IoT を用いたロボット制御における遅延時間の測定

宮浦啓太 伊藤順治 [1]

概要：ロボット制御における遅延時間は重要な課題である．著者らは，Chat System を用いたロボット制御を行っており，遅延時間の測定を行った．具体的には，有線や，有線+ Wi-Fi 等の遅延時間を測定した．有線は，最小:1ms，最大:365ms だった．一方で，有線+ Wi-Fi は，5GHz 帯の場合，最小:2ms，最大:14ms．2.4GHz 帯の場合，最小:2ms，最大:117ms なった．どちらも最小の遅延時間は同じだが，比較すると，2.4GHz 帯の遅延時間はテールを引くように伸びている．これは，他のデータが入ってきたときに待ち時間が発生するからだと考えられる．加えて，Wi-Fi を用いた画像データの遅延時間も測定した結果，上記の遅延時間に加え，画像データの圧縮，展開にかかる時間が加算されることが分かった．

キーワード：2100201 データ通信プロトコル　2040202 リアルタイム　2110307 ロボットシステム

Measurement of delay time in robot control using IoT

KEITA MIYAURA , JUNJI ITO[†1]

Abstract: The authors studied robot control using the Chat Systems. In this paper we measured the delay time.by using LAN cable Wi-Fi and smart phone system(LTE) LAN cable was minimum:1ms and maximum:365ms. On the other hand, Wi-Fi has a minimum of 2ms and a maximum of 14ms in the 5GHz band. In the case of LTE, the minimum is 52ms and the maximum is: 1467ms. In addition, as a result of measuring the delay time of the image data using Wi-Fi, it was found that the time required for compression and decompression of the image data is added to the above delay time.

Keywords: 2100201 Data communication protocol 2040202real time 2110307 Robot system

1. 背景及び目的

1.1 背景

新型コロナウイルス感染症の流行や，インターネット技術，テクノロジーの進化等を背景に，パソコンやスマートフォンなど従来のスマートデバイスに加え，家電や自動車，ビルや工場などの IoT 化が進んでいる[1][2]．また，バイオニックロボットの 1 つとして，Hexapod robot の研究[3][4][5]が行われており，最近では，起伏の多い土地や 60 度程度の斜面を登攀出来るようになってきている[6]．一方で，エッジコンピューティングによる高速処理が可能になり，更に高度なセンシング技術の向上に伴い，デプスカメラのような 3 次元的な環境把握も実現可能になってきている．これらに伴い，IoT や 5G を用いたロボット制御が可能になってきている．

現時点では，1 対 1 のロボット制御が主流だが，今後 IoT 化が更に進むにつれ，1 対多数のロボット制御が必要になってくると考えられる．そこで課題になるのが，複数ロボットの制御方法と遅延時間であると考える．リアルタイム制御を行う際に遅延時間が発生することで，予期せぬことが起こった場合の対処が遅れてしまうことや，想定していた経路等から大きく外れてしまいソフトウェアクラッシュが懸念される．

実際に，総務省による「高速移動時において無線区間 1ms，End-to-End で 10ms の低遅延通信を可能とする第 5 世代移動通信システムの技術的条件等に関する調査検討」[7]で言われているように，遠隔自動運転を実現するには，10~100ms の遅延時間で制御する必要がある．

1.2 目的・方法

本研究では chat system を利用し，前述の課題に対して取り組んでいく[8]．現在 chat system は多くのところで使われている[9][10][11][12]．chat の特徴としては 2 人以上の人とリアルタイムの通信ができる．一般的に，会話を行っているユーザーは相手と個別にやり取りしていると思われている．しかし，実際には様々な箇所に配置されたサーバー上で処理されているため，ユーザーの環境によって遅延時間が発生している．そこで今回は，無線と有線の遅延時間の比較を行うことにより，ここから出てくる課題やさらなる活用への検討を行った．更に，Chat の特徴を利用することにより，ロボット制御へのアプローチも行った．また，画像データの遅延時間を測定することにより，低用量データの遅延時間と比較し，考察を行う．物理レイヤーには IEEE802.11ac を使って検証を行った．

