交差点における路側エッジへの自動運転機能のオフロード

平田 真唯[1]　佐藤 友哉[1]　塚田 学[1]　落合 秀也[1]　江崎 浩[1]

概要：近年，自動運転技術が注目されている．自車に搭載されているセンサを用いた，衝突被害軽減ブレーキやレーンキープアシスト，自動駐車などの機能はすでに商業化されている．ところが，このような自律走行では，死角に存在する車両，人等の，自社のセンサで感知しきれない情報に関して安全を保証できないという問題がある．また，自動運転車が生み出すデータの量や，車両で使用されるアプリケーションが増加している．ところが，車両に搭載することができる端末には，ストレージやコンピューティング能力の制限がある．本研究では，これらの問題を解決する一歩として，自動運転の機能のうち，経路計画の機能を路側エッジにオフロードする分散コンピューティングモデルを提案した．提案手法は，Wi-Fi6 を用いて評価した．その結果，経路計画を CPU や GPU の性能が高い路側エッジで行うことにより，少ないネットワークトラフィック量で，車で処理するよりも素早く経路を計算できることがわかった．

Offload of Autonomous Driving Function to Roadside Edge at Intersection

MAI HIRATA[1]　TOMOYA SATO[1]　MANABU TSUKADA[1]　HIDEYA OCHIAI[1]　HIROSHI ESAKI[1]

1. はじめに

ITS（高度道路交通システム: Intelligent Transport System）とは，情報通信技術を用いて，人と道路と車両との間で情報の送受信を行い，交通事故，渋滞などの道路交通問題を解決する目的とした交通システムである [1]．既存の ITS サービスには，ETC（自動料金支払いシステム: Electronic Toll Collection System）や VICS（道路交通情報通信システム: Vehicle Information and Communication System）などがある．近年，ITS の一つである自動運転技術が注目されている．日本でも 2020 年にはレベル 3 の条件付き自動運転の市場化が可能になるよう，必要な技術の開発を行っている [2]．

従来の運転手を支援するシステムでは，自車に搭載されているセンサを用いている．それらのセンサを利用した，衝突被害軽減ブレーキやレーンキープアシスト，自動駐車などの機能はすでに商業化されており，国土交通省によると，実際に 2016 年現在，新車の 66.2%に衝突被害軽減ブレーキは搭載されている．これらの機能は，SAE International の定義によると，レベル 1，2 に当てはまる．ところが，自律走行では死角に存在する車両，人等の自車のセンサで感知しきれない情報に関して，安全を保証できないという問題がある．

そこで，協調型 ITS が注目されている．協調型 ITS は，周囲の車両や，道路に設置された対応機器と自動車が通信を行う．このシステムでは，自車のセンサだけでは得られない情報を得ることができる．トヨタの ITS Connect [3] では，日本全国の人や車の往来が激しい交差点の一部に ITS 専用無線装置を設置している．ITS 専用無線装置が設置された交差点では，右折時注意喚起や赤信号注意喚起などを行うことができる．

また，近年では自動運転に分散コンピューティングを取り入れる動きがある．現在，車には平均 60〜100 個のセンサが搭載されている．将来，センサの数は 200 個に達すると言われている．Intel [4] によると，1 台の車は，1 日に 4000GB のデータを生成するようになる．これらのデータは、通信・ストレージ・コンピューティングリソースに大きな負荷を与える．自動車に搭載されている端末には，ストレージやコンピューティング能力の制限がある．そのため，分散コンピューティングを利用する必要がある．

そこで，本研究では，車の処理を減らすため，自動運転

[1] 東京大学大学院情報理工学系研究科

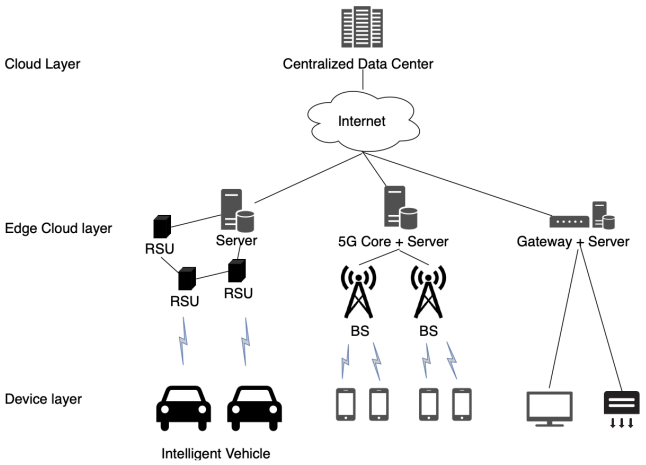


図 1　分散コンピューティングの 3 層構造.

