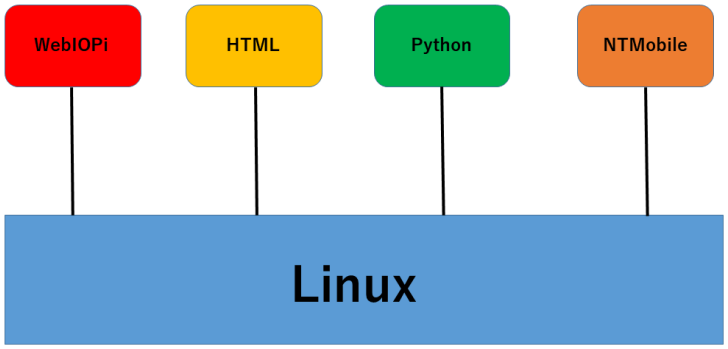


図 9　モジュール構成 (ラズベリーパイ)

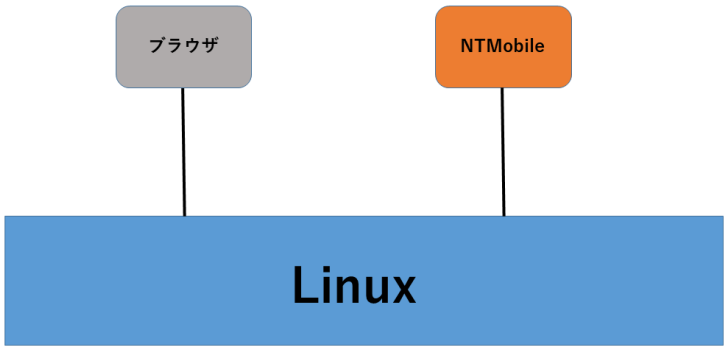


図 10　モジュール構成 (クライアント)

5.2 動作検証

まずLAN環境においてスマートフォンのブラウザからラズパイ戦車が動作することを確認した。スマートフォンのブラウザのアドレスバーに「http : //ラズベリーパイのIPアドレス : 8000/bb/」と入力することにより、用意したHTMLのWebページに推移する。Webページには上下左右の矢印と停止ボタンを表示させた。矢印を押すとその方向に車輪が回転し、停止ボタンを押すことにより回転が止まる。LAN環境でのパケットをwire sharkで確認したものを図11、シーケンスを図12に示す。スマートフォンのブラウザの矢印をタッチした時、はじめに3ウェイハンドシェイクによってコネクションを確立する。次にPOSTでサーバへリクエストを送る。その後サーバから応答とOKが送信されラズパイ戦車の車輪が回転する。最後にコネクションを切断して一通りの操作を終える。停止ボタンを押して回転を止める時も同様の処理が行われる。

次にラズベリーパイとLinuxにNTMobileを実装し、それぞれ別のネットワークに接続しなおしてLAN環境と全く同じ動作を確認した。NTMobileで動作させる場合は、アドレスバーにラズベリーパイのIPアドレスではなく「ras0.dc.ntm.jp」とNTMobile特有の名前を入力する。NTMobileを実装した場合のパケットをwire sharkで確認したものを図13に示す。NTMobileを実装したことによりパケットは全てカプセル化され暗号化されており、94Byte分長くなっている。異なるプライベートアドレス空間からでも直接通信できることを確認した。

