ROS準拠ロボット及びエッジを用いた 環境情報収集・ストリーム処理を行うIoTシステムの構築と評価

佐々木怜名† 竹房あつ子†† 中田 秀基††† 小口 正人†

† お茶の水女子大学 〒112–8610 東京都文京区大塚 2–1–1
†† 国立情報学研究所 〒101–8430 東京都千代田区一ツ橋 2–1–2
††† 産業技術総合研究所 〒305–8560 茨城県つくば市梅園 1–1–1
E-mail: †reina@ogl.is.ocha.ac.jp, ††takefusa@nii.ac.jp, †††hide-nakada@aist.go.jp, ††††oguchi@is.ocha.ac.jp

あらまし IoT 機器に装備されたセンサによって収集したデータを活用して，お年寄りやペットの見守り，室内環境監視などを目的とした，スマートホームのためのサービスが実現されている．個々の家庭のデータをクラウドに収集する IoT システムを構築するには，通信遅延の低減，転送データ量の削減，プライバシの保護への対策も必要となる．室内環境で多様なデータを収集する場合，屋内の複数箇所にセンサを設置し，複数センサの情報を収集する必要がある．しかし，一般家庭に多数のセンサを設置するとコストが高くなるだけでなく，必要な情報を得るためのセンサの再配置も容易ではない．本研究では，ROS で実装された車輪型移動ロボットを用いて室内環境情報を収集し，エッジを介してクラウド上での解析処理を行う，スマートホームのための IoT システムの構築と評価を行う．さらに，構築したシステムを活用して室内二酸化炭素濃度監視アプリケーションを試作するとともに，センサロボットとエッジ間の ROS 通信，エッジとクラウド間の Kafka 及び MQTT 通信の性能特性を調査する．

キーワード IoT，スマートホーム，クラウドロボティクス，ROS

1 はじめに

ネットワークに接続した家電や住宅設備などの IoT 機器の普及が進んでいる．それに伴い，複数の IoT 機器を活用して，防犯サービスやペット・お年寄りの見守りサービスや家電の省電力化管理，温度，湿度等の室内生活環境の監視などを行う，スマートホームと呼ばれるサービスの普及が期待されている．スマートホームは，生活の質向上や効率化を目的とし，多種複数のセンサから収集したデータをクラウドで蓄積し，解析を行う．しかし，スマートホームを実現する IoT システムを構築するには，対処しなければならない課題が幾つか存在する．ネットワークに接続したセンサデバイスを空間内に分散して配置し，センサデータを収集する必要があるが，個別の間取りを考慮したり，必要に応じて IoT 機器を再配置しなければならない．また，センサから得られる情報は大量のストリームデータであり，遠隔のクラウドへ送信すると，応答時間や通信量が大幅に増大する．個々の家庭で収集したデータをクラウドに送信し解析処理を行うシステムを構築するためには，必要に応じて転送データ量を削減する必要がある．さらに，個人に関わるデータを収集するため，プライバシの保護への配慮やサイバー攻撃への対策も必要である．

本研究では，性能，プライバシ保護，セキュリティ対策を考慮した，スマートホームのための IoT システムの検討を行う．ROS (Robot Operating System) で実装された車輪型移動ロボットをセンサ端末として，駆動機能を用いて室内環境情報を収集し，エッジを介して，クラウド上で解析処理を行う．エッジでは，必要に応じてデータの前処理や一部の解析処理を行う．クラウドへの収集には IoT の標準的な通信プロトコルをサポートし，暗号化等の機能を提供する SINETStream を用いる．我々は，センサロボットからエッジへ適切にセンサデータを収集する手法を検討し，センサロボットとエッジで，CO2 濃度と位置情報を収集・可視化するアプリケーションを試作した [1]．本稿では，提案する IoT システムにおいて，センサロボットからクラウドへ，エッジを介して適切にセンサデータを収集する手法を検討するため，センサロボットによるカメラ画像と環境情報データの収集と，ストリーム処理に向けた試行実験を行う．また，ロボットを活用したスマートホームアプリケーションの実装に向けて，センサロボットを室内空間で走行させ，任意の場所の CO2 濃度を監視し，リアルタイムで可視化するアニメーションシステムも試作する．さらに，提案する IoT システムの評価に向けて，カメラ画像送信時の遅延時間とスループットを計測する．

2 提案システムの概要

本稿で提案する車輪型移動ロボットを活用したスマートホーム向け IoT システムの概要について述べる．まず，提案する IoT システムのシステムモデルについて述べた後で，提案システムを構成する技術である ROS と SINETStream について説明する．