1 日本文理大学
Nippom Bunri Univercsity

2. 実験の詳細

2.1 Chat System の利用

chat の大まかな仕組みとしては複数のクライアントが chat server に接続することで一人のメッセージがリアルタイムに，繋がっているクライアントに表示されるというものだ．一般的には chat のクライアントは人であるが今回はそれが人とだけではなくあらゆる機器もクライアントである．一つの chat server にあらゆる機器を接続し，何か問題が起きた時には chat server にメッセージを送る．chat はリアルタイム性が高いため server に接続している人はすぐに異常が確認できる．また，chat server を通しての遠隔操作も可能である．このように chat system を利用していく．

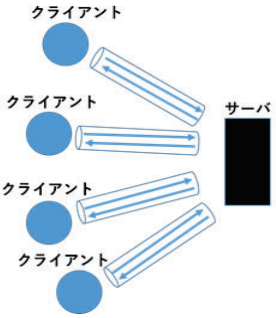

図 1 Chat の流れ

2.2 Chat System の構築

chat system を構築するにあたり，プログラムを JavaScript で書いていくことにした．その理由としては，現在多くのエンジニアが利用している言語であり，Node.js を利用することでクライアントとサーバーをどちらも同じ言語で書くことができるからだ．また，Node.js はシングルスレッドでのイベントループやノンブロッキング I/O という特徴を持っている．

chat system はリアルタイム web 技術によって構築されている．これまでに使われてきた技術には，ポーリングや Comet，Websocket などが挙げられる．そのような中で今回は Socket.IO という Node.js のライブラリの一つを利用した．これは，先程述べたリアルタイム web 技術を透過的に利用できるという特徴がある．

2.3 Chat System の構築（遅延時間）

Chat system を用いて有線や無線の遅延時間を測定するために UTC（協定世界時）を用いた．今回作成したプログラムでは JavaScript の Date.now()関数を利用した．

Date.now()関数とは，UTC (協定世界時) での 1970 年 1 月 1 日 0 時 0 分 0 秒 から現在までの経過時間をミリ秒単位で返すものである．

これを利用することで，送信時間を送信データに入れて送り，そのデータを受信することによって，1ms 単位で遅延時間を計測した．

2.4 Chat System の構築（制御）

chat system の動きを確認するために身近にあった鉄道模型の遠隔操作の実験を行った．今回作成したプログラムは chat server を構成するプログラムと web 上で人間が鉄道模型を遠隔コントロールすることを実現した画面を形成する html 言語で作成したプログラム．実際にエッジコンピューティングとして鉄道働模型上に実装したシングルコンピューター:Raspberry Pi Zero 上で実行される動作プログラミングの 3 つである．エッジコンピューティングは従来のマイコン上のシングルタスクではなく，Linux 上で複数のプログラムが同時に動くマルチタスク動作を行っている．

実際の動作アルゴリズムの流れは以下の様になる．

①chat server を起動し 2 つのクライアントを接続する．

②html で書かれたプログラムでコントローラーをハッキングし信号を読み取る．

③読み取った信号を chat server にメッセージとして送る．

④クライアントが chat server にきたメッセージを読み取りその通りに動く．

更に web アプリ上のコントローラー画面上では搭載された 5MPixel のイメージセンサによって取得し，Motion-JPEG に圧縮された動画がほぼリアルタイムに配信表示されている．

初回実験においてコントローラーでの速度変化時や走行中に速度変化以外のボタンを押した際に一瞬停止する，本来の順序とは異なる信号が送られてくる問題が発生した．検討の結果，コントローラーから信号を取得する際に同じ信号をいくつも取得してしまうことや，他のボタンを押したときに一瞬だけ速度 0 の信号が送られることが原因であると判明した．改善方法として，既存のコントローラーを改造することはハードウェア変更のため費用と時間がかかる．そこで，ソフトウェア上で工夫をすることにより問題解決を行った．具体的には，エッジコンピューティングを行っているクライアントのプログラムの変更を行った．以上のことから改良したプログラムを実装して実験を行った．その結果，コントローラーの信号を忠実に実現し

正常に鉄道模型の制御が可能となった．しかしながら，依然として送られてくる信号の順番が異なるという問題は解決できなかった．これは人が制御する際，次にどんな信号が来るかの判断ができないために発生するものであるが，送られてくる信号は次の信号においては真値のものであるため，実用上の問題ないと考え無視した．以上の検討を行った結果．chat system を利用した遠隔操作を実現することができた．