機能のうち，経路計画の機能を路側エッジにオフロードするモデルを提案した．提案手法は東京大学本郷キャンパスの走行データを用いて評価した．実験は，Wi-Fi6 を用いて行った．

本研究の貢献は以下のとおりである．

(1) 自動運転システム用オープンソースソフトウェアの Autoware [5] を用いて路側エッジへの自動運転機能のオフロードの性能を測定した．また，Wi-Fi6 を用いて、通信の評価も行った．

(2) 共有資源である無線での通信量を減らす、新たな分散コンピューティングモデルを提案した．

本論文の構成は次のようになっている．まず，2 章で関連技術と関連研究を紹介し，3 章で提案手法について説明する．4 章で実験の概要と実験の結果について説明する．5 章で本稿のまとめと課題について述べる．

2. 関連技術と関連研究

2.1 協調型 ITS

ADAS（Advanced driver-assistance systems）などの運転手を支援するシステムは，安全なスピードや距離を維持したり，レーンキープアシストを行ったりすることで，安全な交通システムに貢献してきた．一方，個々の車が，他の車や道のインフラストラクチャと通信を行うことで，さらに交通を安全に行うことができる．

そこで近年では，車を他の車やインフラストラクチャと通信させる協調型 ITS が注目されている．

2.2 エッジコンピューティング

自動運転車に乗っている端末には，ストレージやコンピューティングの制限がある．そのため，分散コンピューティングを利用して，自動運転車の端末の負荷を軽減する試みがある．分散コンピューティングには，主にクラウドコンピューティングとエッジコンピューティングがある．その構成図を図 1 に示す．

エッジコンピューティングは，ネットワークの端（edge）でデータ処理を行うモデルであり [6]，図 1 の一番下の層にあたる．ここでの edge とは，データの発生元からクラウドのデータセンターまでの道沿いにあるコンピューティング，ネットワークリソースと定義する．例えば，スマートフォン，スマートホームでのゲートウェイ，ミクロデータセンター，cloudlet [7] などである．ネットワークの端，つまりクラウドよりもユーザに近いところで処理を行うことで，レスポンスタイムをはやくしたり，エネルギーの消費量を減らすことができる．

エッジコンピューティングを実現する技術として，広く用いられているものとして，ETSI によって標準化が進められている MEC（Multi-access Edge Computing） [8] などがある．

自動運転におけるエッジコンピューティングでは，BS や RSU（路側機: Road Side Unit）に置かれることを想定されている．本研究では，RSU に置かれたエッジコンピューティングリソースのことを路側エッジとよぶ．SSD のようなストレージユニットや，GPU のようなコンピューティングユニットから構成されることを想定する．

2.3 自動運転における無線通信技術

これらの協調型 ITS や分散コンピューティングを実現するためには, 無線通信技術が必要である．例えば，車車間通信を実現する無線通信技術として，IEEE802.11 シリーズの 11b，11g，11a などの既存の無線通信規格がある．また，車車間通信に特化した通信技術として，IEEE 802.11p に基づく，DSRC（Dedicated Short-Range Communications）や，ESTI ITS-5G，セルラー通信に基づく LTE-V がある．DSRC は日本では 5.8 GHz 帯，欧州では 5.9 GHz 帯，アメリカでは 5.9 GHz 帯が割り当てられている，DSRC を利用して，日本では ETC2.0 渋滞情報サービスが提供されている．日本では，5.8 GHz 帯の他にも，車車間の通信に 760 MHz 帯が割り当てられている．ITS Connect では，この周波数帯を用いている．また，免許不要の周波数帯を利用している Bluetooth，Zigbee なども通信方式として利用することができる．さらに，高速かつ低遅延を可能にする最新の無線技術として，Wi-Fi6 [9] がある．Wi-Fi6 は IEEE 802.11ax ともよばれ，IEEE が標準化を推進している．2020 年に標準化予定である．周波数帯は，2.4 GHz 帯と 5 GHz 帯を利用している．最大通信速度は理論値で 9.6 Gbps である．また，第 5 世代の移動通信テクノロジーとなる 5G は，高速・大容量，低遅延，他接続が特徴である．最大通信速度は，理論値で 20 Gbps といわれている．

2.4 ROS（Robot Operating System）

ROS [10] は，オープンソースのミドルウェアフレームワークで，Linux OS 上で動作する．ROS の実装の基本的

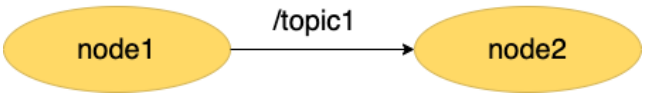


図 2　ROS でのノードとトピックの関係.