2.1 IoT システムモデル

本研究では，ROS で実装された車輪型移動ロボットをセンサ

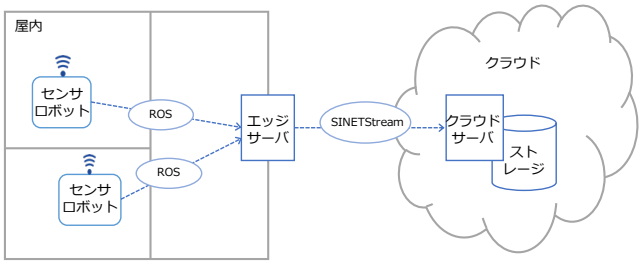


図 1: 提案する IoT システムの概要

端末とし，エッジ，クラウドサーバで構成される IoT システムを検討する．図 1 に想定する IoT システムの概要を示す．屋内を移動する複数センサロボットから収集されるセンサデータをエッジに集約し，エッジからクラウドサーバに集約したデータを送信する．エッジでは，収集したセンサデータの間引き，匿名化等のフィルタ処理を行うとともに，グローバルネットワークから室内ネットワークを分離する役割も担う．センサロボットとエッジ間は室内 Wi-Fi で接続し，ROS でデータ通信を行う．エッジとクラウドサーバ間は MQTT 等の IoT における標準的な通信プロトコルを用い，通信プロトコル層を抽象化する SINETStream で実装する．SINETStream を用いることで MQTT や Apache Kafka 等のメッセージブローカに対応できるだけでなく，通信やデータの暗号化，データの圧縮といった機能も利用できるようになる．ROS はロボット制御のためのライブラリが充実している一方，TLS 等のセキュリティ対策はされておらず，セキュリティ上の課題がある．また，ROS で流れるすべてのデータを収集するには通信性能の点で課題があり，必要なデータを取捨選択して適宜加工できるようにするためにこのような構成とした．以降で，提案する IoT システムの構成技術である ROS と SINETStream の概要について述べる．

2.2 ROS

ROS (Robot Operating System) はロボット開発を支援するためのライブラリや通信の仕組みを提供するオープンソースのロボットソフトウェアプラットフォームで，既存の OS 上で動作する．ROS の特徴を表す四要素として plumbing (通信)，tools (ツール群)，capabilities (機能群)，ecosystem (エコシステム) が挙げられている [2]．ROS では，センサやアクチュエータ等のハードウェアをモジュール化し，モジュール間のインタフェースの統一によりソフトウェアとハードウェア間の通信の管理が自動化された，Pub-Sub 型のメッセージング通信システムが提供されている (plumbing)．これにより関心のある部分にのみ集中してシステムの開発が可能となる．またシステムの起動・停止・デバッグ・可視化・ログ取得等の多様なツールや移動・マニピュレーション・知覚といった機能をロボットに実装する多様なライブラリ群など，ロボットに必要とされる機能をまとめたライブラリ/パッケージや開発効率を促進するツールが提供されており，開発コストを低減し，容易に機能を拡張できるという利点がある (tools/capabilities)．これは rviz 可視化ツール,Gazebo シミュレータ，MoveIt 軌道計算ライブラリが例に挙げられる．そして ROS はインテグレーションとドキュメーションに重点を置いた大規模なコミュニティによって支えられ発展している．ros.org [3] では世界中の開発者が提供した ROS パッケージが集約されており適宜活用していくことができる (ecosystem)．これらの機能を有した ROS を用いることで効率的でコストを抑えたロボット開発が可能となる．

表 1: センサロボットの構成

機種名	Raspberry Pi 4 Computer Model B 4GB RAM
CPU	ARMv7 Processor rev 3(v7l)
OS	Raspbian GNU/Linux 10
搭載機器	RplidarA1 レーダー/IMU/HD カメラ
ROS	Kinetic

2.3 SINETStream

広域に分散したデータを活用する研究では，広域ネットワークを介して，センサ等から取得されるデータを欠損なく確実に収集し，解析に用いることが求められている．本研究では，エッジからクラウドへのセンサデータ収集に SINETStream [4], [5] の利用を検討している．SINETStream は，国立情報学研究所で開発された広域データ収集・解析プログラム開発支援ソフトウェアパッケージであり，安全確実なデータ収集，認証・認可，圧縮/解凍，暗号化，メトリクス収集機能といった，IoT データの収集・蓄積・解析に必要な機能が予め提供されている．また，IoT 通信プロトコルを抽象化する API が提供されているため，センサデータの通信パターンに応じて適宜バックエンドに配備するメッセージブローカを選択できるという利点もある．

3 ストリーム処理に向けたセンサデータ転送方法の検討

本章では，提案する IoT システムにおける，カメラ画像と環境情報・位置情報の処理方法について検討する．はじめに，本研究で使用するセンサロボット，エッジの構成を紹介する．センサロボットには ROS ベースの車輪型移動ロボットを用い，エッジには Surface Pro4 を用いる．次に，センサロボットからエッジを介しクラウドへ適切なセンサデータを収集する手法を検討するため，センサロボットからエッジへのカメラ画像・環境情報・位置情報の収集の 3 つの試行実験を行う．また，ロボットを活用したスマートホームアプリケーションを実証するため，センサロボットを室内空間で走行させて任意の場所の CO2 濃度データを収集し，リアルタイムで可視化するシステムを試作する．

