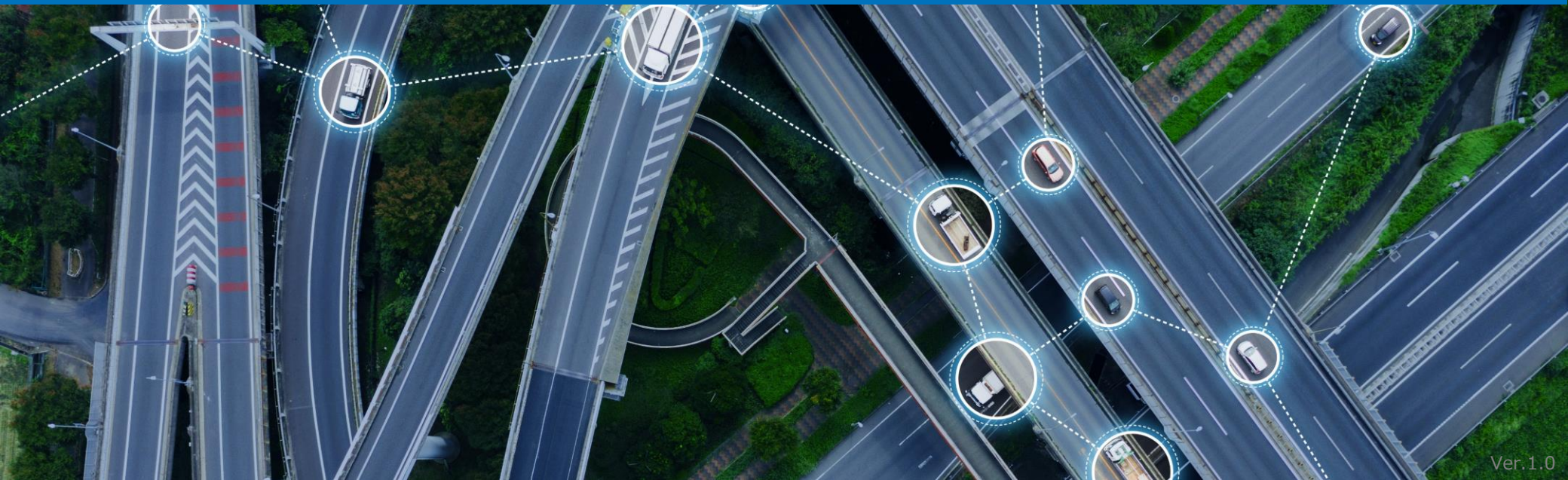
TOYOTA　NTTグループ

コネクティッドカー向けICT基盤に関する技術資料

2021.11.29



Ver.1.0

はじめに

TOYOTA NTTグループ

自動車業界は今まさに100年に一度の変革期に直面している。この数年間で多くのコネクティッドカーが市場に投入されたことによって、運転者・同乗者やメーカー、保険会社、整備工場などに対して様々なサービスが実現しつつある。今後、さらなる普及と市場拡大が予想される。

コネクティッドカーの普及により、センサーが検知した膨大なデータを活用することが可能となる。
ICT基盤技術を用いたデータ分析結果の活用は、安全な移動(事故・渋滞ゼロ)や災害被害軽減等の社会課題解決、さらには新たな価値を提供するモビリティサービスへの貢献が期待されている。

本技術資料は、下記基盤システムに関する技術成果を取りまとめたものである。

- コネクティッドカーが生み出す、大量データを高速・効率的に収集
- コネクティッドカー向けICT基盤として技術難易度の高いユースケース検証
  （例えば、実世界をサイバー空間上に再現）
- 実フィールドにて実車を用いたEnd to Endの実証実験を支えるシステム

本技術資料が、コネクティッドカー、ICT、モビリティ関係者だけでなく、他業界に広く参考になれば、幸いである。

Index

TOYOTA NTTグループ

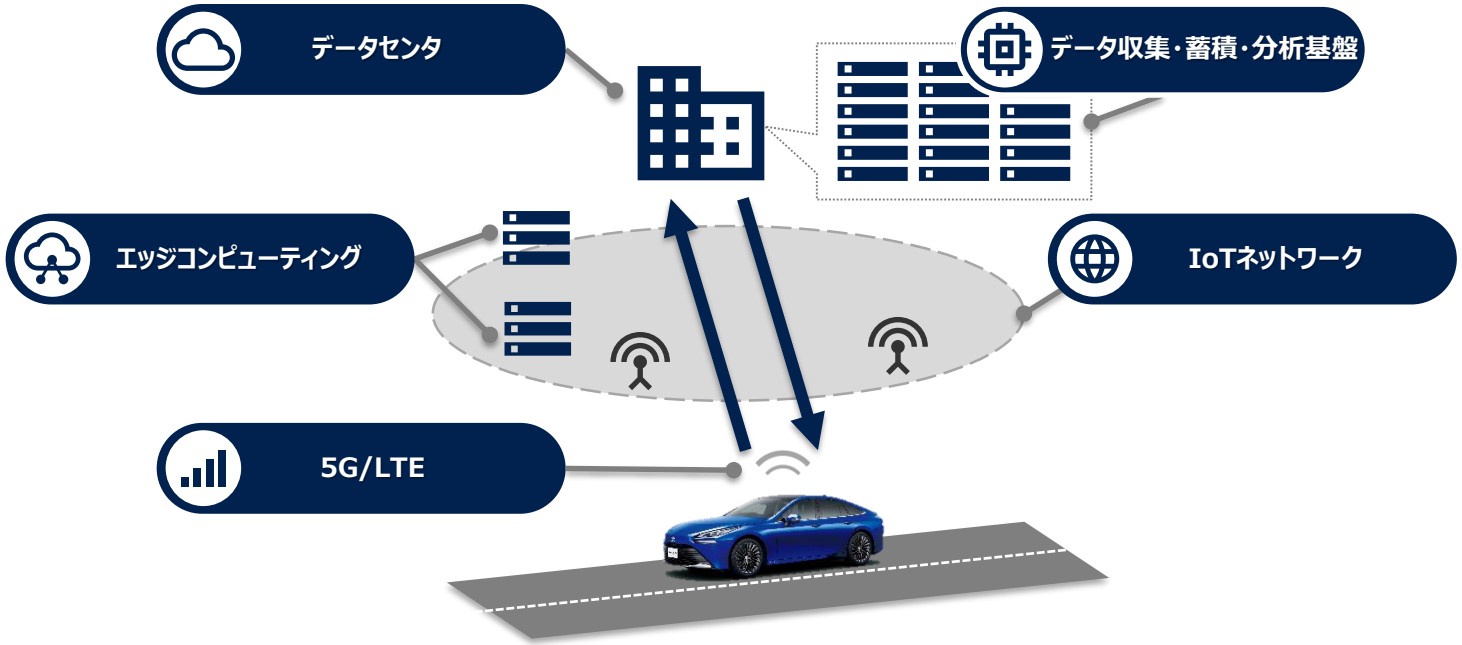
- コネクティッドカー向けICT基盤とは
- 実証実験の取り組み
  - 全体概要・アーキテクチャ
  - システム構成・ハード構成・ソフトウェア構成
  - ユースケース1: 静的地図生成
  - ユースケース2: 障害物検知
  - ユースケース3: 渋滞検知
- 今後の取り組み

コネクティッドカー向けICT基盤とは

コネクティッドカー向けICT基盤の全体像

TOYOTA NTTグループ

コネクティッドカー向けICT基盤は、通信機能を装備したクルマとエッジコンピューティングおよびクラウド間をLTE・5GとIoTネットワークで接続し、クルマのデータをコンピューティングリソースを用いて、データ収集・蓄積・分析する基盤である。



コネクティッドカー向けICT基盤の課題

TOYOTA NTTグループ

コネクティッドカー向けICT基盤の主な課題は、「接続車両数」「リアルタイム性」「精度」がある。大量のクルマのリアルタイム且つ高精度にてデータ処理することで、クルマのデータは、広範な利用価値が生まれる。

課題①	**接続車両数（量）**	数千万台を超える大量の車両から、車両/画像データを収集し活用するためには、その基盤に超大量、超高トランザクションが求められる。
課題②	**リアルタイム性**	高速に移動する車両の位置情報などをリアルタイムに収集・蓄積し、各社サービスにリアルタイムの情報を提供することが求められる。
課題③	**精度**	車両から得られるデータに基づき、車両や標識、信号機などの地物の位置を高精度(GPS以上の精度)で推定し、センター側で管理することが求められる。

実証実験の取り組み

実証実験の目的

実証実験の検証では、コネクティッドカー・自動運転車が本格普及した際も対応できるクラウド基盤技術／通信インフラ技術の見極め、および次世代の車両側への要件を整理することを目的とする。

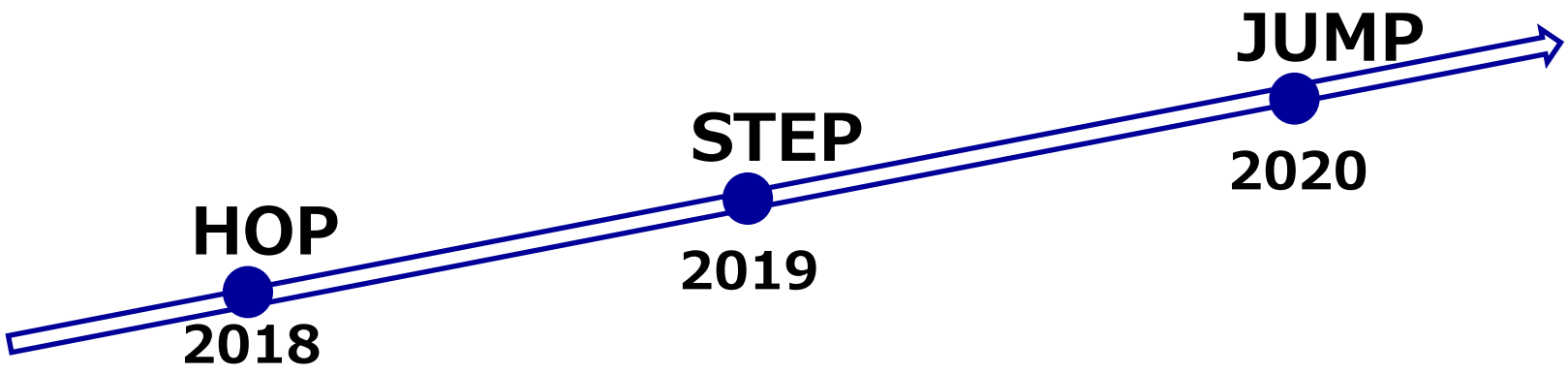
実証実験の目的

1	**大規模基盤における クラウド・通信インフラ技術の 見極めと技術課題の抽出**	2023年を大規模基盤実現のターゲットと仮置きし、今日までの協業活動で培ったノウハウを集大成した設計・検証・評価を行い、コネクティッドカー・自動運転基盤を支えるクラウド技術、および次世代通信インフラに求められる技術課題を明確にする。
2	**次世代車両開発への 要件の抽出**	2023年のコネクティッドカーの本格普及、および将来に控える自動運転車の普及に向けては、早めにクラウド・通信インフラ側からの制約を明らかにし、足の長い車両開発への要件を明確にする。