な概念として，ノード，メッセージ，トピック，サービスがある．ノードは，計算を行うプロセスである．システムは通常，多くのノードで構成されている．

ノード同士はメッセージをやり取りすることで通信している．ノードは，単純な文字列である特定のトピックにメッセージを発行することで，メッセージを送信し，特定のデータに関心があるノードは，適切なトピックに対して配信を申し込むというパブリッシュ/サブスクライブ・モデルである．ノードとトピックの関係は図 2 のように表される．この図は，ノード 1 がトピック 1 にメッセージをパブリッシュし，ノード 2 がトピック 1 をサブスクライブすることで，ノード 1 がパブリッシュしたメッセージを受け取ることを示す．

このような特徴から，ROS は複数 PC 間にまたがって実行することができる．また，ROS は ROSBAG ファイルとしてトピックデータを保存することができる．

2.5　Autoware

Autoware [5] は，自動運転システム用オープンソースソフトウェアである．国内外 200 社に導入されている．Autoware は，Linux と ROS をベースに開発されており，自己位置推定や，物体認識，経路計画，車両制御などの機能を備えている．実証実験も積極的に行われている．

2.6　Sasaki らの研究

Sasaki ら [11] では，Autoware [5] を用いて，自己位置推定と経路計画の機能をエッジに移行することにより，分散コンピューティングを行う手法を提案した．この手法では，図 3 のように，自己位置推定をエッジで行っているため，LiDAR で取得している点群データをリアルタイムにエッジに送っている．また，経路計画のためのコストマップや目的地もエッジに送っている．エッジからは，自己位置推定により計算された自己位置と経路計画により算出された経路を車が受け取っている．Sasaki らは，この手法を 1 Gbps のイーサネットケーブルを用いて評価した．

2.7　近年の協調型 ITS における認識情報の共有に関する研究

本節では，近年の協調型 ITS における認識情報の共有に関する研究について簡単に紹介する．

まず，自動運転車がどのように車両の周囲の状況を認識しているか説明する．自動運転車には，様々なセンシングデバイスが搭載されている．主要なものを下にあげる．

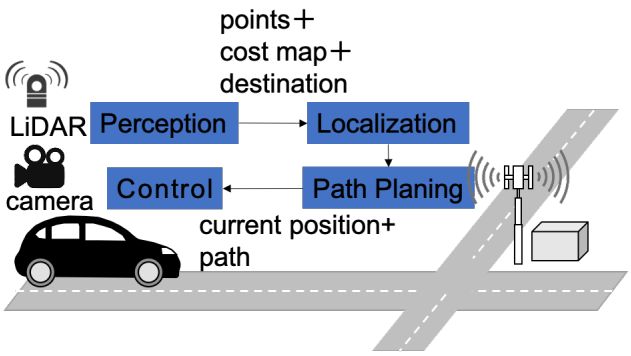


図 3　Sasaki ら [11] の提案した分散コンピューティングモデル.

カメラ　CNN と組み合わせて，周辺の物体の検知や分類，白線の認識に用いられる．また，自己位置推定に用いられることもある．

LiDAR（Light Detection and Ranging）　レーザー光を走査しながら対象物に照射してその散乱や反射光を観測することで，対象物までの距離を計測したり対象物の性質を特定したりする，光センサー技術のことである。LiDAR は，周辺の物体の検出，自己位置推定に用いられる．

ミリ波レーダ　物体の検知に用いられる．

自動運転では，これらのセンサの情報を組み合わせることで，周囲の状況を認識している．

これらのセンサを用いて得た認識情報を融合することで，物体認識を行う．センサデータの融合技術は，大きく分けて 3 つのカテゴリーに分けることができる．

- **low-level**
  前処理のされていない生のデータを融合する．
- **feature-level**
  データの融合を行う前に，前処理として生のデータから特定の特徴を抽出する．
- **track-level**
  それぞれのセンサが独立にトラッキングアルゴリズムを実行し，障害物リストを生成する．

近年の認識情報の共有に関する研究を例に上げると，TruPercept [12] と Shi ら [13] の研究，Gabb ら [14] の研究，Kitazato ら [15] の研究，Tsukada ら [16] の研究では，自分の車に搭載されているセンサと自車以外の周囲の車のセンサデータを track レベルで融合している．一方，F-Cooper [17] では，自分の車に搭載されているセンサと自車以外の周囲の車のセンサデータを feature レベルで融合している．low レベルの融合を行う融合は，ネットワーク負荷が大きいため，近年の研究ではあまり扱われていない．

3.　提案手法

3.1　要件定義

自動運転機能を分散コンピューティングするにあたって

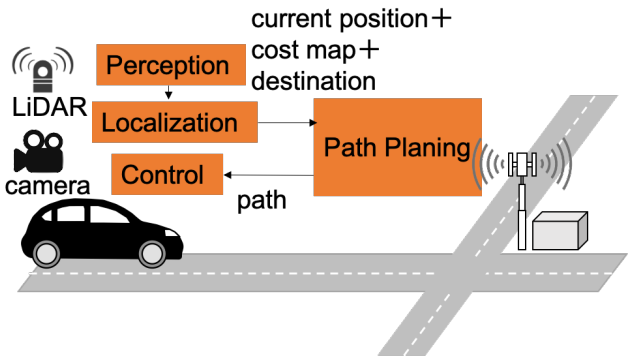


図 4　提案する分散コンピューティングモデル.