	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	198.19.165.51	198.19.177.195	TCP	60	57850 → 8000 [SYN] Seq=0 Win=25200 Len=0 MSS
2	0.104751013	198.19.177.195	198.19.165.51	TCP	60	8000 → 57850 [SYN, ACK] Seq=0 Ack=1 Win=2736
3	0.104790503	198.19.165.51	198.19.177.195	TCP	52	57850 → 8000 [ACK] Seq=1 Ack=1 Win=25216 Len
4	0.107110525	198.19.165.51	198.19.177.195	HTTP	656	POST /macros/forward/ HTTP/1.1
5	0.196493472	198.19.177.195	198.19.165.51	TCP	52	8000 → 57850 [ACK] Seq=1 Ack=605 Win=28672 L
6	0.205084863	198.19.177.195	198.19.165.51	HTTP	211	HTTP/1.0 200 OK
7	0.205102543	198.19.165.51	198.19.177.195	TCP	52	57850 → 8000 [ACK] Seq=605 Ack=160 Win=26368
8	0.207394494	198.19.165.51	198.19.177.195	TCP	52	57850 → 8000 [FIN, ACK] Seq=605 Ack=160 Win=
9	0.210463646	198.19.177.195	198.19.165.51	TCP	52	8000 → 57850 [FIN, ACK] Seq=160 Ack=605 Win=
10	0.210481303	198.19.165.51	198.19.177.195	TCP	52	57850 → 8000 [ACK] Seq=606 Ack=161 Win=26368
11	0.292300800	198.19.177.195	198.19.165.51	TCP	52	8000 → 57850 [ACK] Seq=161 Ack=606 Win=28672
12	1.803340588	198.19.165.51	198.19.177.195	TCP	60	57852 → 8000 [SYN] Seq=0 Win=25200 Len=0 MSS
13	1.990037083	198.19.177.195	198.19.165.51	TCP	60	8000 → 57852 [SYN, ACK] Seq=0 Ack=1 Win=2736
14	1.990073415	198.19.165.51	198.19.177.195	TCP	52	57852 → 8000 [ACK] Seq=1 Ack=1 Win=25216 Len
15	1.992396017	198.19.165.51	198.19.177.195	HTTP	653	POST /macros/stop/ HTTP/1.1
16	2.076130616	198.19.177.195	198.19.165.51	TCP	52	8000 → 57852 [ACK] Seq=1 Ack=602 Win=28672 L
17	2.080991055	198.19.177.195	198.19.165.51	HTTP	211	HTTP/1.0 200 OK
18	2.081008552	198.19.165.51	198.19.177.195	TCP	52	57852 → 8000 [ACK] Seq=602 Ack=160 Win=26368
19	2.083639664	198.19.165.51	198.19.177.195	TCP	52	57852 → 8000 [FIN, ACK] Seq=602 Ack=160 Win=
20	2.099188200	198.19.177.195	198.19.165.51	TCP	52	8000 → 57852 [FIN, ACK] Seq=160 Ack=602 Win=
21	2.099214032	198.19.165.51	198.19.177.195	TCP	52	57852 → 8000 [ACK] Seq=603 Ack=161 Win=26368
22	2.211533443	198.19.177.195	198.19.165.51	TCP	52	8000 → 57852 [ACK] Seq=161 Ack=603 Win=28672