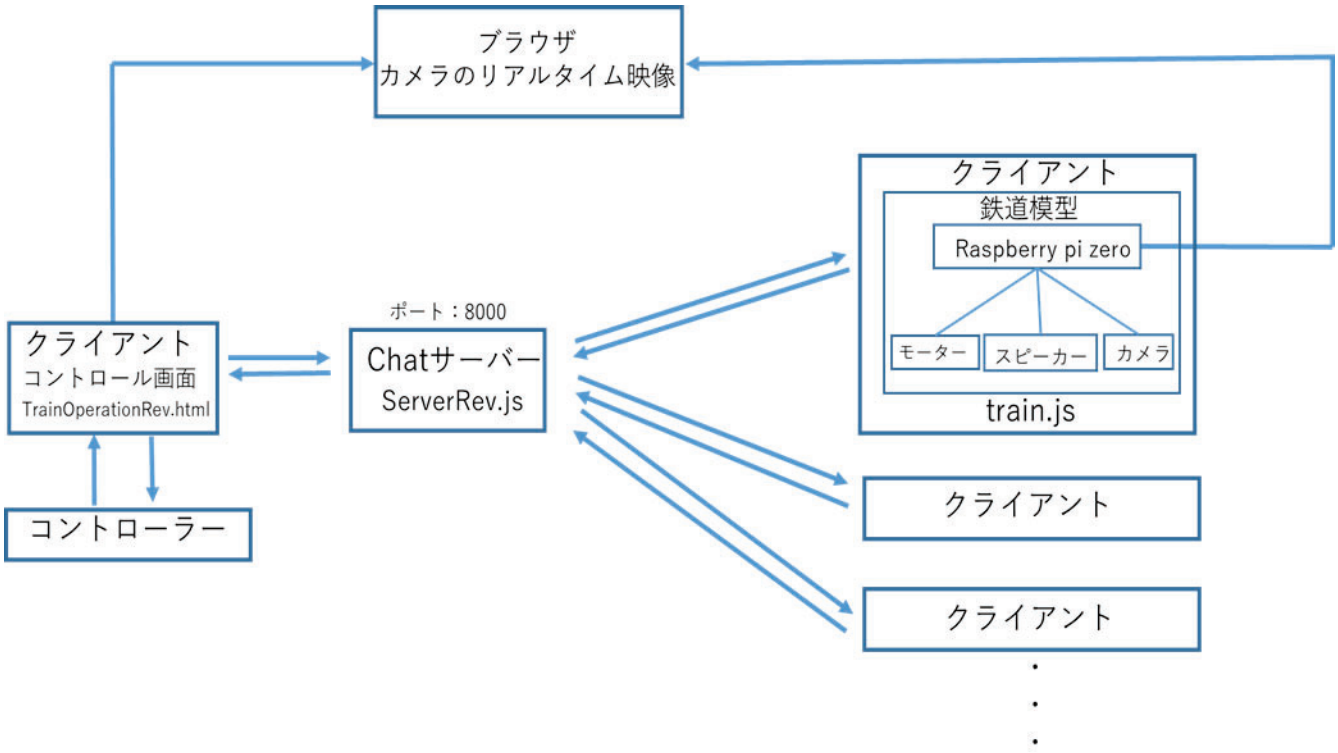

図 2 システム構成図

3. 実験の結果

3.1 有線，無線の遅延時間の測定内容

本実験では，有線，有線+ Wi-Fi，有線+LTE によるデータ遅延時間の測定を行った．

Wi-Fi には 2.4GHz 帯を使用している IEEE802.11n 規格と 5GHz 帯を使用している IEEE802.11ac，を使用した．

・実験方法

以下に具体的な実験のシーケンスを示す．

1. UTC で測定するプログラムを立ち上げる．
2. クリックを 350 回する．
3. Excel にデータを出力する．
4. 結果をグラフで表示し，比較する．

3.2 結果

実験は，2 回行った．1 回目はイントラネットの有線と 3km，460km 地点からの有線を使用した時の遅延時間を測定した（図 2-1）．2 回目は Wi-Fi の 5GHz 帯，2.4GHz 帯 LTE 使用時の遅延時間を計測した（図 2-2）．

1 回目のイントラネットの有線の遅延時間は最小:1ms. 最大:6ms になった．3km 地点は最小:27ms. 最大:365ms. 460km 地点では最小:20ms. 最大 27ms になった．