実証実験のロードマップ

本実証実験は2020年度までの中長期に向けた段階的な目標を定める。

中長期的な実証実験のロードマップ



	HOP 2018	STEP 2019	JUMP 2020
性能	接続車両　数十万台 処理時間　10分程度 処理データ　数十項目	接続車両　数百万台 処理時間　数十秒程度 処理データ　数百項目	接続車両　数千万台 処理時間　秒単位 処理データ　数千項目

実証実験の全体概要

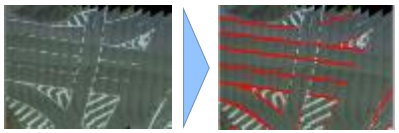
実証実験では、車両や外部からのデータを収集し、クラウド内でデータ処理・統合することで、**クラウド上へ実世界を再現すること**をめざす。また、**コネクティッドカーが本格普及した規模でも再現ができることも併せて検証する**。将来の自動運転やモビリティサービス実現に必要なシーンの再現に、優先的に取組む。車両データについては、お台場にて実車を走らせ収集する。

データ収集（車両から）

車両内部状態データ
CAN・ECUデータ

周辺監視データ

データ処理
データ統合



例）白線検知

データ収集（外部から）

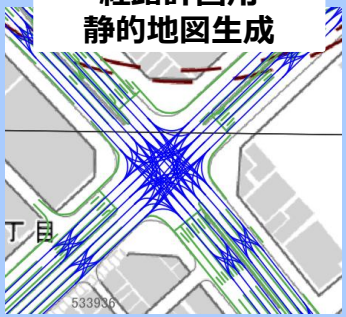
詳細な道路情報
- ・高精度地図
- ・交通事故多発地点

道路交通状況
- ・渋滞情報
- ・道路工事、事故
- ・天気・路面情報

データ処理
データ統合

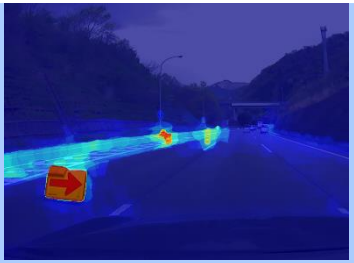
現況把握・予測（例）

経路計画用
静的地図生成



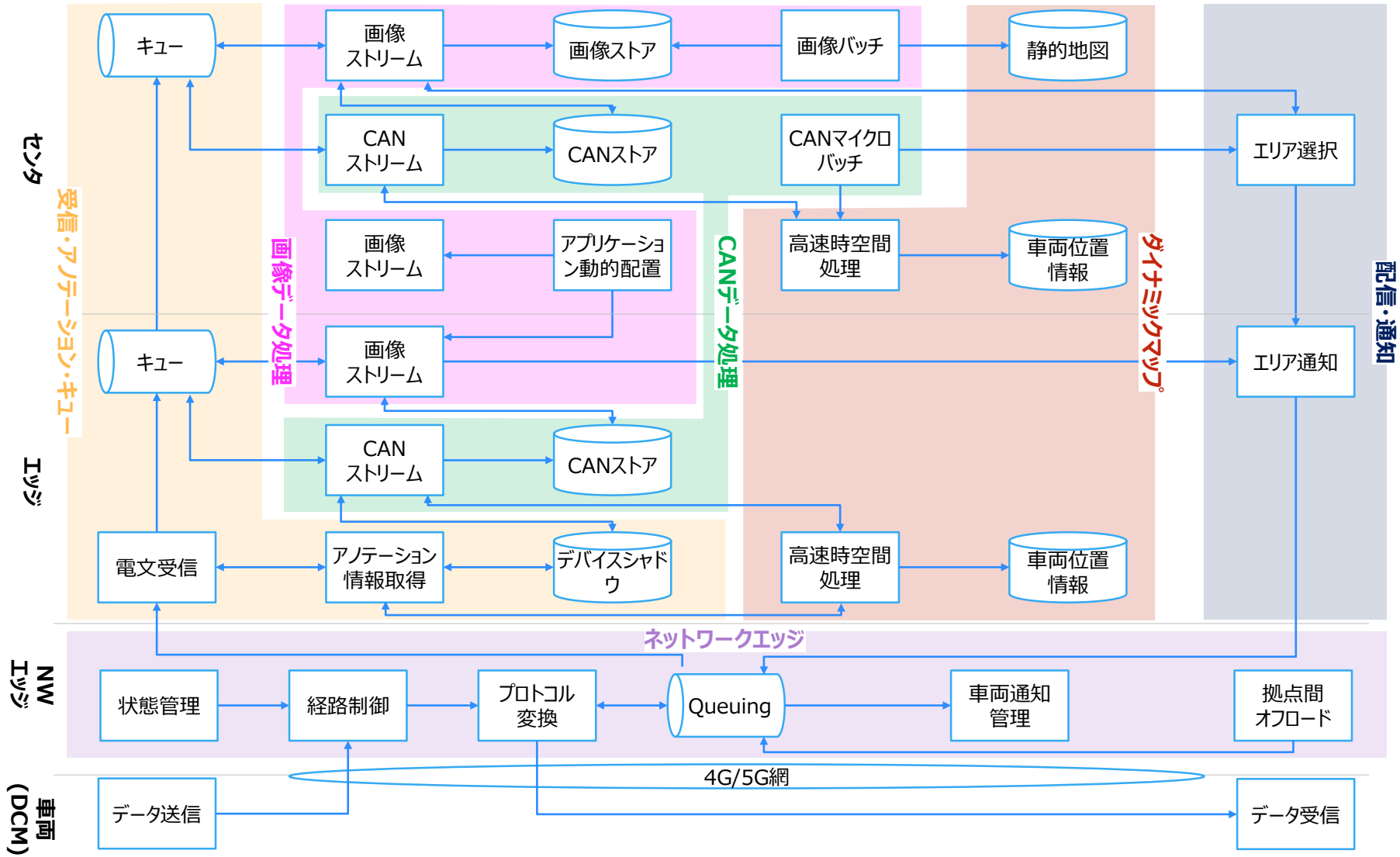
レーンレベル渋滞
検知＆渋滞表示

落下物/障害物検知＆後続車への通知



実証実験のアーキテクチャ

本実証実験は以下のアーキテクチャで実施する。



ユースケース検証

検証概要

ユースケース検証は、コネクティッドカー向けICT基盤の課題である接続車両数、リアルタイム性、精度が求められる3つのユースケースについて検証する。

#	ユースケース検証	実施概要
1	**静的地図生成**	お台場エリアを走行する車両の前方カメラ画像を収集し、地図生成アプリと画像処理基盤にて、静的地図を生成する。
2	**障害物検知**	お台場エリアを走行する車両が障害物を（疑似的に）検知し、障害物が撮影された動画データをクラウド側へ送信する。クラウド側は、障害物特定を行った後、障害物がある道路を走行予定の車両へ障害物が存在する旨の通知を行う。
3	**渋滞検知**	お台場エリアを走行する車両データから道路レベルの渋滞データを作成し、当該道路を走行する車両へ動画送信を依頼する。クラウド側は動画からレーンレベルの渋滞の有無を判定し、渋滞するレーンを走行予定の車両に渋滞を通知する。

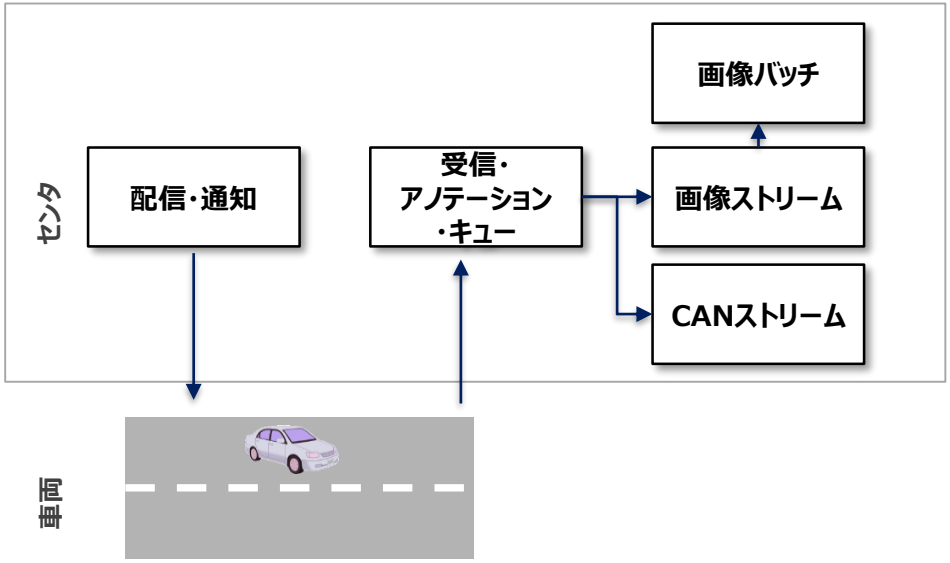
静的地図生成ユースケース検証

検証概要

当該ユースケースは、車両に搭載したカメラ動画の収集から静的地図格納までの一連のシステムを実フィールド環境にて動作させることにより静的地図生成の技術課題を抽出する。

検証概要

オペレータが指示したエリア（お台場）を走行中の車両から、地図生成用の動画データが収集できることを確認する。
センタ側は車両への動画データの収集ができること、および収集した動画データに対して一連の静的地図生成処理が動作することを確認する。