図 11　LAN 環境での wire shark

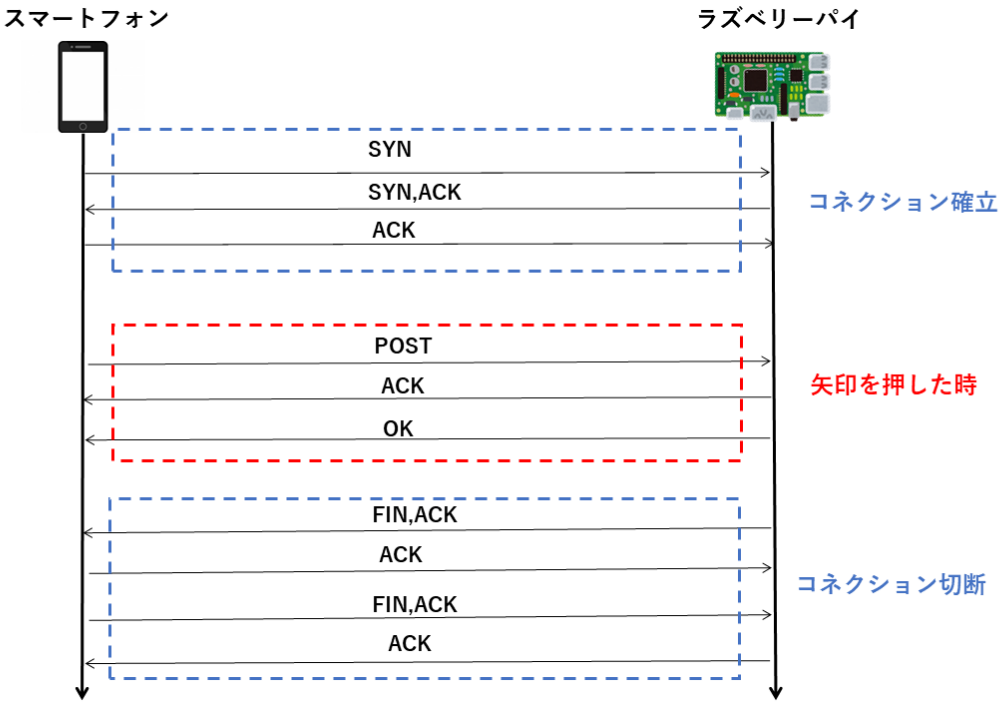
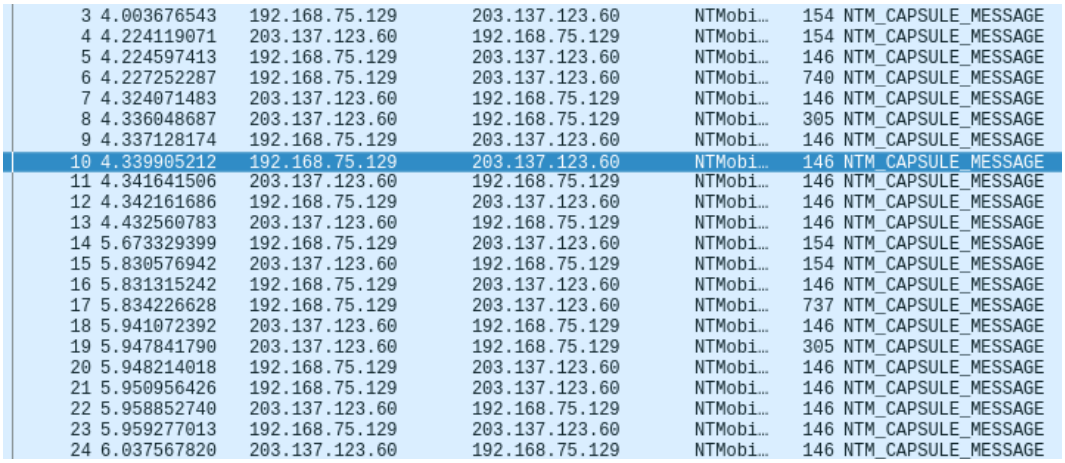


図 12　LAN 環境でのシーケンス



3	4.003676543	192.168.75.129	203.137.123.60	NTMobi…	154	NTM_CAPSULE_MESSAGE
4	4.224119071	203.137.123.60	192.168.75.129	NTMobi…	154	NTM_CAPSULE_MESSAGE
5	4.224597413	192.168.75.129	203.137.123.60	NTMobi…	146	NTM_CAPSULE_MESSAGE
6	4.227252287	192.168.75.129	203.137.123.60	NTMobi…	740	NTM_CAPSULE_MESSAGE
7	4.324071483	203.137.123.60	192.168.75.129	NTMobi…	146	NTM_CAPSULE_MESSAGE
8	4.336048687	203.137.123.60	192.168.75.129	NTMobi…	305	NTM_CAPSULE_MESSAGE
9	4.337128174	192.168.75.129	203.137.123.60	NTMobi…	146	NTM_CAPSULE_MESSAGE
10	4.339905212	192.168.75.129	203.137.123.60	NTMobi…	146	NTM_CAPSULE_MESSAGE
11	4.341641506	203.137.123.60	192.168.75.129	NTMobi…	146	NTM_CAPSULE_MESSAGE
12	4.342161686	192.168.75.129	203.137.123.60	NTMobi…	146	NTM_CAPSULE_MESSAGE
13	4.432560783	203.137.123.60	192.168.75.129	NTMobi…	146	NTM_CAPSULE_MESSAGE
14	5.673329399	192.168.75.129	203.137.123.60	NTMobi…	154	NTM_CAPSULE_MESSAGE
15	5.830576942	203.137.123.60	192.168.75.129	NTMobi…	154	NTM_CAPSULE_MESSAGE
16	5.831315242	192.168.75.129	203.137.123.60	NTMobi…	146	NTM_CAPSULE_MESSAGE
17	5.834226628	192.168.75.129	203.137.123.60	NTMobi…	737	NTM_CAPSULE_MESSAGE
18	5.941072392	203.137.123.60	192.168.75.129	NTMobi…	146	NTM_CAPSULE_MESSAGE
19	5.947841790	203.137.123.60	192.168.75.129	NTMobi…	305	NTM_CAPSULE_MESSAGE
20	5.948214018	192.168.75.129	203.137.123.60	NTMobi…	146	NTM_CAPSULE_MESSAGE
21	5.950956426	192.168.75.129	203.137.123.60	NTMobi…	146	NTM_CAPSULE_MESSAGE
22	5.958852740	203.137.123.60	192.168.75.129	NTMobi…	146	NTM_CAPSULE_MESSAGE
23	5.959277013	192.168.75.129	203.137.123.60	NTMobi…	146	NTM_CAPSULE_MESSAGE
24	6.037567820	203.137.123.60	192.168.75.129	NTMobi…	146	NTM_CAPSULE_MESSAGE