2 回目 Wi-Fi の 2.4GHz 帯では最小:2ms. 最大:117ms になった．また，5GHz 帯では最小:2ms. 最大:14ms になった．LTE では最小:52ms. 最大:1467ms となった．

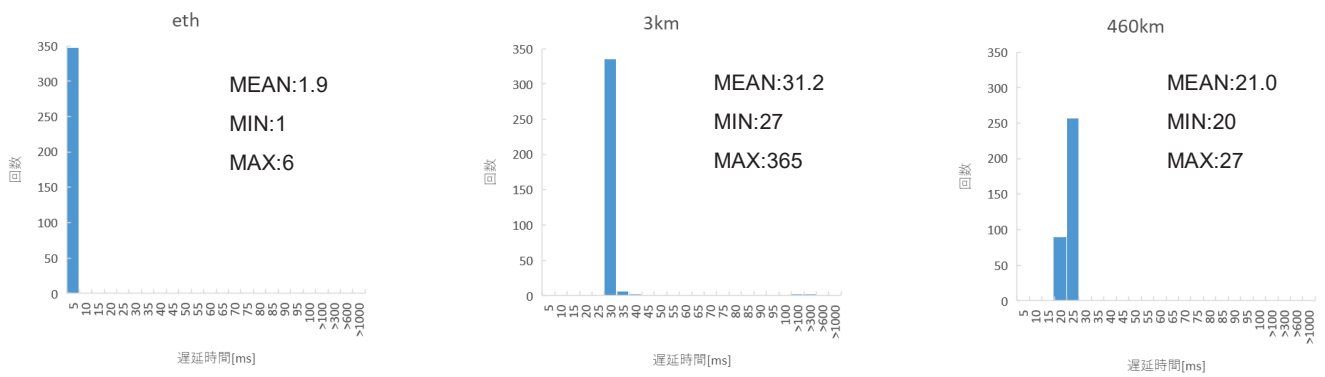

図 3-1 有線の遅延時間

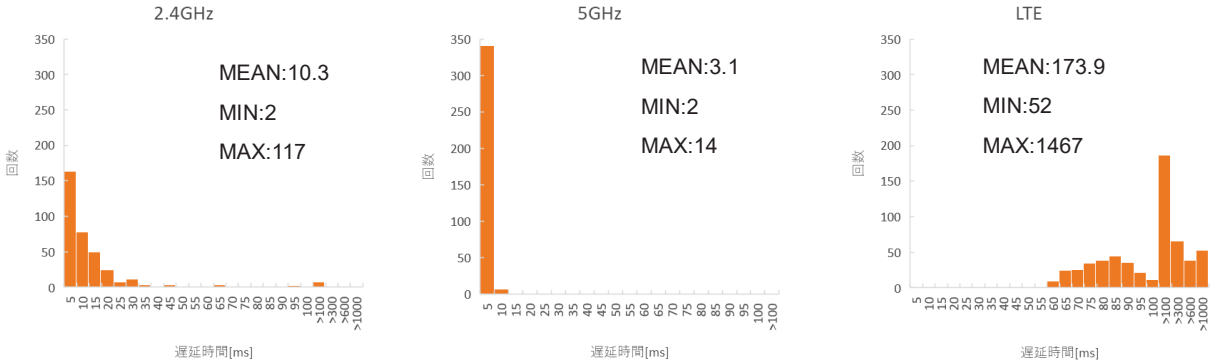

図 3-2　無線の遅延時間

3.3 考察

1 回目のグラフの 3km 地点と 460km 地点を比較すると，460km 地点の遅延時間の方が短いことが分かる．このことから，遅延時間は距離に依存せず、間に入ってくる，ルータの数で差が出てくることが考えられる．また，2 回目のグラフをから，Wi-Fi の 2.4GHz 帯と 5GHz 帯を比較すると，最初の遅延時間は同じだが，2.4GHz 帯の遅延時間はテールを引くように伸びている．これは，他のデータが入ってきた時の待ち時間が原因であると考えられる．また，5GHz 帯の遅延時間が短いのは，2.4GHz 帯と比較すると帯域幅が広いからだと考えられる．さらに，Wi-Fi と LTE を比較した際に LTE の遅延時間が長いのは，LTE は Wi-Fi に比べ，帯域幅が短いことが原因だと考えられる．