検証観点・内容

1. 機能評価
   静的地図生成の一連の流れを確認する。

2. リアルタイム性評価
   オペレータ指示によるデータ収集から静的地図生成までの処理時間を確認する。

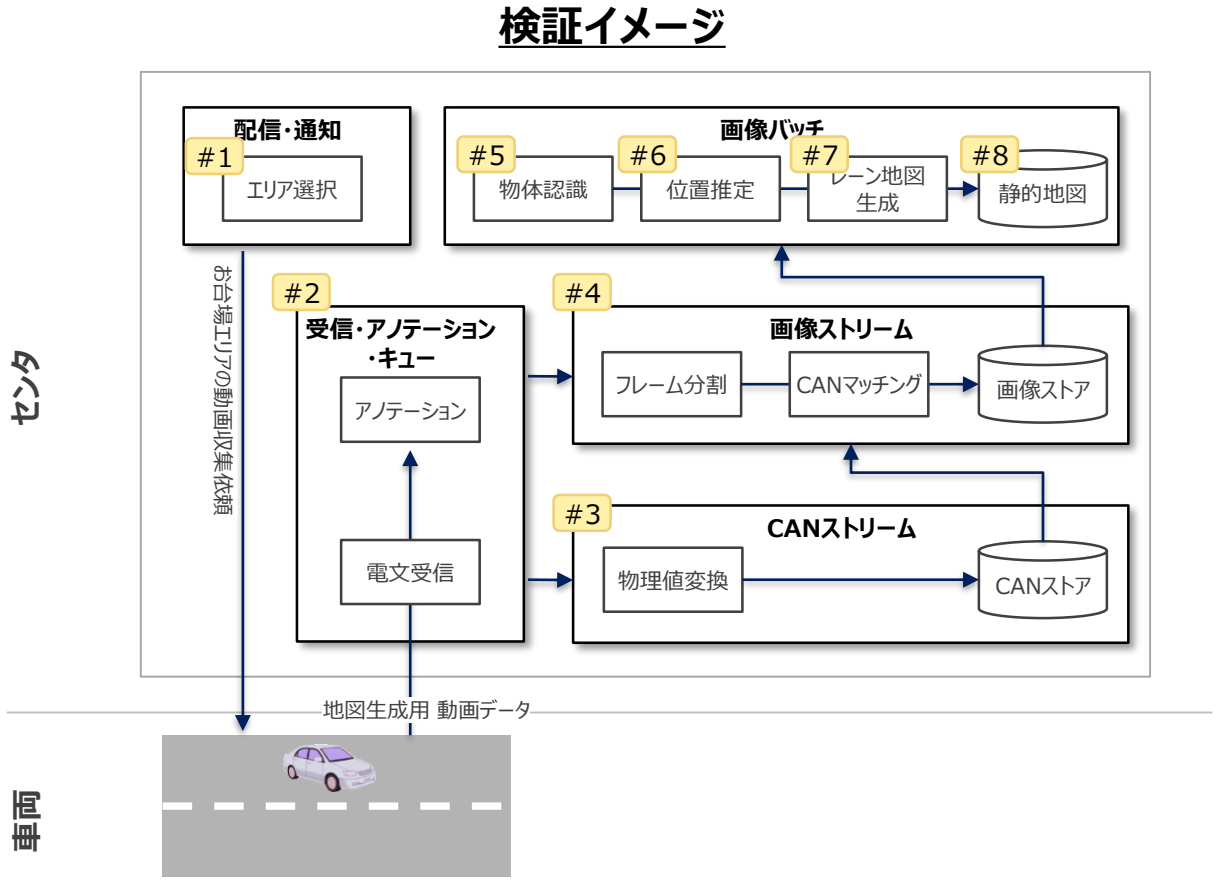
検証１．内容・結果

機能検証では、配信・通知からレーン地図生成までの一連の流れ（＃1～8）を実車を用いて検証を行い、全ての機能が問題なく動作することを確認した。

検証対象

分類	#	機能
配信通知	1	配信通知処理
受信・アノテーション	2	電文受信・アノテーション・キュー
CANストリーム	3	CANストリーム処理
画像ストリーム	4	画像ストリーム処理
画像バッチ	5	物体認識
	6	位置推定
	7	レーン地図生成
	8	静的地図への格納

検証イメージ



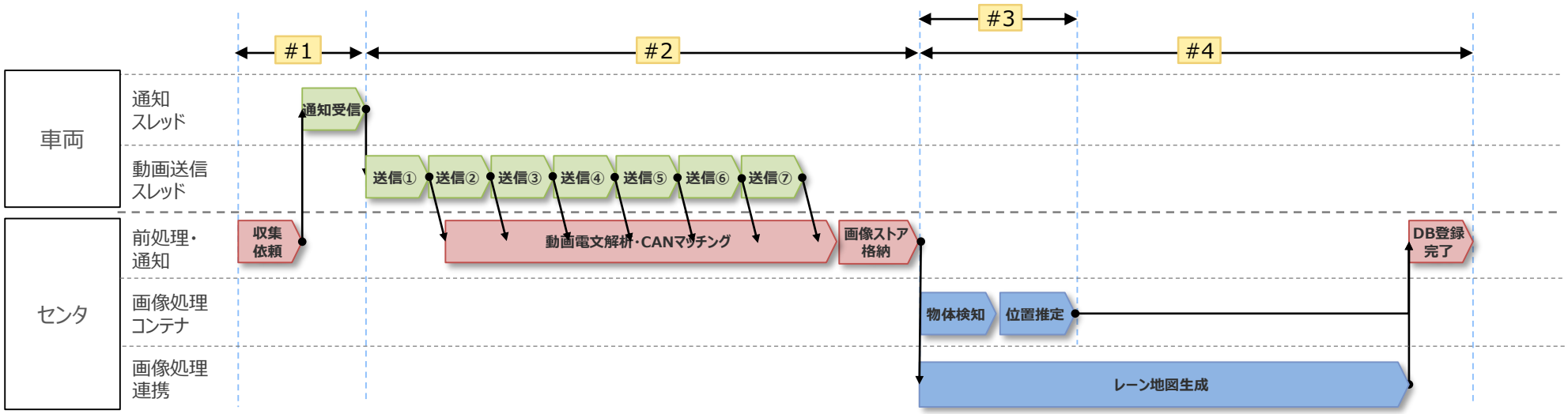
検証２．内容

地図生成時間測定では、動画収集依頼～静的地図生成の各処理毎の時間を計測・分析した。

検証内容

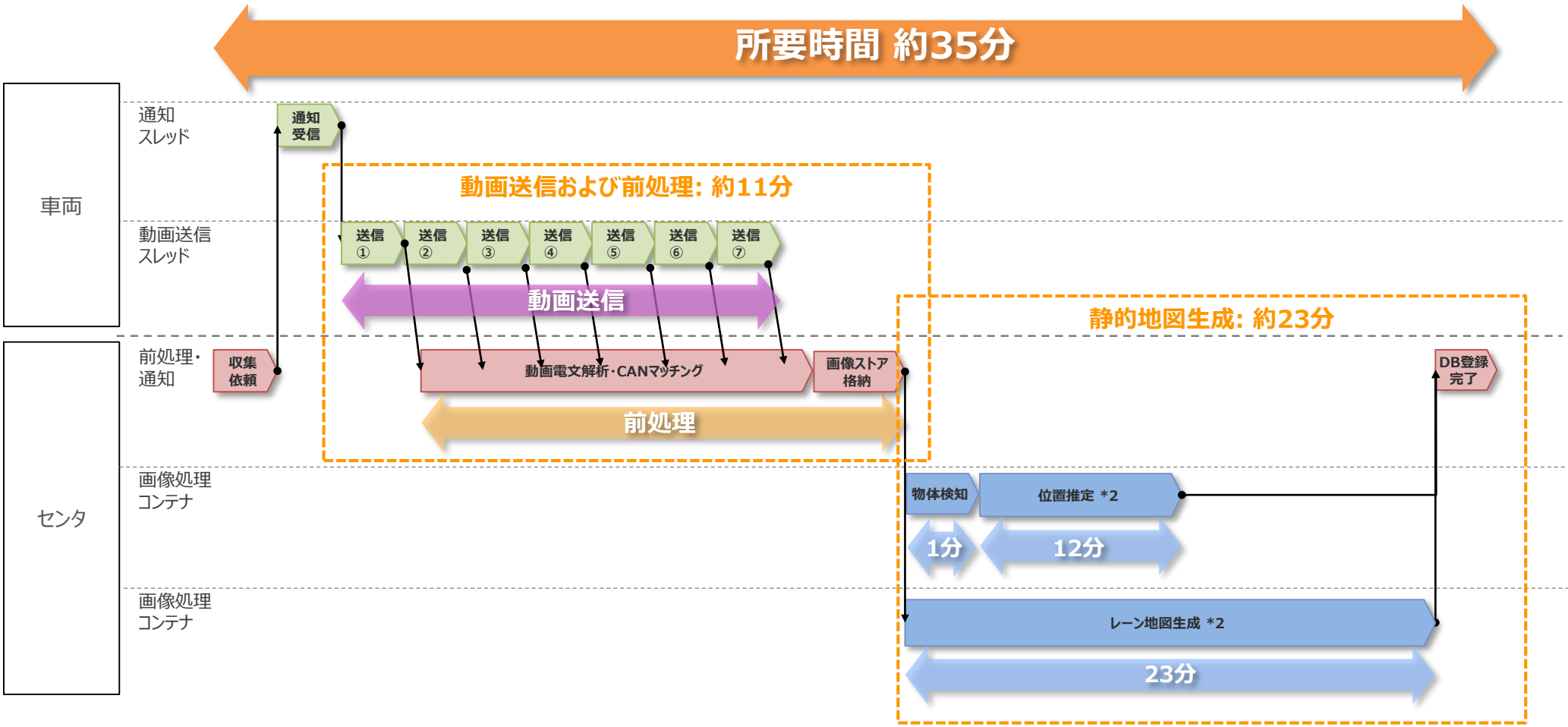
分類	#	検証項目	内容
動画収集依頼	1	動画収集依頼時間	基盤側で動画収集依頼通知を発行してから初回の動画送信開始までにかかった時間
静的地図生成・登録	2	動画送信～前処理時間	全チャンクの動画送信から前処理(画像ストアに格納する処理)が完了するまでにかかった時間
	3	物体検知処理時間	物体認識および全チャンクの動画送信にかかった時間
	4	レーン地図生成時間	前処理(渋滞検知コンテナに投げるまでのデータ整形等)にかかった時間 (チャンク到達待ち含む)

検証イメージ



検証２． 結果

実証実験の試験走行ルート*1において、動画収集依頼から静的地図生成までの実質所要時間は約35分*2であった。



*1：車両1台で約2km走行。詳細は「Appendix:実証実験の試験走行ルート」を参照
*2：Visual SLAMによる位置推定処理を直列で処理した場合、追加で12分程度かかる。本検証ではレーン地図生成と並列で実行しているため、実質所要時間には変化はない。