3.1 実 験 環 境

提案する IoT システムの構成要素である，センサロボット，エッジ，クラウドの詳細を表 1，表 2，表 3 に示す．センサロボットには，ROS ベースの車輪型移動ロボット Raspberry Pi ROS SLAM Robot (XiaoR Geek) を用いる [6]．このセンサロボットは，LIDAR SLAM [7] を用いて人為的に設定された初期位置と目的位置に合わせて自律走行を行うナビゲーションや，

表 2: エッジの構成

機種名	Surface Pro 4
プロセッサ	Intel®Core™i7-6650U CPU@2.20Ghz × 4
OS	Ubuntu 20.04.4 LTS
メモリ	15.6GiB
ROS	Noetic

表 3: mdx VM を用いたクラウドサーバの構成

OS	Ubuntu 20.04.5 LTS
仮想 CPU コア数	16
メモリ	24.19GB
仮想ディスクサイズ	100GB
Broker ソフト	Apache Kafka / Eclipse Mosquitto
バージョン	3.1.0 / 1.6.9

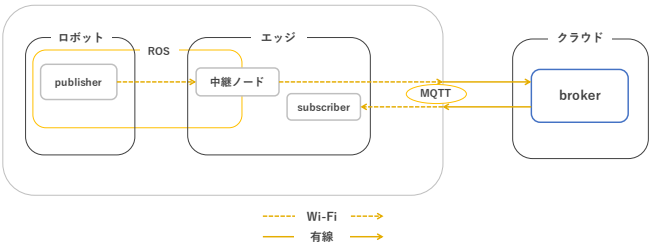


図 2: センサデータの送受信の概要

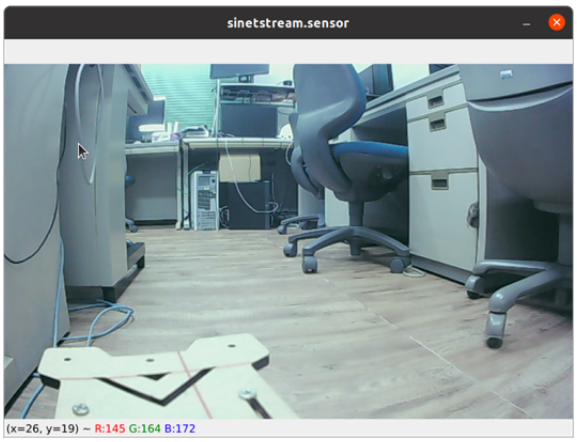


図 3: 転送されたカメラ画像をエッジに標準出力した時のディスプレイのスクリーンショット

その間の障害物検知を行うことができる．センサロボットと通信するエッジには，Ubuntu20.04 をインストールした Surface Pro4 を用いる．センサロボットとエッジが接続するアクセスポイントとして Buffalo WSR-2533DHP2-CG を使用し，センサロボットは IEEE802.11n(256QAM)，エッジは IEEE802.11ac の規格で接続している．クラウドは，データ活用社会創成プラットフォーム mdx [8] を用いる．

3.2 カメラ画像/環境情報/位置情報の転送

センサロボットでカメラ画像，環境情報，位置情報を収集し，エッジを介して，クラウドへ送信するメッセージ通信を行った．センサロボットには，温度・湿度・CO2 濃度を測定するセンサ

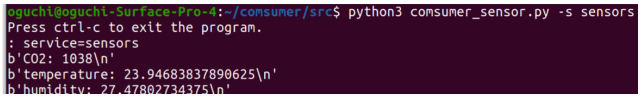


図 4: 転送された環境情報を操作用端末に標準出力した時のディスプレイのスクリーンショット

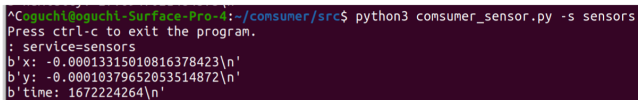


図 5: 転送された位置情報をエッジに標準出力した時のディスプレイのスクリーンショット

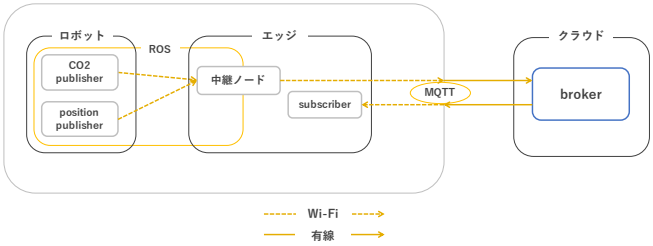


図 6: CO2 濃度可視化アプリケーションの概要

SEK-SCD41（SENSIRION) [9] を新たに搭載する．位置情報は，センサロボット Raspberry Pi ROS SLAM Robot (XiaoR Geek) のプログラムで推定されたオドメトリ情報を利用する．オドメトリでは，センサロボットの左右の車輪の回転数や角度をサブスクライブし，現在のセンサロボットの位置を算出する．計算されたセンサロボットの位置は，ROS 通信環境で/odom トピックとしてパブリッシュされているため，本研究で開発した ROS プログラムで/odom をサブスクライブしてセンサロボットの位置情報/odom/pose/pose/position を取得する．