3.4 小型カメラの概要

解像度：500 万画素　重さ：1. 1g
大きさ：8.6mm×8.6mm　ケーブルの長さ：5 ㎝
・画像圧縮方式について
画像圧縮方式には主に 3 つの方式がある．（表 1）
この方式について説明していく

表 1．各種画像圧縮方式

	データ転送	画像処理	cpu 負荷
M-JPEG	50Mbps	○	10
H.264	5Mbps	×	20
H.265	1Mbps	×	40

1.M-JPEG

この方式は静止画一枚一枚に画像圧縮を行っていく方式である．利点として JPEG 画像以外の圧縮をしないため画質が劣化しにくく，CPU 負荷が小さいことである．また，1 枚ずつ圧縮するために前後のフレームに関係なく画像処理を行うことができる．

2．H.264

M-JPEG と比べてデータ転送が少ないためデータサイズを削減することができ画質は M-JPEG と変わらない．利点として JPEG 画像以外の圧縮をしないため画質が劣化しにくく，CPU 負荷が小さいことが利点である．画像処理を行うことができるが前後のフレームを参照にしないと静止画を生成することができないため処理時間が余分にかかる．

3..H265

H.264 と比べて約 2 倍の圧縮能力がある．つまり，より少ないデータ転送で H.264 同等の画質を実現できる．この３つの中で画像処理を行うことができ，CPU 負荷が少ない M-JPEG を選択し実装した．

3.5 画像データ遅延実験結果

本実験では Wi-Fi による画像配信時の画像データ遅延時間の測定を行う．2.4GHz 帯を使用している IEEE802.11n 規格と，5GHz 帯を使用している IEEE802.11ac.をそれぞれ用いた場合の遅延はどの程度あるかを実験した．

・実験方法
以下に具体的な実験のシーケンスを示す．
1．小型カメラでリアルタイム映像をパソコン画面に映す．
2. 小型カメラで実際のストップウォッチと映像の中のストップウォッチを 40 回ずつ撮影する．
3. 実際のストップウォッチと映像の中のストップウォッチの秒数差を分析する．
4．実験結果をヒストグラムで表示する．
カメラの設定は fps=30,q=10 にして実験を行った．

3.6 結果

実験は 2 回行った. 1 回目はカメラだけを使って遅延差を測った.（図 4-1） 2 回目は実際に鉄道模型を走らせている状態から測った.（図 4-2）

1 回目の 2.4GHz 帯と 5GHz 帯のグラフを比較すると 2.4GHz 帯の平均は 0.130 秒の遅延になり，5GHz 帯は 0.115 秒の遅延になった. 2 回目は 5GHz 帯の平均が 0.15 秒で 2.4GHz 帯の平均が 0.2 秒になった.

3.7 考察

1 回目と 2 回目のグラフを見比べるとカメラだけの実験では遅延時間の平均値が 5GHz 帯だと 115ms で 2.4GHz 帯だと 130ms である. 対して鉄道模型を走らせている状態での実験では 5GHz 帯の平均値は 150ms で 2.4GHz 帯は 200ms あることがわかる. この理由として考えられることは実際に動いている状態で実験しているためカメラだけの時と比べて通信が不安定になったのではないかと考える. また, 2.4GHz 帯と 5GHz 帯の遅延時間を見ると 1 回目も 2 回目も 5GHz 帯の遅延時間が短いことがわかる. 更に, 有線, 無線の遅延時間の測定結果と比較すると, 画像データの圧縮, 展開にかかる時間が有線, 無線等の遅延時間に加算されることが分かった.

4. 結論

今回は，Chat System を用いたロボット制御と遅延時間の測定を行った. ロボット制御では，今回時間の都合上，1 つのクライアントの制御を行っている. しかし, 原理上は数百，数千のクライアントにも対応することができるため，今後取り組んでいこうと考えている. また, 遅延時間の測定では, 有線，有線+ Wi-Fi，有線+LTE 等によるデータ遅延時間を測定した. 有線は，最小:1ms，最大:365ms だった. 一方で，有線+ Wi-Fi は，5GHz 帯の場合，最小:2ms，最大:14ms. 2.4GHz 帯の場合，最小:2ms，最大:117ms となった. 有線+LTE の場合，最小:52ms，最大:1467ms となった. また，5G は一般的に遅延時間が 1ms と言われているので, Ethernet と同様の遅延時間が期待できる. 今回は機器の準備ができなかったため，今後測定を予定している. 加えて，Wi-Fi による画像配信時の画像データ遅延時間の測定も行った. その結果, カメラだけでの遅延時間の場合，5GHz 帯だと平均 115ms で 2.4GHz だと平均 130ms である. 一方でロボットを動かしている状態だと，5GHz 帯の平均値は 150ms で 2.4GHz 帯は 200ms であった.