まとめ

検証内容・観点	まとめ	課題
1. 機能評価	画像収集から静的地図生成まで、一連の流れを実車を用いて確認することができた。 画像収集から静的地図生成までの処理は以下となる。 ・配信通知処理 ・電文受信・アノテーション ・CANストリーム処理 ・画像ストリーム処理 ・物体認識 ・位置推定 ・レーン地図生成 ・静的地図への格納	本検証では基本的な機能動作は確認できたが、実用化に向けては以下の課題がある。 • 前方カメラ（単眼カメラ）およびGPSデータのみから生成した地図に対する精度測定および精度向上 • 現在位置をベースにした、車両配信用の地図データ抽出および地図データ配信する処理の実現 • 収集エリアをメッシュ想定としていたが、道路レベルもしくはレーンレベルなど効率的な収集の実現
2. リアルタイム性評価	画像収集依頼から地図生成までの所要時間は、手動での実行間隔も含め実質的には約35分であった。 • 動画送信および前処理時間：約11分 • 静的地図生成時間：約23分	ー

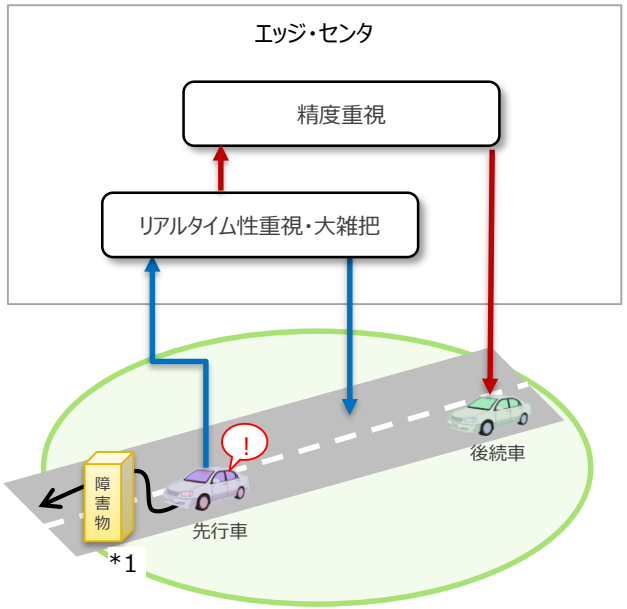
障害物検知ユースケース検証

検証概要

当該ユースケースは、障害物が映った車載カメラ動画を車両から基盤へ送信し、基盤側で特定した上で車両にフィードバックするものであり、リアルタイムなストリーム処理の課題抽出を目的としている。

検証概要

車両が送信した車載カメラ動画から基盤が障害物を特定し、後続車両に通知する。通知は2段階で行う。
　第1段階通知：リアルタイム性重視
　第2段階通知：精度重視



検証内容・観点

1. 機能評価
   障害物の検知から通知まで一連の流れが動作することを確認する。

2. リアルタイム性評価
   車両が障害物を検知してから7秒以内に、後続車両にリアルタイム性を重視した通知ができることを確認する。

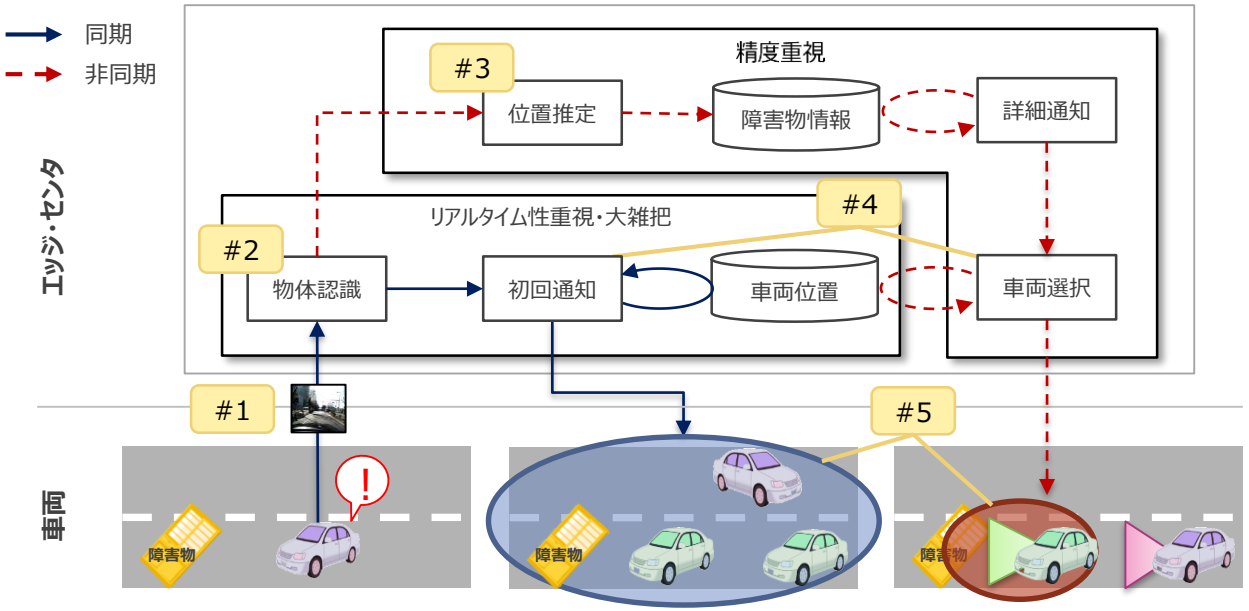
*1：公道で障害物を発生させることが難しいため、信号機を障害物と見立てて検証を行う。

検証１．内容・結果

機能評価では、障害物の検知から通知まで一連の流れを実車を用いて検証を行った。
機能検証項目を以下の通りとし、全て問題なく動作することを確認した。

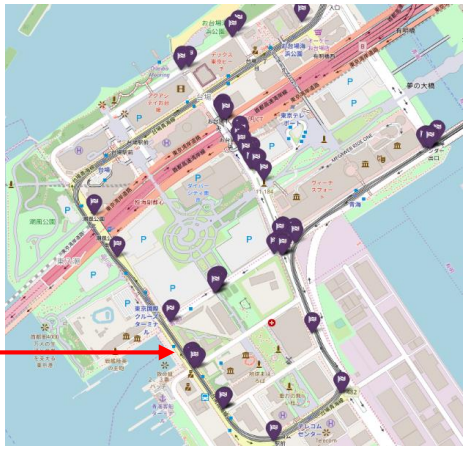
検証内容

#	実施内容
1	車両(疑似DCM)から送信された障害物の動画データが受信できているか
2	物体認識処理により信号機が特定できているか
3	位置推定処理により信号機の位置が特定できているか
4	障害物通知対象の車両を検索できているか
5	動画収集依頼通知およびコンソールへの出力ができているか



#2)画像データの物体認識処理の結果

#3)位置推定処理により抽出された信号機一覧



検証２．内容

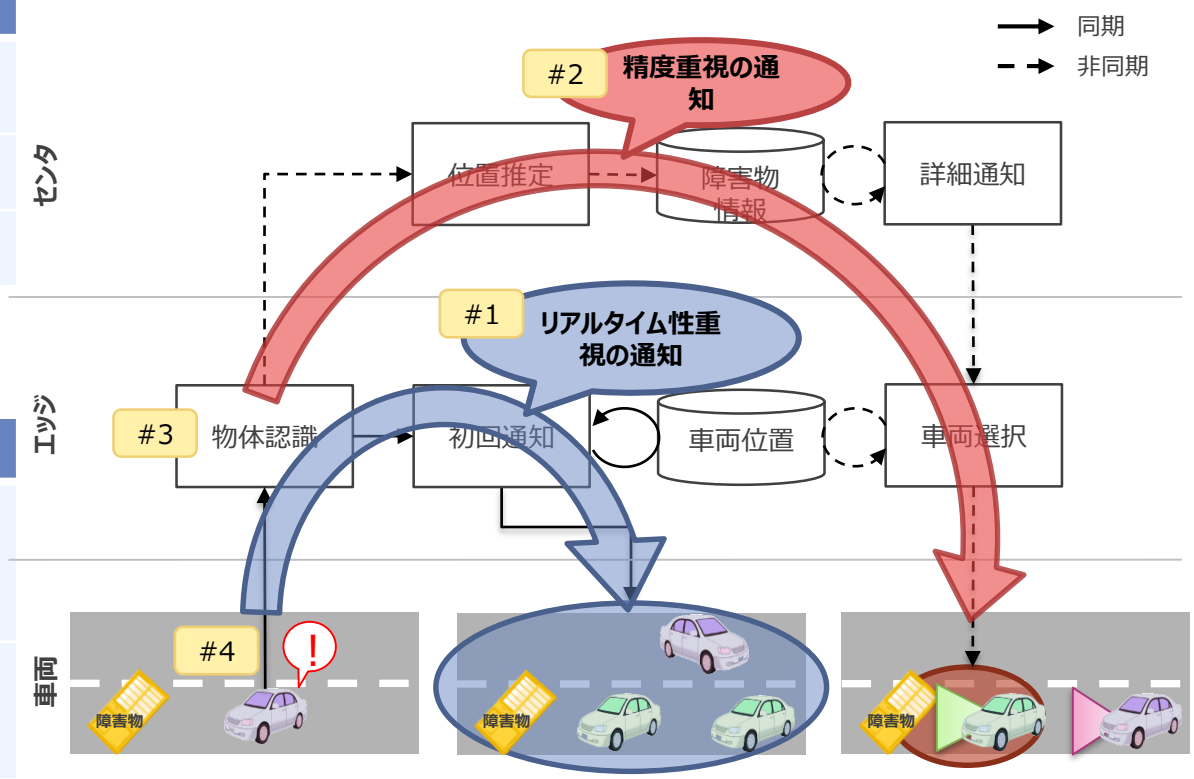
リアルタイム性評価では、障害物の検知から通知までのレスポンスタイムの測定、および評価を行った。

検証内容

#	実施内容
	動画データ収集・画像処理・通知において、下記の時間を測定する
1	リアルタイム性重視(障害物検知～初回通知)
2	精度重視(障害物検知～詳細通知)

ポイントとなる技術要素

#	技術要素
3	エッジへの機能オフロード 高速レスポンスを実施するために必要最低限の機能のみをエッジにオフロードすることで、通知時間の高速化を狙う*1
4	動画の分割送信 チャンク長を短くすることで、基盤側での処理開始待ち時間を短くし、全体処理時間の高速化を狙う*2