センサデータ送受信時のノード関係を図 2 に示す．センサロボットで，各情報を取得し該当トピックに ROS でパブリッシュする publisher ノードを立ち上げる．エッジで，該当トピックから各情報を ROS でサブスクライブし，クラウド上のブローカに各情報を MQTT でパブリッシュする，中継ノードを立ち上げる．さらにエッジで，クラウド上のブローカから各情報を MQTT でサブスクライブする subscriber ノードを立ち上げる．MQTT は IoT の標準的な通信プロトコルである．

中継ノードと subscriber ノードで，サブスクライブした各情報を標準出力し，センサロボットで収集した任意のセンサデータを，エッジを介してクラウドに送信できることを確認した．図 3，図 4，図 5 に，センサロボット上の publisher ノードから，エッジ上の subscriber ノードに画像およびセンサデータを転送しディスプレイ上に表示した時のスクリーンショットを示す．これによりセンサロボットで収集したカメラ画像，環境情報，位置情報をエッジを介してクラウドまで送信可能であることを確認した．

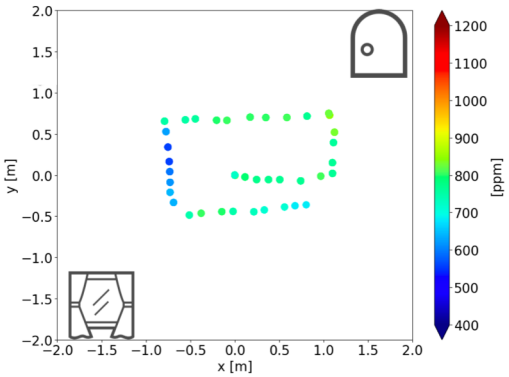


図 7: 換気前：室内環境における CO2 濃度の数値を可視化したヒートマップ

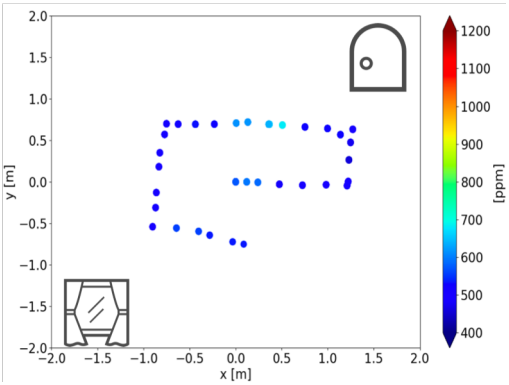


図 8: 換気後：室内環境における CO2 濃度の数値を可視化したヒートマップ

3.3 ロボットを活用した CO2 濃度可視化アプリケーションの試作

3.3.1 アプリケーションの試作

ロボットを活用したスマートホームアプリケーションの実装のため，室内環境でセンサロボットの遠隔操作を行いながら環境情報を収集し，収集データをリアルタイムで可視化する実験を行った．実験では，3.2 節で開発した publisher ノードで収集した環境情報から CO2 濃度値を抽出し，その値を位置情報と照合して 2 次元のヒートマップを作成する．図 6 に本アプリケーションの概要を示す．本研究では，CO2 濃度と収集時点のタイムスタンプを ROS で取得する CO2_publisher ノード，センサロボットの座標と収集時点のタイムスタンプを ROS で取得する position_publisher ノード，CO2 濃度と座標をタイムスタンプで照合し，照合した時刻と CO2 濃度のデータをクラウド上のブローカへ MQTT で送信する中継ノード，照合データをクラウド上のブローカから MQTT でサブスクライブし，リアルタイムでヒートマップを描画する subscriber ノード，計 4 つのノードを作成した．

3.3.2 CO2 濃度可視化結果

図 7，図 8 に CO2 濃度を可視化したヒートマップを示す．図 7 は換気を行っていない密閉状態の室内空間で測定した時のものである．図 8 は図 7 の状態から換気を行った後，同一環境で測定した結果である．この室内環境では，右上の位置にドア，左下の位置に窓があり，ドアと窓を開けて換気した．換気前と比較して換気後の CO2 濃度の数値が減少していることがわかる．固定されたセンサの場合は，室内に複数のセンサを設置しなければ室内全体の CO2 濃度を把握することができないが，ロボットセンサを活用して適宜観測点を動かすことで室内の CO2 濃度を面で把握することができ，ロボットセンサを利用した IoT システムの有用性が確認できた．