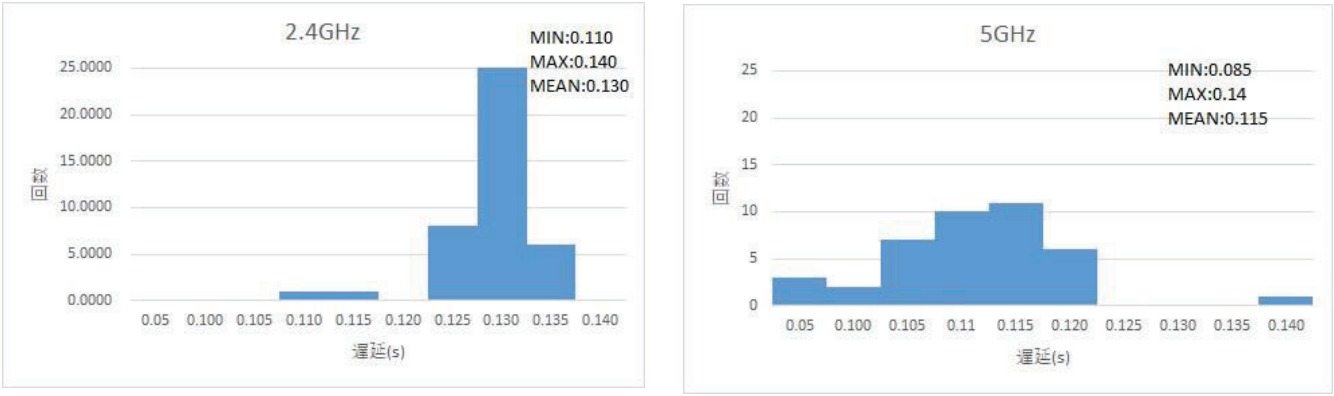

図 4-1 カメラだけの実験結果

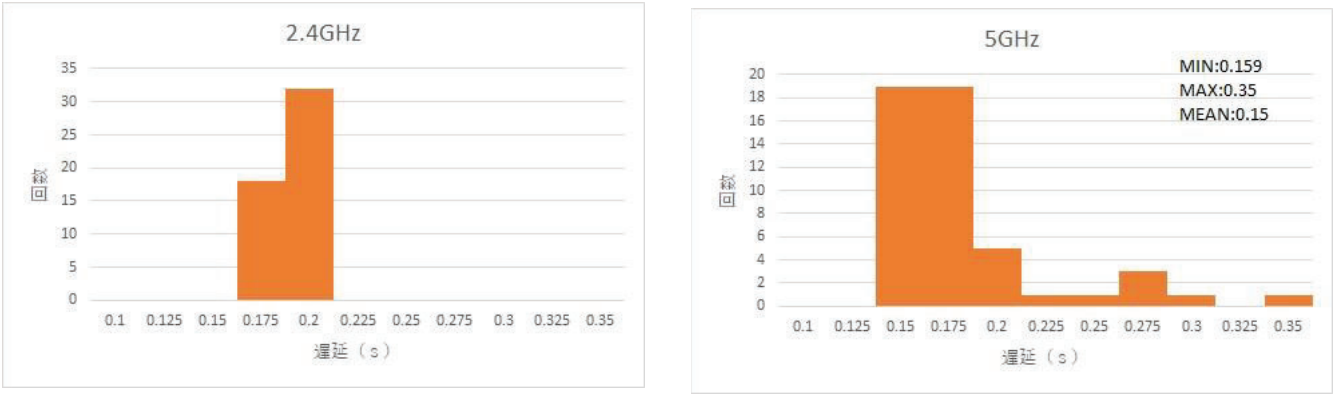

図 4-2 模型を走らせながら実験した結果

参考文献

[1]. 総務省. 人口減少時代の ICT による持続的成長. 2018

[2]. 総務省. 令和 2 年「情報通信に関する現状報告」. 2020

[3]. Beer, Randall and Chiel, Hillel and Quinn, Roger and Espenschied, Kenneth and Larsson, Patrik. A Distributed Neural Network Architecture for Hexapod Robot Locomotion. Neural Computation. 4. 1992. 356-365.