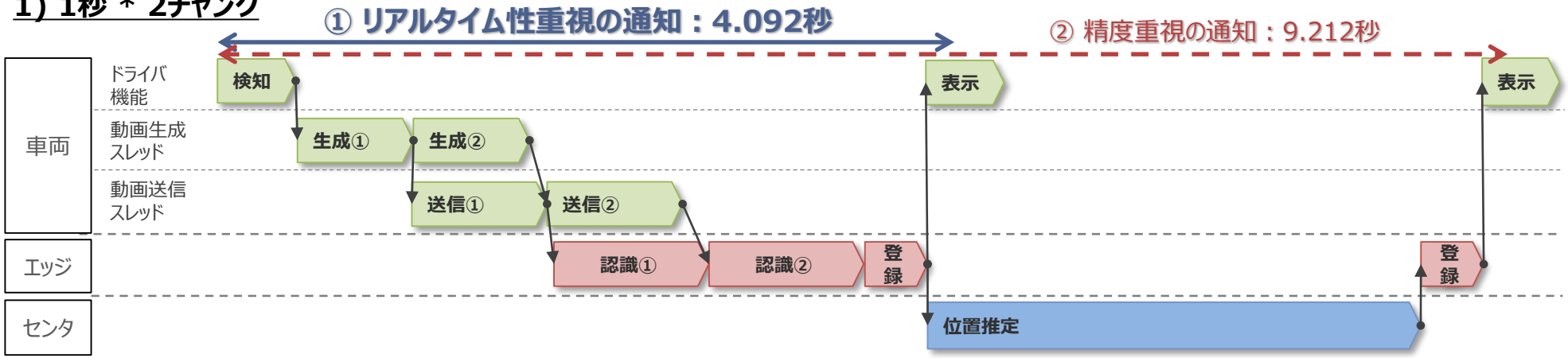


*1:詳細は「Appendix④垂直分散コンピューティング技術」を参照
*2:詳細は「Appendix⑦動画細分化送信技術」を参照

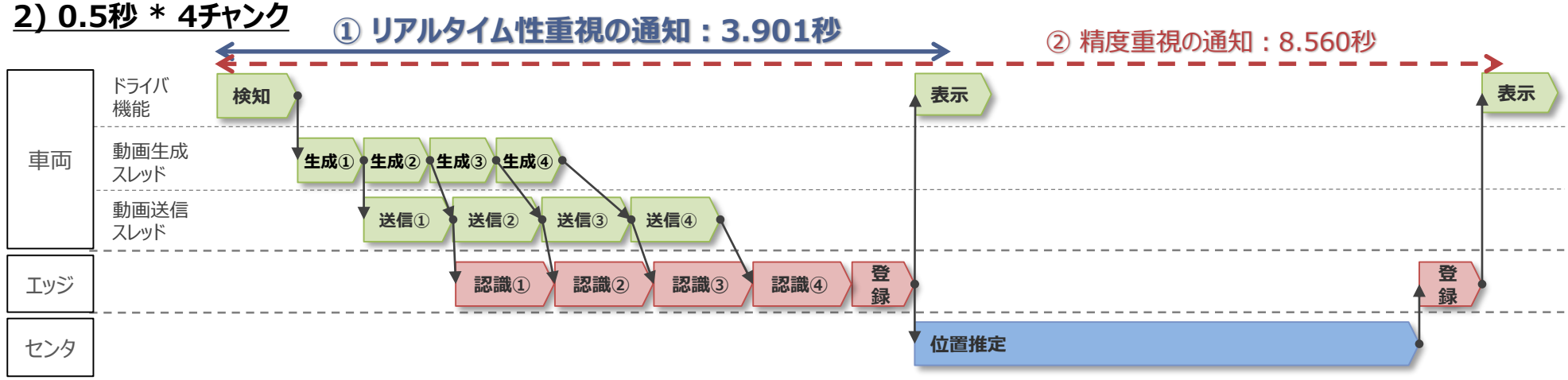
検証２． 結果

動画の分割送信を2パターン行い、障害物検知から通知までの処理時間を測定した結果を以下の矢羽根に示す。
リアルタイム性を重視した通知に関しては、目標とする7秒以内[*1]を達成することができた。

1) 1秒 * 2チャンク



2) 0.5秒 * 4チャンク



*1:本資料では複数回試行した中から最速値を記載しているが、いずれの試行も7秒以内達成を確認した。

まとめ

検証内容・観点	まとめ	考察・課題
1. 機能評価	障害物検知から周辺車両への通知まで、一連の流れを実車を用いて確認することができた。 2段階通知における本検証での達成状況は以下となる。 1. リアルタイム性重視 – 物体認識処理による**障害物**の特定 – **車両位置**を起点とした、周辺車両検索および通知 2. 精度重視 – 位置推定処理による**障害物位置**の特定 – **障害物位置**を起点とした、周辺車両検索および通知	本検証では基本的な機能動作は確認できたが、実用化に向けては以下の課題がある。 • 物体認識、および位置推定の精度検証 – 現物との誤差（位置、数） – 天候や時間帯における精度劣化度合い • 障害物モニタリングを想定した収集範囲の限定 通知を受け取った全車両から動画データを収集した場合、転送・処理コストがかかるため、収集範囲を限定する必要がある 例えば、高速時空間データ管理技術*1におけるポリゴン範囲の車両抽出と車両データ選択的収集アルゴリズム*2を組み合わせることで、通知対象車両を障害物に接近する車両のみに限定することが可能である
2. リアルタイム性評価	以下の技術を組み合わせることで、目標とする7秒以内の通知を実現することができた。 1. エッジへの機能オフロード 2. 動画の分割送信 また、本資料では触れていないが、以下の技術も適用することでエッジを有効活用するアーキテクチャを実現している。 3. 高速時空間データ処理技術 *1 4. 処理ノード動的選択技術 *3	• 車両への機能オフロードによる高速化およびトラフィック量の削減効果 • エッジ機能の負荷分散およびアプリケーション配置最適化 – 負荷状況に応じた一部処理のオフロード – エッジ間データ連携

*1：詳細は「Appendix① 高速時空間データ管理技術と時空間データ高速検索技術」を参照
*2：詳細は「Appendix② 車両データ選択的収集アルゴリズム」を参照
*3：詳細は「Appendix④ 垂直分散コンピューティング技術」を参照

渋滞検知ユースケース検証

検証概要

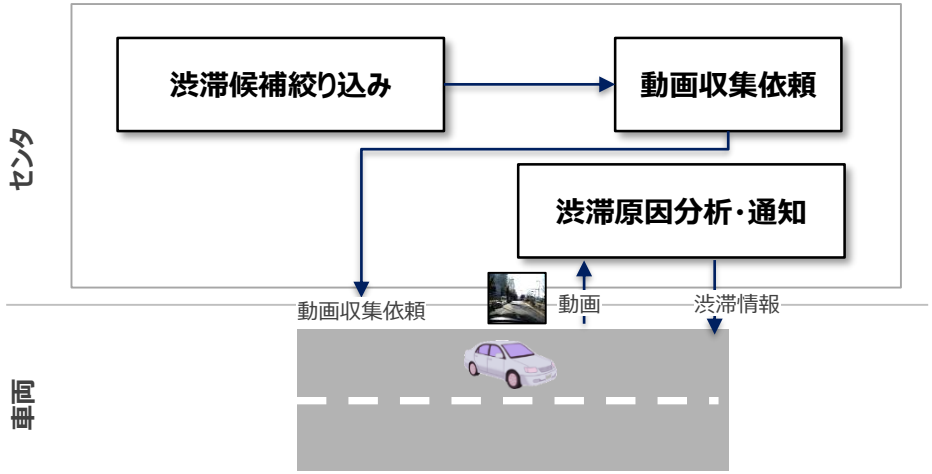
当該ユースケースは、車両に搭載したカメラ動画の収集から分析結果の通知までの一連のシステムを実フィールド環境にて動作させることにより、リアルタイム性を必要とするストリーム処理の技術課題を抽出する。

検証概要

車両が送信した車載カメラ動画から基盤が渋滞を特定し、後続車両に通知する。一連の流れは2ステップで行う

ステップ１． 渋滞候補絞り込み～動画収集依頼

ステップ２． 動画収集依頼～渋滞原因分析・通知



検証内容・観点

1. 機能検証
   渋滞の検知から通知まで一連の流れを確認する。

2. レスポンスタイム検証
   動画収集依頼～渋滞原因分析・通知が、５分以内に処理できることを確認する。

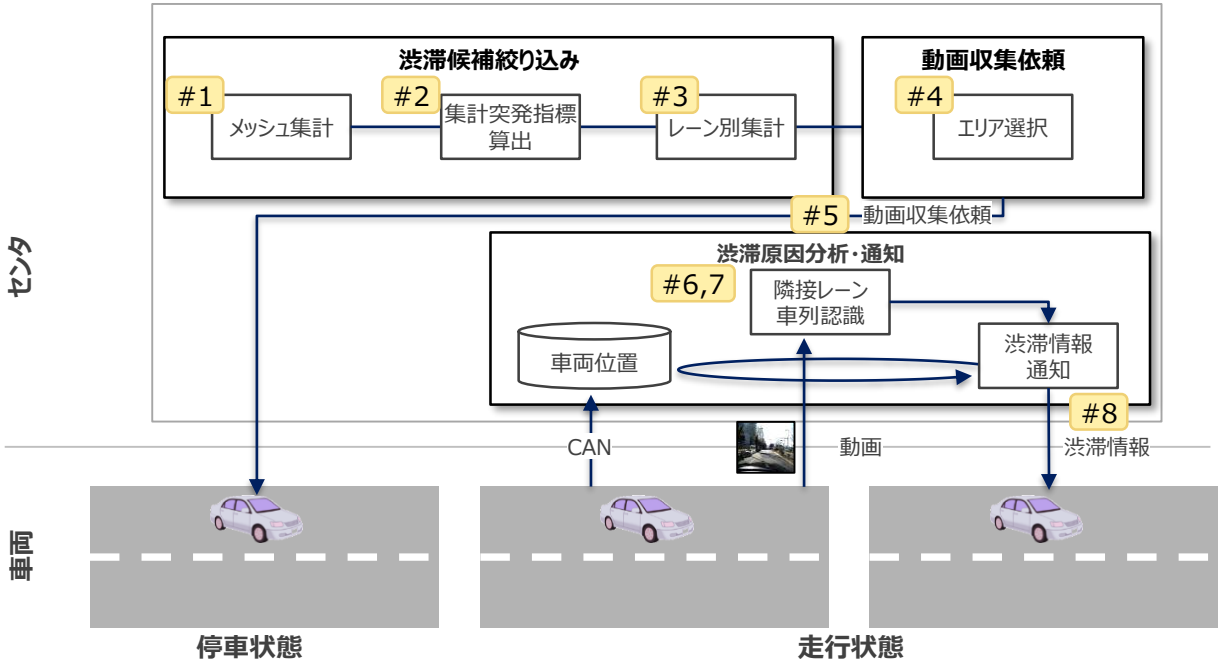
検証１．内容・結果

機能検証では、渋滞の検知から通知まで一連の流れ（＃1～8）を実車を用いて検証を行い、全て問題なく動作することを確認した。

検証対象

分類	#	機能
渋滞候補絞り込み	1	メッシュ集計
	2	集計突発指標算出*1
	3	レーン別集計
動画収集依頼	4	エリア選択
	5	動画収集依頼
渋滞原因分析・通知	6	隣接レーン車列認識
	7	渋滞原因分析
	8	渋滞情報通知