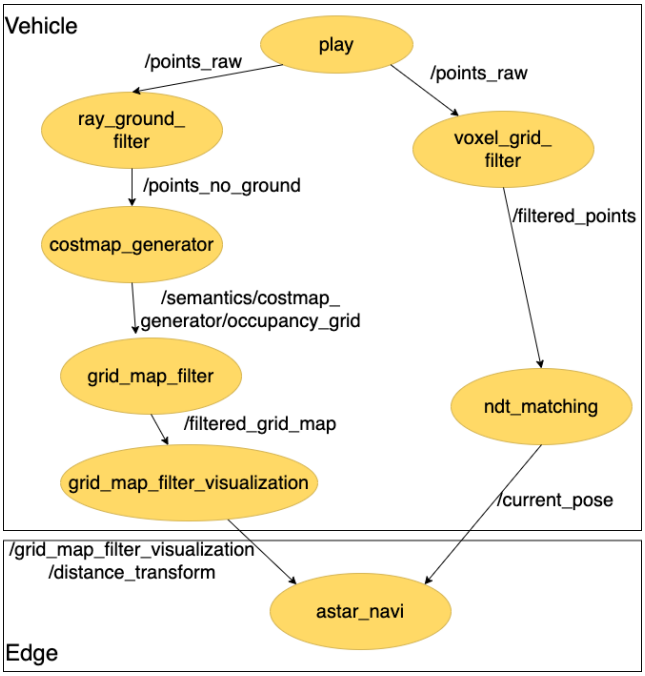


図 5　提案手法の ROS ノードの関係.

の要件を定義する.

1 点目として，実際の無線通信では，ネットワークの負荷を考慮すると，複数の車が大量の点群データを路側エッジに送ることは好ましくない．そのため，大量の点群データを必要とする自己位置推定の機能は路側エッジ側で行わないほうがよいと考える．2.7 章で述べたように近年の協調型 ITS における認識情報の共有に関する研究によると，点群の生のデータをそのまま送ることは望ましくなく，障害物リストとして共有することが望ましい．これらの理由からも点群データを共有することは望ましくないとされる.

2 点目として，実行時間の制約である．今回使用した Autoware [5] では，/astar_navi は 0.5 Hz，それ以外のノードでは，10 Hz で実行される．これらの周期から，経路計画の/astar_navi は 2000 ms 以内，自己位置推定は 100 ms 以内で処理を終える必要がある.

本研究では，以上の 2 つの要件をみたすような分散コンピューティングのモデルを提案した.

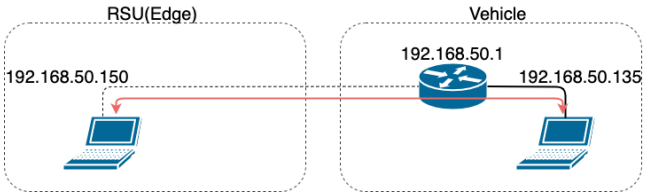


図 6　実験構成.

表 1　実験で使用したコンピュータの性能.

	Vehicle	Roadside Edge
CPU	Intel Core i7-6700	Intel Core i7-9750
Architecture	Intel 4 Core	Intel 6 Core (12 Thread)
CPU Frequency	2.6 GHz	2.6 GHz
Memory	32GB DDR4	8GB
GPU	GTX 1060	RTX 2070
Total CUDA Core	1280	2304
GPU Memory	6GB	8GB GDDR
Linux Kernel	4.4.0-164-generic	5.3.0-42-generic
Ubuntu Version	16.04	18.04
ROS version	kinetic	Melodic
Autoware version	1.12.0	1.12.0

3.2　提案するモデル

本研究では，図 4 のようなモデルを提案する．自動運転には，自己位置推定のための点群データが必要である．また，機能として，自己位置推定や，物体認識，経路計画，車両制御がある．本研究では，この中で経路計画の機能のみを路側エッジにオフロードするモデルを提案する．自己位置推定を路側エッジで行うと，通信の開始時に，自己位置推定のための点群データを送る必要があり，また LiDAR でスキャンされるリアルタイムの点群データを送り続ける必要がある．そのため，自己位置推定の機能は車に残したほうがよいと考えた．一方，物体認識も同様にネットワークの負荷が大きくなるうえ，物体認識と車両制御は自動運転の安全性に重要なため，車に残すべきである.