図 **13**　**NTMobile** を実装した場合の **wire shark**

5.3 HTML、Python

実験で使用した HTML と Python のプログラムをそれぞれソースコード 5.1、ソースコード 5.2 に示す。

ソースコード **5.1** HTML

```
<script src="/webiopi.js"></script>
<script>
function init() {
}
function stop() {
    webiopi().callMacro("stop");
}
function forward() {
    webiopi().callMacro("forward");
}
function reverse() {
    webiopi().callMacro("reverse");
}
function turnLeft() {
    webiopi().callMacro("turnLeft");
}
function turnRight() {
    webiopi().callMacro("turnRight");
}
webiopi().ready(init);
</script>

<script src="https://use.fontawesome.com/6497e53239.js"></script>
<p>
    <i class="fa fa-fw fa-5x" aria-hidden="true"></i>
    <i class="fa fa-arrow-circle-o-up fa-fw fa-5x" aria-hidden="true"
        onmousedown="forward()" ></i>
    <i class="fa fa-fw fa-5x" aria-hidden="true"></i>
</p>
<p>
    <i class="fa fa-arrow-circle-o-left fa-fw fa-5x" aria-hidden="true"
        onmousedown="turnLeft()" ></i>
    <i class="fa fa-stop-circle-o fa-fw fa-5x" aria-hidden="true" onmousedown
        ="stop()"></i>
    <i class="fa fa-arrow-circle-o-right fa-fw fa-5x" aria-hidden="true"
        onmousedown="turnRight()" ></i>
</p>
<p>
    <i class="fa fa-fw fa-5x" aria-hidden="true"></i>
    <i class="fa fa-arrow-circle-o-down fa-fw fa-5x" aria-hidden="true"
        onmousedown="reverse()"></i>
    <i class="fa fa-fw fa-5x" aria-hidden="true"></i>
</p>
```