表 4: ロボット-エッジ間における iPerf3 測定結果

	Interval	Transfer	Bitrate
sender	0.00-10.0 sec	27.2MBytes	22.8Mbits/sec
receiver	0.00-10.5 sec	26.8MBytes	22.3Mbits/sec

表 5: エッジ-クラウド間における iPerf3 測定結果

	Interval	Transfer	Bitrate
sender	0.00-10.0 sec	72.0MBytes	60.4Mbits/sec
receiver	0.00-10.7 sec	69.7MBytes	58.0Mbits/sec

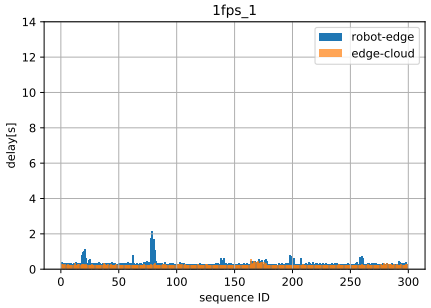
4 システムの性能評価

本章では，提案する IoT システムにおける，カメラ画像転送時の遅延時間とスループットを計測する．3.2 節で実装した 3 つのノードを用いて，センサロボットとエッジ間の ROS 通信，エッジとクラウド間の Kafka を用いた通信の遅延時間とシステム全体のスループットを計測する．センサロボットではカメラ画像，カメラ画像に付与した通し番号，カメラ画像取得時のタイムスタンプを収集する．収集した 3 つのデータを publisher ノードから中継ノードへ転送し，カメラ画像を中継ノードから subscriber ノードへ，5 分間転送する．publisher ノードの送信周期，すなわち fps 値を 1，2，3，4，5，10 で変化させる．各 fps 値につき 10 回転送実験を行う．画像の大きさは 640px×480px，画像のファイルサイズは 1 枚あたり約 500KB である．無線環境のパフォーマンスを確認するため，iPerf3 [10] を用いてセンサロボットとエッジ間，エッジとクラウド間のスループットを計測した．結果を表 4 表 5 に示す．表 4 表 5 から，ロボットとエッジ間より，エッジとクラウド間の方が通信スループットが高く，画像のファイルサイズが 500KB のときの最大 fps 値は約 5.6 と想定される．センサロボットとエッジの時刻合わせを行い，さらにカメラ画像転送と同時にセンサロボットとエッジのクロックの差を 10 秒おきに測定するプログラムも動作させる．表 6 にセンサロボットとエッジのクロックの差の平均値を示す．最大 200msec 程度の誤差が生じていた．

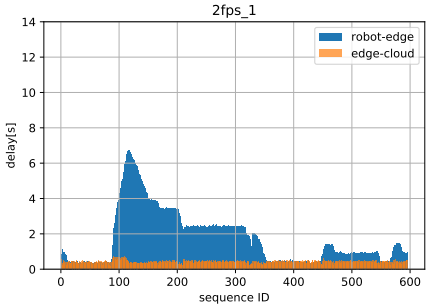
4.1 実験結果

4.1.1 遅延時間の比較

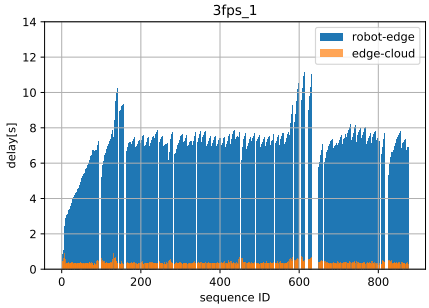
センサロボットとエッジ間，エッジとクラウド間の遅延時間について，10 回転送実験行ったうちの 1 実験を図 9 に示す．横軸はカメラ画像に付与した通し番号を示し，縦軸は遅延時間を秒単位で示す．青色の棒グラフはセンサロボットとエッジ間の遅延時間を，橙色の棒グラフはエッジとクラウド間の遅延時間