検証イメージ



*1：詳細は「Appendix⑤ 集計突発指標算出技術」を参照

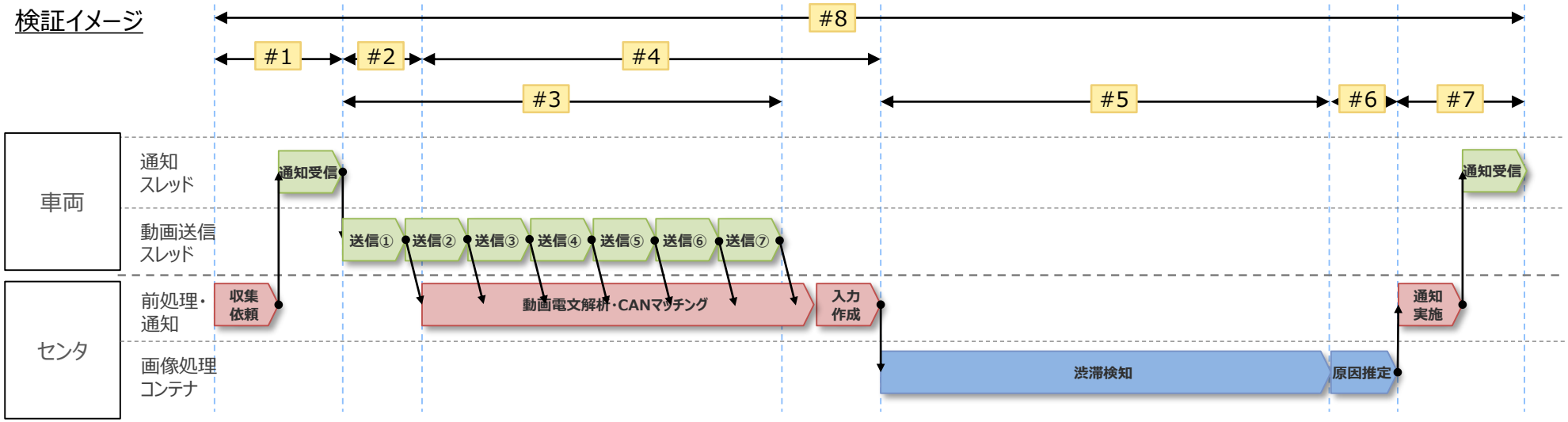
検証２．内容

レスポンスタイム検証では、動画収集依頼～渋滞原因分析・通知の、各処理毎の時間を計測・分析した

検証内容

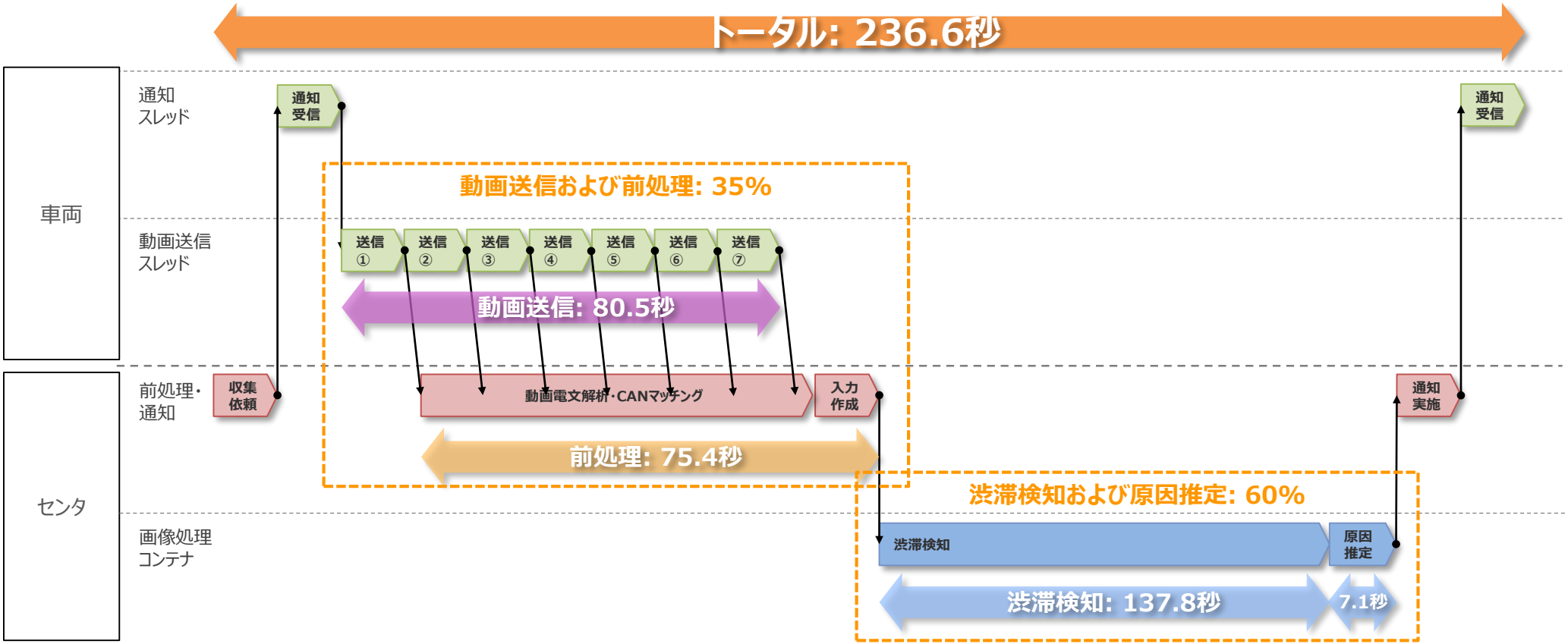
分類	#	検証項目	内容
動画収集依頼	1	動画収集依頼時間	基盤側で動画収集依頼通知を発行してから初回の動画送信開始までにかかった時間
渋滞原因分析・通知	2	初回チャンク送信待ち時間	初回の動画送信を開始してから、そのチャンクが基盤で処理開始されるまでにかかった時間
	3	動画送信時間	全チャンクの動画送信にかかった時間
	4	前処理時間	前処理(渋滞検知コンテナに投げるまでのデータ整形等)にかかった時間 (チャンク到達待ち含む)
	5	渋滞検知処理時間	渋滞検知処理にかかった時間
	6	原因推定処理時間	渋滞原因推定処理にかかった時間
	7	通知時間	基盤側で通知処理を開始してから車両に到達するまでにかかった時間
	8	トータル時間	系全体でかかった時間

検証イメージ



検証２． 結果

実フィールド環境で処理が成功した最速ケース*1に対して、各処理毎の時間の計測・分析した。
処理時間の約60%が渋滞検知処理、約35%が動画送信および前処理であることが分かった。



*1：実フィールド環境での試行結果概要は右記。（処理成功：6回　最速236.6秒　最遅311.4秒　平均264.7秒）

まとめ

検証内容・観点	まとめ・考察	課題
1.機能検証	• 集計突発指標算出技術については、机上検討を行い、レーン集計対象を最高で8割削減できることを確認した • 実車から収集した画像を用いて本機能の動作確認を行い、実世界で発生している車列を認識できることが確認できた。 • 既存の車両選択ロジックや通知ロジックとも連携させ、適切な車両からの動画収集や適切な車両への渋滞情報通知まで通しで実現できた。	• 学習を用いた集計突発指標テーブルの更新 • エリア選択の自レーン・隣接レーン・対向レーンの絞り込みおよび最適レーンの選択 • 動画収集車両選択の最適車両の選択ロジックの検討 • CANデータの欠損・解像度不足への対応 • 複数動画の検知結果の意味づけ • エッジコンピューティングとの融合 • 処理ノード動的選択技術との融合
2.レスポンスタイム検証	• 次フェーズ以降に並列分散が可能となるようにSpark[*1]やKubernetes[*1]のフレームワーク上に構築し、各技術が連携して動作することを確認した。 • 70秒動画(60秒動画収集依頼通知で収集した動画)に対する渋滞検知および原因分析処理に関して、4~5分で実施できることを確認した。 • 処理時間に関する分析を行い、改善ポイントを複数抽出することができた。特に、I/F改善により、基盤側での処理開始待ちを少なくすることで、1分程度の短縮が見込めると考える。	• 集計突発指標算出の並列処理化によるスループット・TATの向上 • 集計突発指標算出のI/F改善によるスループット・TATの向上 • 集計突発指標テーブルの外出しによるスケーラビリティ向上 • 隣接レーン車列認識の前処理におけるファイル書き込み排除によるスループット・TATの向上 • 隣接レーン車列認識の前処理との機能統合によるスループット・TATの向上 • 隣接レーン車列認識の余分な動画処理の回避によるスループット・TATの向上

*1:Spark/Kubernetesを用いたアーキテクチャの詳細は「Appendix.アーキテクチャ(詳細版)」を参照。

基盤検証

検証概要

基盤検証は、コネクティッドカー向けICT基盤のアーキテクチャにおける5つのコンポーネントについて検証[*1]する。

#	基盤検証	実施概要
1	受信・アノテーション・キュー	500万台走行ピーク/送信間隔10秒以下の条件にて、車両から送信されるCAN・動画データの収集基盤における限界負荷の測定を行う。
2	CANデータ処理	500万台走行ピークでの条件下で、車両から送信されたCANデータをストリーム処理し、ダイナミックDBに高速格納可能か検証する。 格納された位置データに関しては、マイクロバッチ処理による集計を行い、性能傾向を把握する。
3	画像データ処理	車両から送信された動画に対するストリーム処理、および、画像ストアに蓄積された大量の画像に対するバッチ処理において、画像処理基盤の限界点を検証する。
4	配信・通知	車両への動画収集依頼通知およびデータ配信における、対象車両の検索・選択する処理の性能検証を行う。
5	ネットワークエッジ	複数のNW機能を持たせた基盤システム（NWエッジ）によって、車両の移動やエッジ拠点の局所的な負荷集中等に対応するデータフロー制御を実現できるか、機能及び性能検証を行う。