4.　評価

4.1　実験に使用したデータ

実験に使用したデータは，東京大学本郷キャンパスを実際に走行して得た ROSBAG データである．車はコムスを使用し，LiDAR は VELODYNE VLP-16 を使用している.

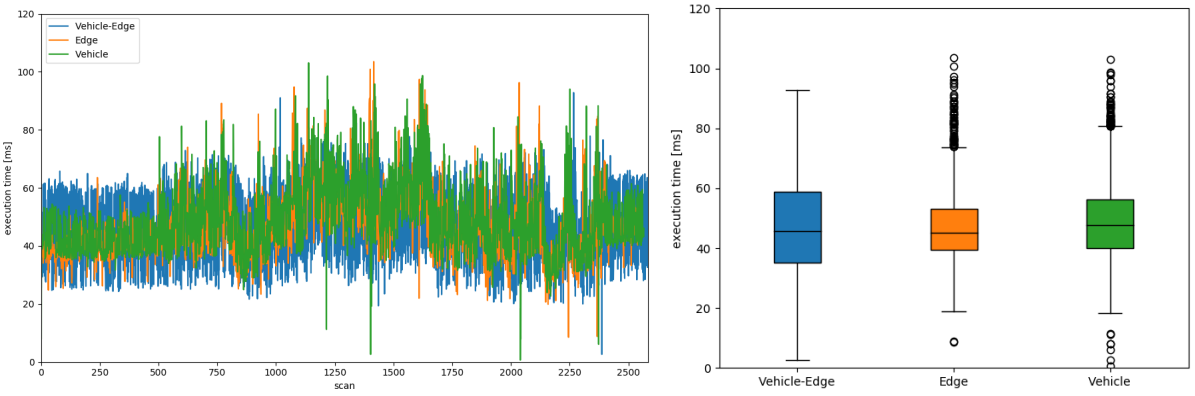
4.2　提案するモデル

このモデルは，図 5 のような ROS のノードとトピックからなる．それぞれの ROS ノードについて簡単に説明する.

- **play**

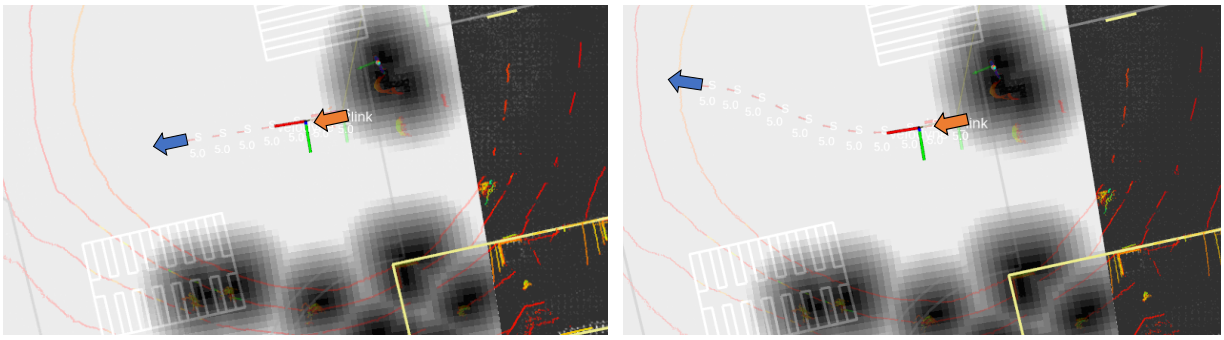
  ROSBAG を再生し，記録時のセンサ情報を再現する．ここでは，LiDAR のデータを再現し，/points_raw にパブリッシュする.

- **voxel_grid_filter**



(a) スキャンごとの実行時間の結果　　(b) (a) を箱ひげ図で表したもの

図 7　Wi-Fi6 を使用したときの ndt matching の実行時間の評価.



(a) ケース 1 の経路　　(b) ケース 2 の経路

図 8　astar の実験で用いたスタート地点 (橙色の矢印) とゴール地点 (青色の矢印).

スキャンデータを間引いて，このあとの ndt_matching で使われる点群を削減する．具体的には，/points_row から得たスキャンデータを間引いて，/filtered_points にパブリッシュする．

- **ndt_matching**
  ndt matching を実行する．ndt matching は自己位置推定の手法であり，3 次元地図と LiDAR から得たスキャンデータに対し，スキャンマッチングを行うことにより自己位置を推定する．
- **ray_ground_filter**
  スキャンデータから地面の部分を取り除く．
- **costmap_generator**
  認識した結果を用いて，計画に用いるためのコストマップを生成する．ここでは，ray_grand_filter から得たポイントクラウドを用いて，コストマップを生成している．
- **grid_map_filter**
  センサデータ, 地図, 認識したデータをくっつけ, ナビゲーションに使用するデータを加えた OccupancyGrid を生成する．
- **astar_navi**
  A-star アルゴリズムを実行し，現在の場所から特定の場所までのうまく経路を算出する．