[4]. Fred Delcomyn,Mark E Nelson" Architectures for a biomimetic hexapod robot" Robotics and Autonomous SystemsVolume 30, Issues 1–2, 31 January 2000, Pages 5-15

[5]. M. a. E. A. Malmros, Artificial Intelligence and Terrain Handling of a Hexapod Robot, 2016.

[6]. M. I. Uddin, M. S. Alamgir, J. Chakrabarty, M. I. Hossain and M. A. Abdulla Samy, Multitasking Spider Hexapod Robot. International Conference on Robotics, Automation, Artificial-intelligence and Internet-of-Things (RAAICON), pp. 135-140, 2019 IEEE

[7]. 総務省 平成 30 年度 5G 総合実証試験の実施概要

[8]. 伊藤順治, 川原慎太郎, 市田幸生 IoT を用いた鉄道模型制御システムに関する研究 日本文理大学紀要第 48 巻第 2 号令和 2 年 10 月

[9]. C.-M. Huang, Proximate Sharing of Geo Data Downloading Based on the MSNP-Oriented Ubiquitous Machine-to-Machine (M2M) Communication Paradigm, IEEE Access Special Editorial, 2018.

[10]. A.D. S.S.Pulugurtha, Comparative Evaluation of Synchro and VISSIM Traffic Simulation Software to Model Railroad Crossings, International Workshop on Computing in Civil Engineering 2007, 2007.

[11]. H. Shan, G. Jiang , K. Yoshihira, Extracting Overlay Invariants of Distributed Systems for Autonomic System Management, 2010 Fourth IEEE International Conference on Self-Adaptive and Self-Organizing Systems, 2008.

[12]. S. M. Babamir, M2M Architecture: Can It Realize, vol. 6, KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS VOL. 6, NO. 2, Feb 2012, 2012, pp. 566-578.

【 この位置に改ページを入れ, 以降のページを印刷対象外とする 】

1 総務省. 人口減少時代の ICT による持続的成長. 2018
2 総務省. 令和 2 年「情報通信に関する現状報告」. 2020
3 Beer, Randall and Chiel, Hillel and Quinn, Roger and Espenschied, Kenneth and Larsson, Patrik. A Distributed Neural Network Architecture for Hexapod Robot Locomotion. Neural Computation. 4. 1992. 356-365.
4 Fred Delcomyn,Mark E Nelson" Architectures for a biomimetic hexapod robot" Robotics and Autonomous Systems Volume 30, Issues 1–2, 31 January 2000, Pages 5-15

5 M. a. E. A. Malmros, Artificial Intelligence and Terrain Handling of a Hexapod Robot, 2016.
6 M. I. Uddin, M. S. Alamgir, J. Chakrabarty, M. I. Hossain and M. A. Abdulla Samy, Multitasking Spider Hexapod Robot. International Conference on Robotics, Automation, Artificial-intelligence and Internet-of-Things (RAAICON), pp. 135-140, 2019 IEEE

7 総務省 平成 30 年度 5G 総合実証試験の実施概要
8 伊藤順治, 川原慎太郎, 市田幸生 IoT を用いた鉄道模型制御システムに関する研究 日本文理大学紀要第 48 巻第 2 号令和 2 年 10 月
9 C.-M. Huang, Proximate Sharing of Geo Data Downloading Based on the MSNP-Oriented Ubiquitous Machine-to-Machine (M2M) Communication Paradigm, IEEE Access Special Editorial, 2018.
10 A. D. S.S.Pulugurtha, Comparative Evaluation of Synchro and VISSIM Traffic Simulation Software to Model Railroad Crossings, International Workshop on Computing in Civil Engineering 2007, 2007.
11 H. Shan, G. Jiang , K. Yoshihira, Extracting Overlay Invariants of Distributed Systems for Autonomic System Management, 2010 Fourth IEEE International Conference on Self-Adaptive and Self-Organizing Systems, 2008.
12 S. M. Babamir, M2M Architecture: Can It Realize, vol. 6, KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS VOL. 6, NO. 2, Feb 2012, 2012, pp. 566-578.