*1:システム構成の詳細は、「Appendix 基盤検証のシステムについて」を参照。

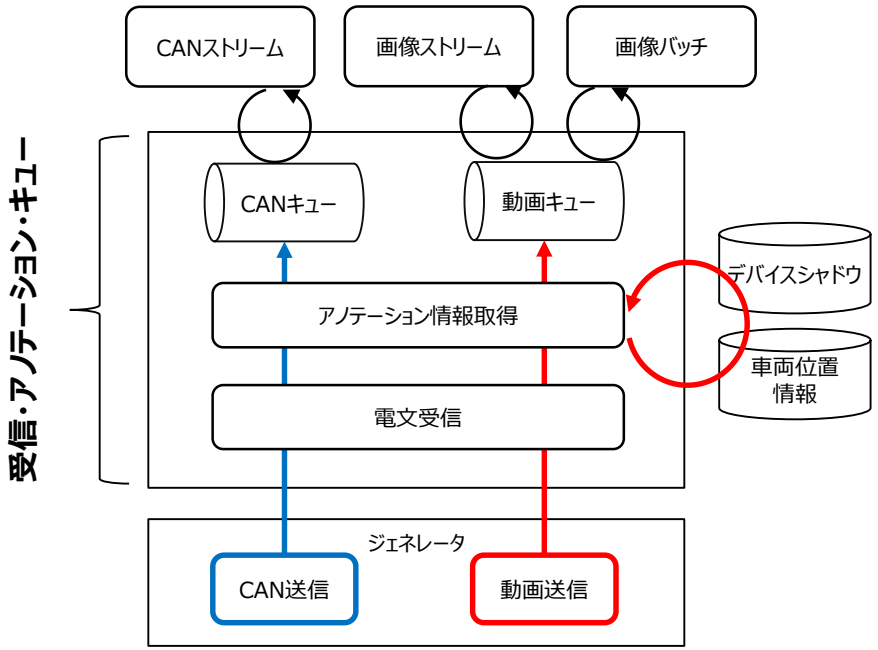
受信・アノテーション・キュー基盤検証

検証概要

500万台走行ピーク/送信間隔10秒以下の条件にて、受信・アノテーション・キュー基盤における限界負荷の測定を行う。

検証概要

車両から送信される膨大なデータ(CAN・動画)に対して、リアルタイムでの処理が求められる。本検証では、ストリーム処理に活用するための前処理部分となる、受信・アノテーション・キュー基盤の性能測定を行う。



検証内容・観点

受信・アノテーション・キュー基盤において、データ種別ごとに性能検証を行う。

1. CANデータのストリーム処理（前処理）
   同時走行車両台数60万台(500万xピーク稼働率12%)の達成可否、および各コンポーネントの性能傾向を確認する。

2. 画像データのストリーム処理（前処理）
   a. 車両の位置や速度情報、およびエリア内同時処理数を用いた動的優先度付けの限界負荷と処理時間を測定する。

   b. キューサーバの限界負荷の測定とリソースボトルネック箇所を確認する。

検証１．内容

受信・アノテーション・キュー基盤において、目標車両台数60万台(500万xピーク稼働率12%)の達成可否、および各コンポーネントの性能傾向を確認する。

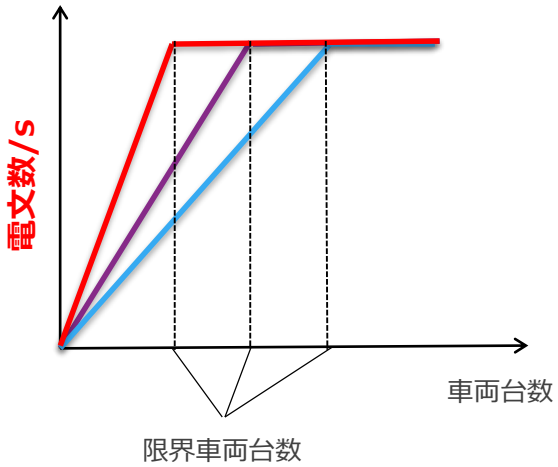
検証内容

CANデータの性能傾向の確認にあたっては、コンポーネントを2つに分けて検証する。

- 受信・アノテーションサーバの性能傾向
- キューサーバの性能傾向

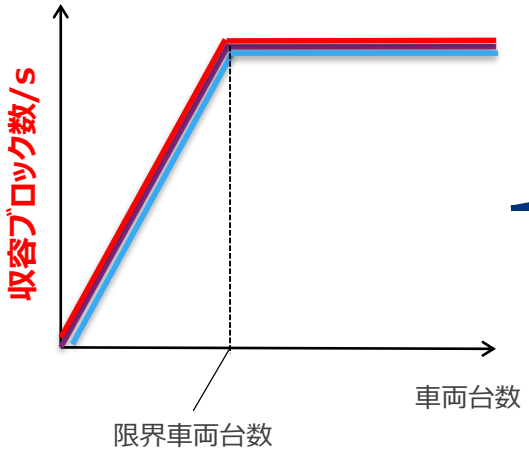
受信・アノテーションサーバの性能傾向

[仮説]
電文送信間隔を短くすると電文数が増えるため、限界車両台数は少なくなる想定である。



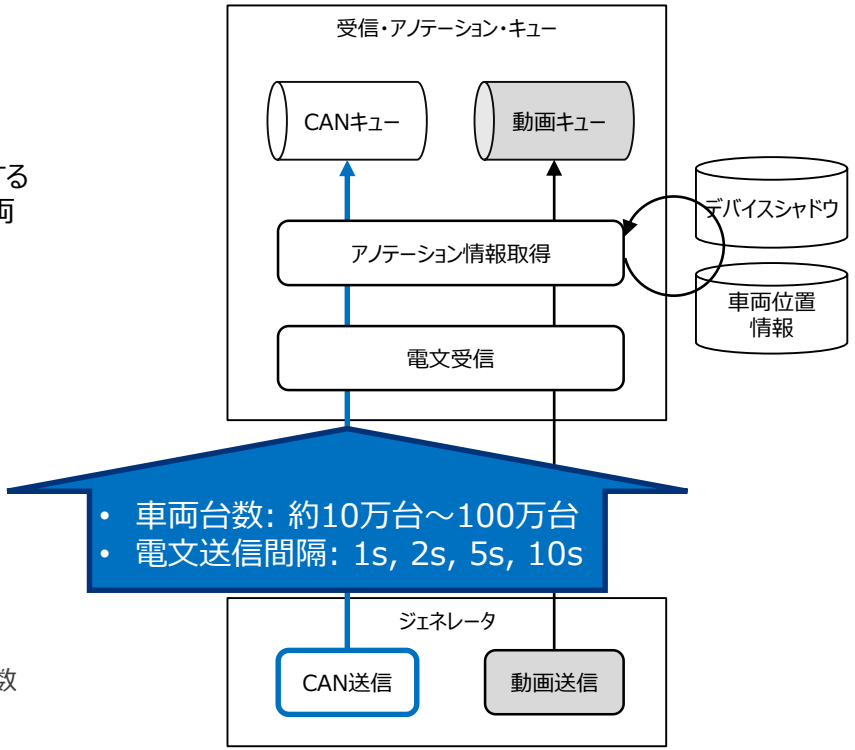
キューサーバの性能傾向

[仮説]
電文送信間隔を変えてもトータルで処理する収容ブロック数は変わらないため、限界車両台数は一定となる想定である。



検証環境

電文送信間隔を1秒から10秒の範囲の4種類で、それぞれの限界処理台数を測定する。



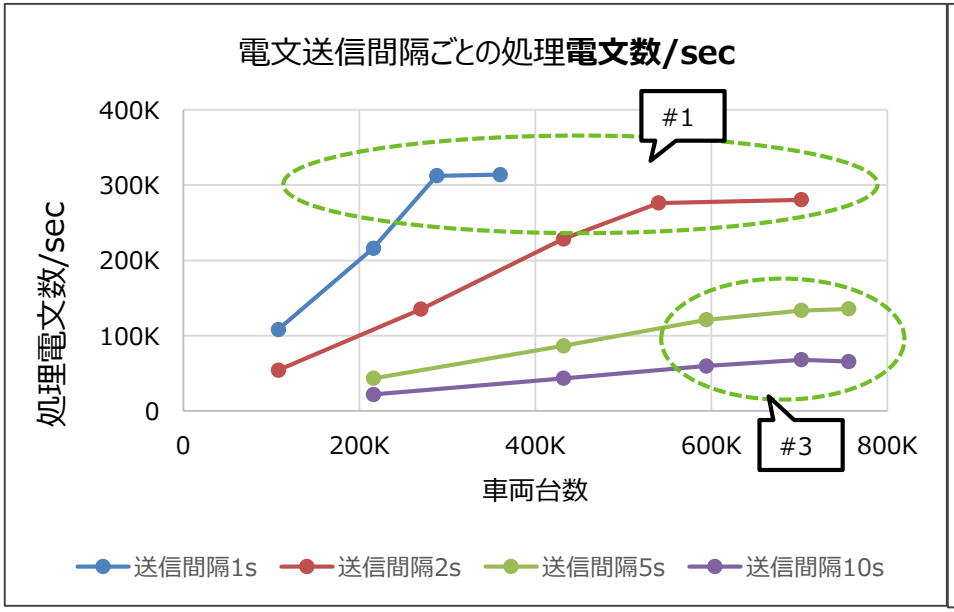
検証１．結果

電文送信間隔ごとの限界車両台数を測定した。
電文送信間隔が5秒以上の場合に、性能目標である60万台を達成できることがわかった。

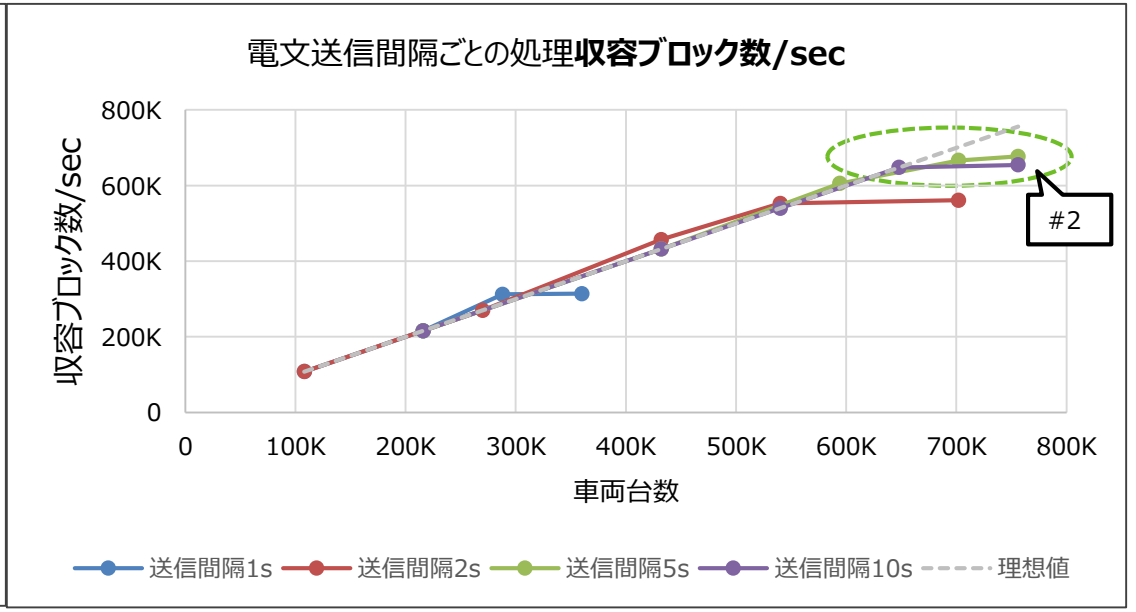
電文送信間隔[s]	限界車両台数	達成状況	ボトルネック
1	306,000	×未達成	受信・アノテーションサーバ
2	540,000	×未達成	受信・アノテーションサーバ
5	648,000	○達成	キューサーバ
10	648,000	○達成	キューサーバ