4.3　実験の構成

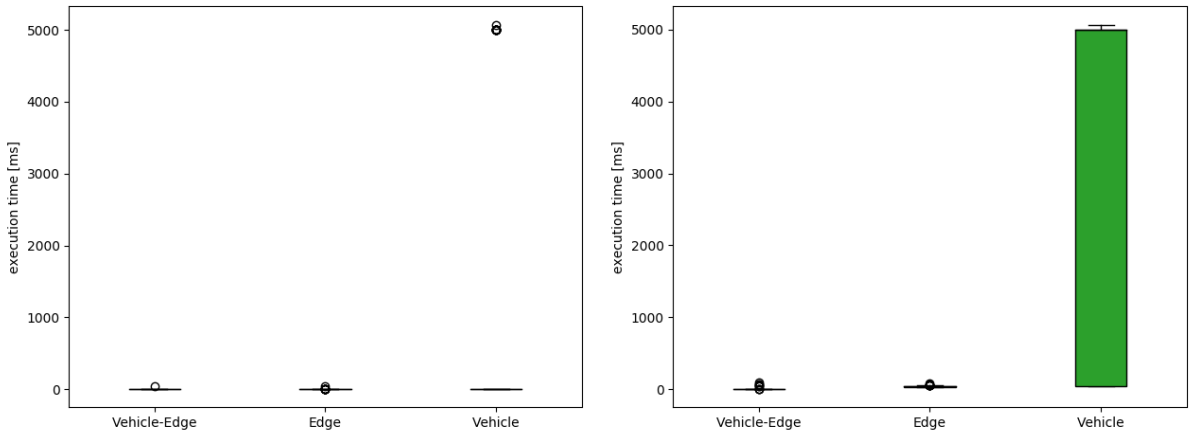
図 6 のような構成で実験を行った．今回使用したデバイスの性能は表 1 に示した．表 1 に示したように，メモリは路側エッジ側が 32 GB，車側が 8 GB，GPU メモリは路側エッジ側が 8 GB，車側が 6 GB 等と，路側エッジ側のデバイスのほうが CPU や GPU の性能がよいパソコンを用いた．ルータは，ASUS RT-AX88U を使用した．このルータは，802.11ax を用いることで，5 GHz 帯で最大 4804 Mbps の高速通信を実現している．

実際に図 6 の構成で通信帯域の測定を行った．50 cm 離したところで測定すると，900 Mbps 以上の帯域幅を確認できた．今回の実験では，車側のルーターとパソコンは 1 Gbps イーサネットを用いて通信している．

このような Wi-Fi6 の環境下で実験を行った．

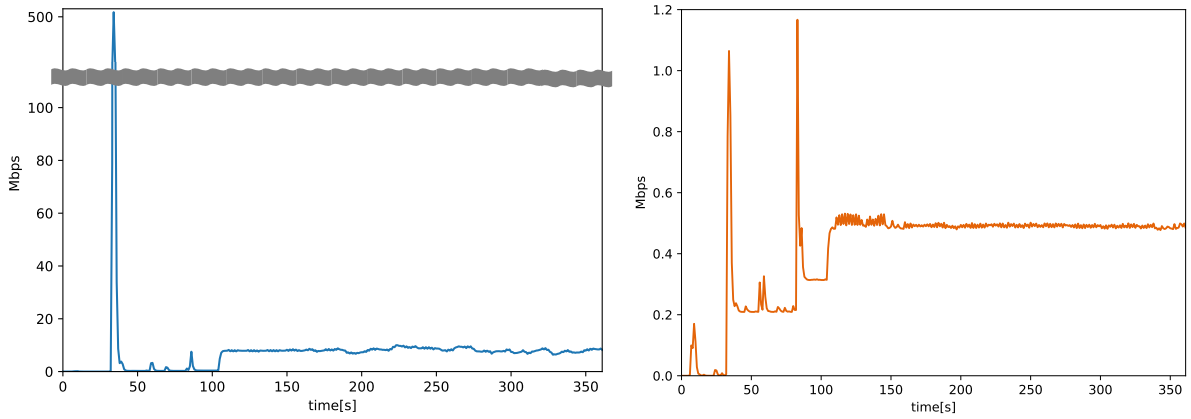
4.4　自己位置推定の実行時間

Sasaki ら [11] が提案したモデルをもとに，Wi-Fi6 を用いて自己位置推定，つまり ndt matching の実行時間を評価した．その結果を図 7 に示す．この ndt matching の実行時間は，Sasaki ら [11] の研究と同様に，play ノー



(a) ケース 1　　(b) ケース 2

図 9　astar の実行時間.