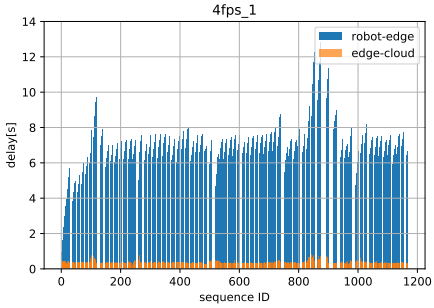
(a) 1fps



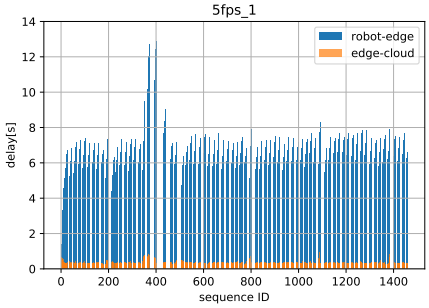
(b) 2fps



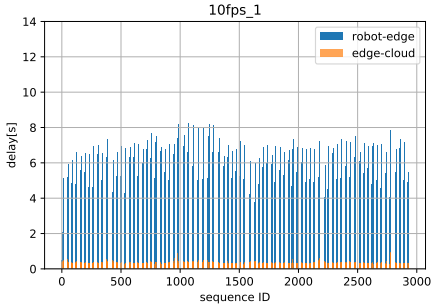
(c) 3fps



(d) 4fps



(e) 5fps



(f) 10fps

図 9: センサロボットとエッジ間の ROS 通信とエッジとクラウド間の Kafka を用いた通信の遅延時間

表 6: ロボット-エッジ間のクロックの差の平均

1fps	2fps	3fps	4fps	5fps	10fps
-5.75×10^{-3}	7.54×10^{-3}	1.74×10^{-2}	1.90×10^{-2}	1.99×10^{-2}	1.34×10^{-2}

表 7: ロボットとエッジ間での欠損データの数と割合

	1fps	2fps	3fps	4fps	5fps	10fps
ロボットからの送信画像枚数	2985	8939	17652	29307	43879	73056
欠損データの数	0	5	1658	6166	13488	35014
欠損データの割合 [%]	0.000	0.099	9.40	21.0	30.7	47.9

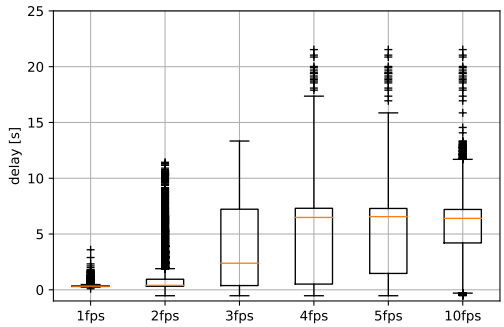


図 10: ロボット-エッジ間の ROS 通信の遅延時間

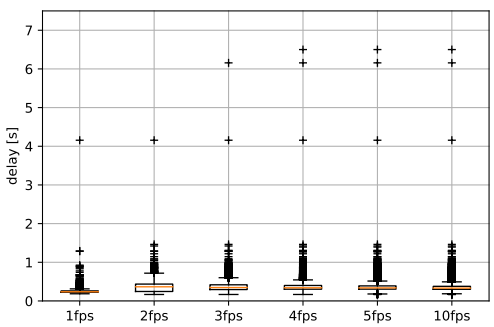


図 11: エッジ-クラウド間の Kafka を用いた通信の遅延時間

を示す．二つの棒グラフは積み上げグラフではない．表 7 に，10 回の転送実験における，ロボットから送信した総画像枚数，及びロボットとエッジ間での欠損データの数と割合を示す．センサロボットとエッジ間の遅延時間は，図 9 の青の棒グラフと表 7 より，fps 値が大きくなると遅延時間が増加し，fps 値が 3 以上の時は一定周期で遅延時間の増加とデータの欠損を繰り返すことが分かる．また，iPerf3 から推定された最大 fps 値の理論値よりも低い場合も，遅延時間の増加やデータの欠損が発生

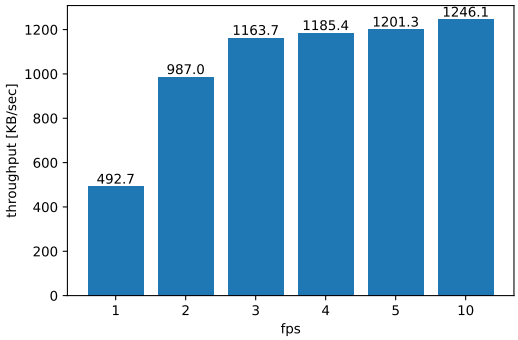


図 12: システムのスループット

しており，iPerf3 より ROS 通信のオーバーヘッドが大きいことが分かった．

図 10 に，10 回の転送実験におけるセンサロボットとエッジ間の遅延時間の分布を, 図 11 に，10 回の転送実験におけるエッジとクラウド間の遅延時間の分布を示す．図 10，図 11 について，箱の下端が第一四分位数，箱の橙色の線が中央値，箱の上端が第三四分位数を示す．また，ひげの下端が第一四分位から四分位範囲の 1.5 倍以内の最小値，ひげの上端が第三四分位から四分位範囲の 1.5 倍以内の最大値を示し，ひげの範囲外の外れ値を “+”で示す．図 10 より遅延時間は，fps 値が 3 以上の時，データの分布が広がり，fps 値が 4 を境にデータの分布が縮んでいる．fps 値が 4 以上の時，中央値や最大値が定常になることが分かる．これらは，fps 値が大きくなるほど，遅延時間の大きいフレームの数が増大していることが分かる．さらに，負の値の遅延時間をとっていることが分かる．これはセンサロボットとエッジの測定中に生じたクロックの差が原因の一つにあると考えられる．