ソースコード **5.2　Python**

```
import webiopi
GPIO = webiopi.GPIO

leftMotorDrivePWM1 = 11
leftMotorDrivePWM2 = 9
rightMotorDrivePWM1 = 22
rightMotorDrivePWM2 = 23

def setup():
    webiopi.debug("setup")

    GPIO.setFunction(leftMotorDrivePWM1, GPIO.PWM)
    GPIO.setFunction(leftMotorDrivePWM2, GPIO.PWM)
    GPIO.setFunction(rightMotorDrivePWM1, GPIO.PWM)
    GPIO.setFunction(rightMotorDrivePWM2, GPIO.PWM)

@webiopi.macro
def forward():
    webiopi.debug("forward")

    GPIO.pwmWrite(leftMotorDrivePWM1, GPIO.LOW)
    GPIO.pwmWrite(leftMotorDrivePWM2, GPIO.HIGH)
    GPIO.pwmWrite(rightMotorDrivePWM1, GPIO.LOW)
    GPIO.pwmWrite(rightMotorDrivePWM2, GPIO.HIGH)

@webiopi.macro
def reverse():
    webiopi.debug("reverse")

    GPIO.pwmWrite(leftMotorDrivePWM1, GPIO.HIGH)
    GPIO.pwmWrite(leftMotorDrivePWM2, GPIO.LOW)
    GPIO.pwmWrite(rightMotorDrivePWM1, GPIO.HIGH)
    GPIO.pwmWrite(rightMotorDrivePWM2, GPIO.LOW)

@webiopi.macro
def turnLeft():
    webiopi.debug("turnLeft")

    GPIO.pwmWrite(leftMotorDrivePWM1, GPIO.LOW)
    GPIO.pwmWrite(leftMotorDrivePWM2, GPIO.HIGH)
    GPIO.pwmWrite(rightMotorDrivePWM1, GPIO.HIGH)
    GPIO.pwmWrite(rightMotorDrivePWM2, GPIO.LOW)

@webiopi.macro
def turnRight():
    webiopi.debug("turnRight")

    GPIO.pwmWrite(leftMotorDrivePWM1, GPIO.HIGH)
    GPIO.pwmWrite(leftMotorDrivePWM2, GPIO.LOW)
    GPIO.pwmWrite(rightMotorDrivePWM1, GPIO.LOW)
```

```
51      GPIO.pwmWrite(rightMotorDrivePWM2, GPIO.HIGH)
52
53  @webiopi.macro
54  def stop():
55      webiopi.debug("stop")
56
57      GPIO.pwmWrite(leftMotorDrivePWM1, GPIO.LOW)
58      GPIO.pwmWrite(leftMotorDrivePWM2, GPIO.LOW)
59      GPIO.pwmWrite(rightMotorDrivePWM1, GPIO.LOW)
60      GPIO.pwmWrite(rightMotorDrivePWM2, GPIO.LOW)
```

HTML の 1 行目は WebIOPi の Javascript ファイルである「webiopi.js」を読み込む処理である。3 行目から 19 行目は Python で定義したマクロ関数を呼び出す処理である。23 行目から 38 行目でそれぞれ矢印と停止ボタンを表示させる。

Python の 1、2 行目は WebIOPi を使用するために記述する。4 行目から 7 行目で 2 つのモーターとラズベリーパイのピン番号を指定する。9 行目から 15 行目でそれぞれモーターの関数を定義する。17 行目から 60 行目でそれぞれモーターの回転を制御する。

5.4 考察

今回行ったWebプログラミングでの開発は専門的な知識がなくともHTMLやPython等のプログラムが書ければ誰でも行えるため、開発が容易である。開発するのもサーバ側だけで済み、クライアント側はブラウザだけ用意すればよい事も利点である。サーバがプライベート空間にあるので試験がLAN内の通信で確認でき、セキュリティ面の向上、サーバの負担が軽減される。今回はラズベリーパイを用いてNTMobileを応用させたが、コミュニケーションロボットに限らず様々な開発に応用できると考える。

第6章　まとめ

本稿では、コミュニケーションロボットにNTMobileを応用して何ができるかを検討した。ラズパイ戦車をコミュニケーションロボットに見立て、Webプログラミングを利用してコミュニケーションロボットを簡単に操作できることを確認した。NTMobileを利用することによってコミュニケーションロボットにサーバを置いても異なるプライベートアドレス空間で直接通信することができる。また、クライアント側はブラウザさえ用意できれば操作できる。この開発方法はコミュニケーションロボットに限らず、あらゆるシステムに応用していくことができる。

謝辞

本研究を進めるにあたり、多大なるご指導を賜りました、指導教官である名城大学理工学部情報工学科　渡邊晃教授に心から感謝致します。

最後に、本研究を進めるにあたり、日頃から多くの有益なご意見を賜りました、渡邊研究室の皆様に感謝致します。

参考文献

[1] https://aibo.sony.jp/

[2] https://www.softbank.jp/robot/consumer/products/

[3] https://mjirobotics.co.jp/

[4] https://www8.cao.go.jp/kourei/whitepaper/w-2018/html/zenbun/index.html

[5] 上酔尾一真, 他: IPv4/IPv6 混在環境で移動透過性を 実現する NTMobile の 実装と評価, 情報処理学会論文 誌 Vol.54,No10,pp.2288-2299,Oct.2013.

[6] https://webiopi.trouch.com/

[7] https://qiita.com/imcuddles/items/c05cbea95db1f7469fed