(a) 送信側　　(b) 受信側

図 10　車側から測定した Sasaki らの提案したモデルを Wi-Fi6 で実行中のトラフィック量の変化.

ドが/points_raw をパブリッシュしてから，ndt_matching ノードが/current_pose をパブリッシュするまでの時間を測定した．図 7 では，Vehicle-Edge が Sasaki らのモデルに従って処理を分散させた場合，Edge は路側エッジで全ての処理を行った場合，Vehicle は車で全ての処理を行った場合の結果を示している．

先行研究では，有線を用いて実験を行っていたが，Wi-Fi6 を用いても，処理のオフロードが遅延なく行えていることがわかった．

4.5　経路計画の実行時間

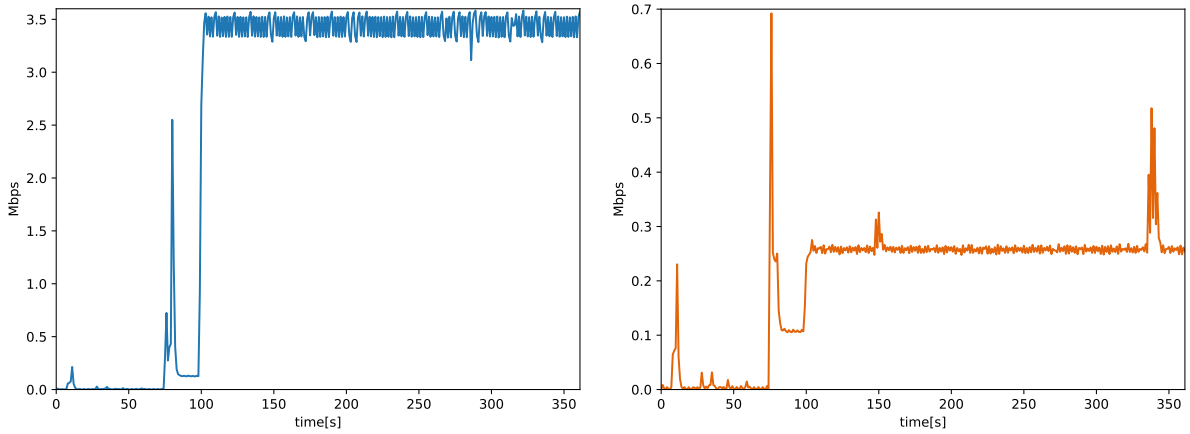
経路計画，つまり A-star アルゴリズムの実行時間の評価を行った．A-star アルゴリズムの実行時間は経路のスタート地点とゴール地点に左右される．今回は図 8 に示した 2 つの経路で実験を行った．図 8 の橙色の矢印がスタート地点，青色の矢印がゴール地点を示す．2 つの矢印間の赤い矢印が，A-star アルゴリズムで計算された経路である．

その結果を図 9 に示した．図 9 では，Vehicle-Edge が提案したモデルに従って処理を分散させた場合，Edge は路側エッジで全ての処理を行った場合，Vehicle は車で全ての処理を行った場合の結果を示している．ケース 1，ケース 2 双方で，車だけで処理を行ったときのほうが，処理時間が長くなるときがあった．経路計画のアルゴリズムのデッドラインは 2000 ms なので，車だけで処理を行った場合は，このデッドラインをこすことがあるとわかった．これは，A-star アルゴリズムは，CPU の性能に依存するため，車で処理するよりも, より CPU 性能の高い路側エッジで処理したほうが処理時間が短くなったといえる．

4.6　実験中のネットワークのトラフィック量の変化

実験を行ったときのネットワークのトラフィック量の変化を測定した．Sasaki ら [11] のモデルで実験を行ったときのトラフィック量の変化を図 10 に，提案したモデルで実験を行ったときのトラフィック量の変化を図 11 に示す．これらのトラフィック量は車側から測定したものを示した．

Sasaki ら [11] のモデルで実験を行った場合は，ndt



(a) 送信側　　(b) 受信側

図 11　車側から測定した提案したモデルを Wi-Fi6 で実行中のトラフィック量の変化.

matching を実行したときに通信量が 537 Mbps と最大になった．これは，ndt matching の開始の際に，本郷キャンパスのポイントクラウドマップを送るためだと考える．110 秒後からの通信帯域の平均は 8.09 Mbps と最大になった．

一方，A-star アルゴリズムのみ路側エッジ側で処理を行った場合は，図 11(a) のように，ネットワークの負荷が最大のときで 3.58 Mbps となった．トラフィック量は Sasaki らのモデルと比べて半分以下になった．