エッジ-クラウド間の遅延時間は，図 9 の橙色の棒グラフと図 11 より，fps 値による遅延時間の中央値や分布の大きな変化がないことが分かる．

4.1.2 総スループットの比較

図 12 に，ロボットからエッジを経由してクラウドに通信を行った時の，システム全体の平均スループットを示す．転送データ量が多くなると Wi-Fi 環境での ROS 通信のスループットが飽和することがわかる．以上の実験から，グローバルネットワークより室内 Wi-Fi 環境の方に通信のボトルネックがあることがあり，特に画像を活用するようなロボットセンサでは室内無線環境の広帯域化が不可欠であることが示された．

5 関 連 研 究

スマートホームのシステム構築に向けた研究は盛んに行われている．鯵坂ら [11] は複数のセンサデバイスを空間内に分散配置し，知的制御を行うための推論器を複数のレイヤに分けて実装し，実用的で効率的に構築可能なシステムを提案している．ユーザとの対話とセンシング機能，対話とセンシングで得られた情報を基にする推論機能，推論機能で決定されたアクチュエーションを行う制御機能の，3 つの機能を備えたシステムの実現に向け，推論器をデバイス内，ローカルネットワーク内，クラウドネットワーク内の複数レイヤに分けて実装し，安定で効率的な推論を行うことを可能にしている．また．ROS を使用してアプリケーションの実装を行うことで，開発の容易性と拡張性を確保している．

ロボットが地図作成，物体把持，物体認識，位置推定などの処理を行う場合，センサから大量の情報を収集し，膨大な計算を実時間で処理することが要求される．しかし，ロボット単体で処理を行うためには，高性能な計算機の搭載が不可欠となり，ロボットの大型化，高価格化，高電力化に繋がる．そこで，ロボットとクラウドを連携させ，大規模な計算やデータベースを要求するような処理をクラウド上で担う，クラウドロボティクスと呼ばれる考えが普及している．G. Mohanarajah らは，複数のロボットによるクラウドロボティクスのオープンソースプラットフォームである Rapyta を実装した [12]．Rapyuta は，安全でカスタマイズ可能なコンピューティング環境をクラウド上で提供している．ロボットは膨大な計算をクラウドにオフロードし，他のロボットと結果を共有することで，マルチロボットの協調的制御を実現している．Rapyuta は ROS と互換性のある環境を用意しており，ほぼ全てのオープンソース ROS パッケージを変更することなく動作可能である．

移動ロボットによる物体認識システムにおいて，膨大な通信能力や計算能力，記憶能力が必要となる．従来はクラウドコンピューティング技術を用いて．計算リソースやストレージを確保してきたが，通信遅延やサービスの中断，プライバシの問題の観点から，クラウドコンピューティングの代わりにエッジノードを用いる手法が提案されている．Meixia Fu ら [13] は移動ロボットの物体認識を行う，ロボット・フォグ・クラウドで構成されたシステムを考案した．ロボットから収集した視覚情報を Wi-Fi ルータを経由させフォグノードに送信する．フォグは Wi-Fi ルータ・コンピュータ・GPU などの共通設備を持つ多数のフォグノードから構成され，クラウドやロボットと接続しサービスを提供している．フォグノードはロボットから収集したビデオデータを処理するための計算機能も有している．クラウドは，様々な IoT 要件に対応したフォグノードと接続し，リソースのスケジューリングとコンピューティングタスクの割り当てを行い，処理された情報をフォグノードに返す．このシステムによって人間レベルの精度で画像処理が可能となり，またロボットに人間のような周囲のものを認識させる物体認識システムが実現できる．

ロボットアプリケーションには ROS が広く利用されており，リアルタイム分散組込みシステムへの活用に向けて，ROS や ROS2 のリアルタイム性が評価されている [14]．ROS や ROS2 におけるノード間の様々な通信状況について，レイテンシやスループットなどを測定している．また，ROS2 では DDS（Data Distribution Service）を採用しており，DDS のベンダーや構成による ROS2 の性能評価に向けて，異なるデータサイズ及び同一周期でメッセージを通信し，レイテンシ/スループット/スレッド数/メモリ消費量を測定している．本研究では，スマー

トホームの IoT 機器に不可欠であるカメラを活用したセンサロボットの実装に向けて，画像を用いた一定のデータサイズ及び異なる転送レートでのメッセージ通信の評価を行っている．

6 まとめと今後の課題

センサが搭載された車輪型移動ロボットを用いて任意の空間で動的に環境情報を収集し，エッジを介してクラウドで解析処理を行う，スマートホームのための IoT システムを構築した．ROS 準拠センサロボットからエッジを介しクラウドへ適切なセンサデータを収集する手法を検討するため，センサロボットからクラウドへの動画像データ，環境情報データ，位置情報データの収集を行った．センサロボットから収集した CO2 濃度データを用いて室内空間の CO2 濃度の数値をリアルタイムで可視化するヒートマップを作成し，センサロボットを用いたスマートホームアプリケーションの有効性を示した．最後に，提案する IoT システムの評価に向けて，カメラ画像転送時の遅延時間とスループットの計測をした．センサロボットとエッジ間におけるカメラ画像転送では，転送レートが大きくなるほど，遅延時間やデータの欠損が増加し，エッジとクラウド間の通信より，センサロボットとエッジ間の Wi-Fi 環境での ROS 通信のスループットが飽和することが分かった．

今後の課題は，オーバーヘッドの原因の特定に向けて，有線通信を用いたカメラ画像の転送実験，異なるメッセージサイズでの転送実験，及び ROS と ROS2 の通信を比較した転送実験を行う．また，提案システムの性能評価として，作成した CO2 濃度可視化アプリケーション実行時における，遅延時間やスループットの測定を行う．さらに，効率的な環境情報収集に向けて，複数センサロボットの適切な制御方法を検討する．

謝　　辞

本研究は JSPS 科研費 JP19H04089 及び 2022 年度国立情報学研究所公募型共同研究 (22S0203) の助成を受けたものです．

文　　献

[1] 佐々木怜名, 竹房あつ子, 中田秀基, 小口正人. ROS 準拠ロボット及びエッジサーバを活用した環境情報収集・処理を行う IoT システムの検討. In *DICOMO2022*, 第 2022 巻, pp. 1505–1511, 2022.

[2] B.Gerkey. What is ros exactly? Middleware, Framework, Operating System? `https://answers.ros.org/question/12230/what-is-ros-exactly-middleware-framework-operating-system/`, December 2011.

[3] ROS wiki. `http://wiki.ros.org/`.

[4] SINETStream. `https://sinetstream.net/`.

[5] Atsuko Takefusa, Jingtao Sun, Ikki Fujiwara, Hiroshi Yoshida, Kento Aida, and Calton Pu. SINETStream: Enabling research iot applications with portability, security and performance requirements. In *Proc. COMPSAC 2021*, pp. 482–492, 2021.

[6] XiaO GEEK. `http://www.xiaorgeek.com/Study/Study/catalog/cid/35`.

[7] Wolfgang Hess, Damon Kohler, Holger Rapp, and Andor Daniel. Real-time loop closure in 2d lidar slam. In *In 2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1271–1278, 2016.

[8] Toyotaro Suzumura, Akiyoshi Sugiki, Hiroyuki Takizawa, Akira Imakura, Hiroshi Nakamura, Kenjiro Taura, Tomohiro Kudoh, Toshihiro Hanawa, Yuji Sekiya, Hiroki Kobayashi, Yohei Kuga, Ryo Nakamura, Renhe Jiang, Junya Kawase, Masatoshi Hanai, Hiroshi Miyazaki, Tsutomu Ishizaki, Daisuke Shimotoku, Daisuke Miyamoto, Kento Aida, Atsuko Takefusa, Takashi Kurimoto, Koji Sasayama, Naoya Kitagawa, Ikki Fujiwara, Yusuke Tanimura, Takayuki Aoki, Toshio Endo, Satoshi Ohshima, Keiichiro Fukazawa, Susumu Date, and Toshihiro Uchibayashi. mdx: A cloud platform for supporting data science and cross-disciplinary research collaborations. In *2022 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, pp. 1–7, 2022.

[9] SENSIRION. SEK-SCD41. `https://sensirion.com/jp/products/product-catalog/SEK-SCD41/`.

[10] iPerf3. `https://iperf.fr/`.

[11] 鰺坂志門, 中村壮亮. マルチレイヤ AI を用いた分散型スマートハウスの実装. 知能と情報（日本知能情報ファジィ学会誌）, Vol. 33, No. 1, pp. 572–581, 2021.

[12] Gajamohan Mohanarajah, Dominique Hunziker, Raffaello D'Andrea, and Markus Waibel. Rapyuta: A cloud robotics platform. *IEEE Transactions on Automation Science and Engineering*, Vol. 12, No. 2, pp. 481–493, 2015.

[13] Meixia Fu, Songlin Sun, Kaili Ni, and Xiaoying Hou. Mobile robot object recognition in the internet of things based on fog computing. In *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, 2019.

[14] Yuya Maruyama, Shinpei Kato, and Takuya Azumi. Exploring the performance of ROS2. In *2016 International Conference on Embedded Software (EMSOFT)*, pp. 1–10, 2016.