5. おわりに

5.1 まとめ

本研究では，路側エッジに自動運転機能をオフロードするための，新しい分散コンピューティングモデルを提案した．また，先行研究のモデルと提案したモデルを Wi-Fi6 を用いて評価した．

自己位置推定を路側エッジで行う場合は経路計画だけを行うよりもネットワーク負荷が大きかったが，無線の状況次第でオフロードが可能であると考えた．また，経路計画のための A-star アルゴリズムは CPU の性能の影響を大きく受けるため，路側エッジの強力なコンピューターで計算するのがよいといえる．

5.2 今後の課題

今後の方針として, 今回の実験では，路側エッジと車の距離を 50 cm でおこなった．無線通信では，距離が離れると，通信がとぎれる可能性がある．どのくらいの距離まで要件をみたしたエッジコンピューティングが行えるのか調査する必要があると考える．

また，交差点に設置した RSU にセンサを設置し，このセンサから得たデータも加え，交差点での経路計画を行いたいと考える．

さらに，複数のエージェントの経路計画に関する multi agent path finding という研究テーマをエッジに取り入れてみようと考えている．

参考文献

[1] 高度情報通信ネットワーク社会推進戦略本部・官民データ活用推進戦略会議. 官民 its 構想・ロードマップ 2019. https://www.kantei.go.jp/jp/singi/it2/kettei/pdf/20190607/siryou9.pdf, 2019. (参照 2020-05-10).

[2] 内閣府政策統括官（科学技術・イノベーション担当）. 戦略的イノベーション創造プログラム（sip）自動走行システム研究開発計画. https://www8.cao.go.jp/cstp/gaiyo/sip/keikaku/6_jidousoukou.pdf, 2018. (参照 2020-05-10).

[3] トヨタ. トヨタ トヨタの最新技術 — 安全技術 — its connect — トヨタ自動車 web サイト. https://toyota.jp/technology/safety/itsconnect/. (参照 2019-10-22).

[4] Brian Krzanich. Data is the new oil in the future of automated driving - intel newsroom. https://newsroom.intel.com/editorials/krzanich-the-future-of-automated-driving/, 2016. (参照 2020-05-10).

[5] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, and T. Hamada. An open approach to autonomous vehicles. *IEEE Micro*, Vol. 35, No. 6, pp. 60–68, Nov 2015.

[6] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, Vol. 3, No. 5, pp. 637–646, Oct 2016.

[7] Mahadev Satyanarayanan, Victor Bahl, Ramon Caceres, and Nigel Davies. The case for vm-based cloudlets in mobile computing. *IEEE Pervasive Computing*, November 2009.

[8] ETSI. Multi-access edge computing - standards for mec. https://www.etsi.org/technologies/multi-access-edge-computing. (参照 2020-05-10).

[9] B. Bellalta. Ieee 802.11ax: High-efficiency wlans. *IEEE Wireless Communications*, Vol. 23, No. 1, pp. 38–46, 2016.

[10] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In

ICRA workshop on open source software, Vol. 3, p. 5. Kobe, Japan, 2009.

[11] Y. Sasaki, T. Sato, H. Chishiro, T. Ishigooka, S. Otsuka, and K. Yoshimuraand S. Kato. An edge-cloud computing model for autonomous vehicles. In *11th IROS Workshop on Planning, Perception, Navigation for Intelligent Vehicle*, 2019.

[12] Braden Hurl, Robin Cohen, Krzysztof Czarnecki, and Steven Waslander. Trupercept: Trust modelling for autonomous vehicle cooperative perception from synthetic data, 2019.

[13] J. Shi, W. Wang, X. Wang, H. Sun, X. Lan, J. Xin, and N. Zheng. Leveraging spatio-temporal evidence and independent vision channel to improve multi-sensor fusion for vehicle environmental perception. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 591–596, June 2018.

[14] M. Gabb, H. Digel, T. Müller, and R. Henn. Infrastructure-supported perception and track-level fusion using edge computing. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1739–1745, June 2019.

[15] T. Kitazato, M. Tsukada, H. Ochiai, and H. Esaki. Proxy cooperative awareness message: an infrastructure-assisted v2v messaging. In *2016 Ninth International Conference on Mobile Computing and Ubiquitous Networking (ICMU)*, pp. 1–6, 2016.

[16] Manabu Tsukada, Masahiro Kitazawa, Takaharu Oi, Hideya Ochiai, and Hiroshi Esaki. Cooperative awareness using roadside unit networks in mixed traffic. In *IEEE Vehicular Networking Conference (VNC)*, University of California - Los Angeles (UCLA), United States, December 2019.

[17] Qi Chen, Xu Ma, Sihai Tang, Jingda Guo, Qing Yang, and Song Fu. F-cooper: feature based cooperative perception for autonomous vehicle edge computing system using 3d point clouds. In *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*, pp. 88–100, 2019.