#1　30万電文数/secが性能限界
#2　64万収容ブロック数/secが性能限界
#3　受信・アノテーションサーバがボトルネックではない

受信・アノテーションサーバの性能傾向



キューサーバの限界性能



検証２-a.　内容

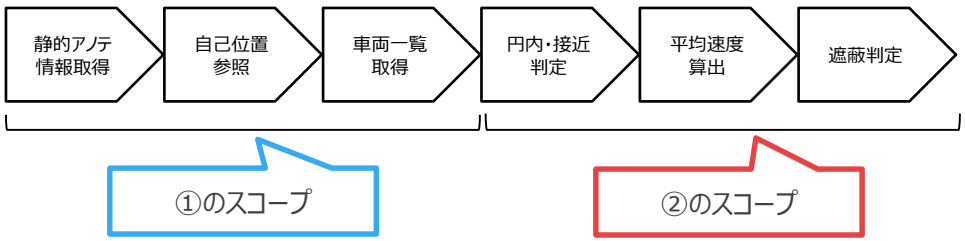
車両の位置や速度情報、およびエリア内同時処理数を用いた動的優先度付けの、限界負荷と処理時間を測定する。

検証内容

動的優先度付け処理は、車両位置情報から集計した統計値をもとに優先度を判断する。統計取得範囲(時空間メッシュ内、および、イベント地点周辺の円)に存在する車両台数により性能が変化すると考えられるため、以下の条件で車両台数を変化させて性能検証を行う。

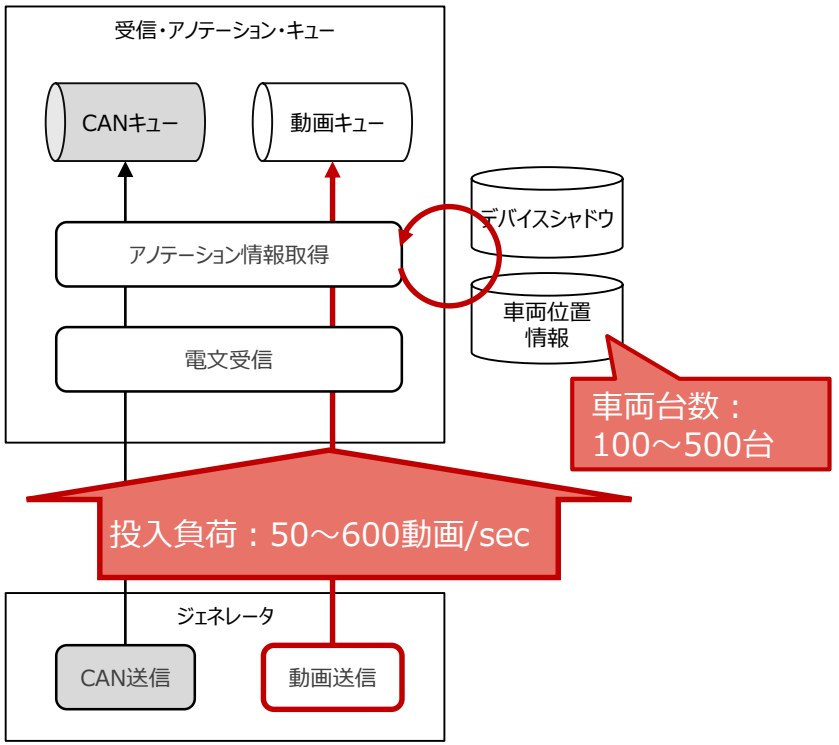
①　メッシュ内車両数の増加
メッシュ内車両数を増加させ、車両情報一覧の取得処理による負荷傾向を測定する。

②　円内車両数の増加
メッシュ内車両数を固定した状態で、**円内車両数を増加**させ、取得した車両一覧に対する平均速度算出や遮蔽判定処理の負荷傾向を測定する。



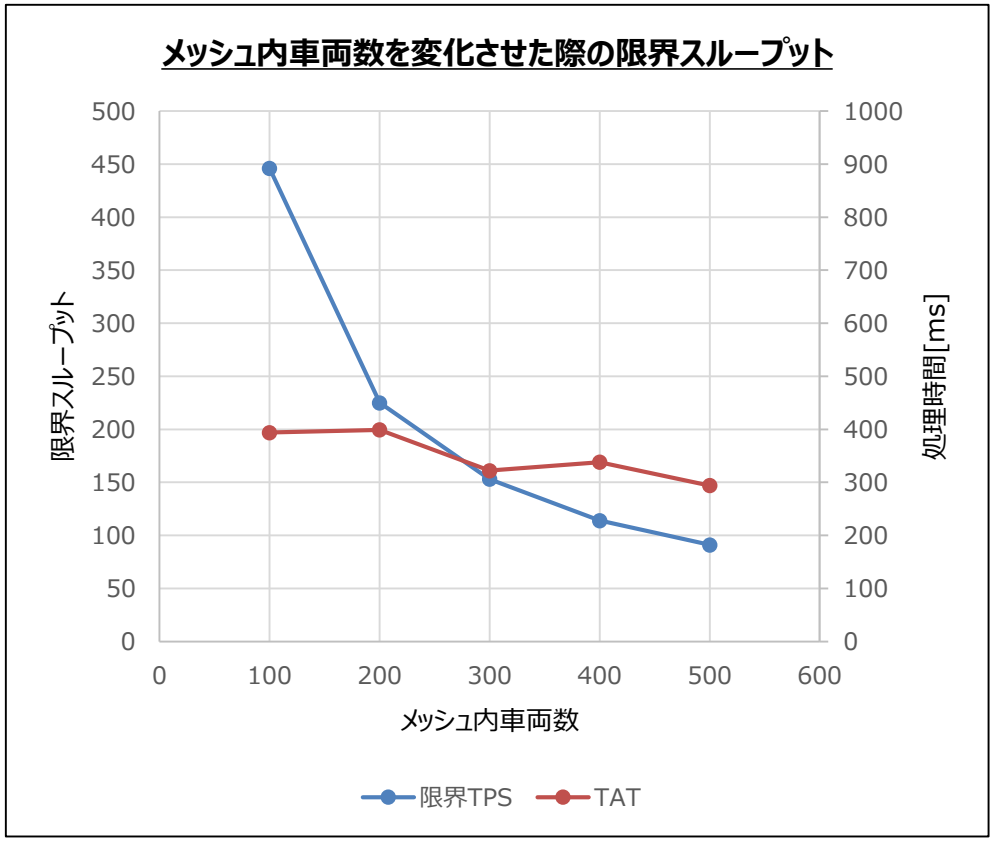
検証環境

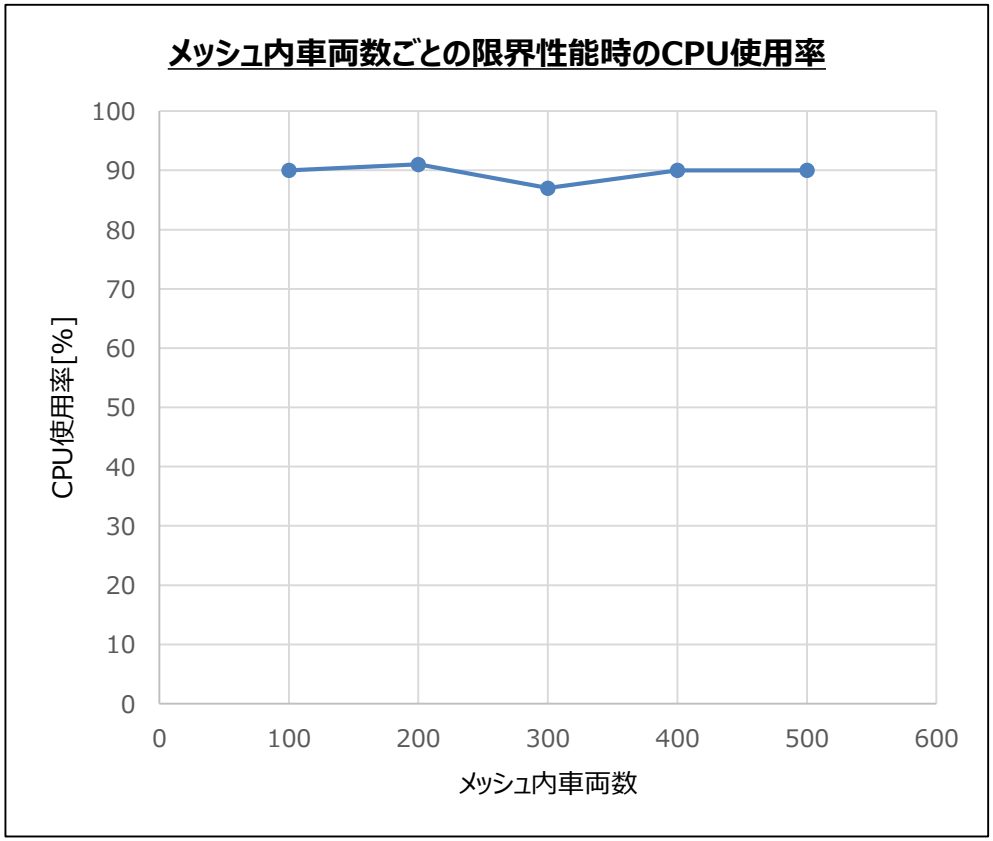
検索対象の車両位置情報を事前に設定し、秒間当たりの負荷に対する限界処理台数を測定する。



検証２-a.　結果(1/2)

負荷測定の結果、メッシュ内車両数を増加させると限界スループットは低下することがわかった。
また、ボトルネックとなるリソースは受信・アノテーションサーバのCPU使用率であった。



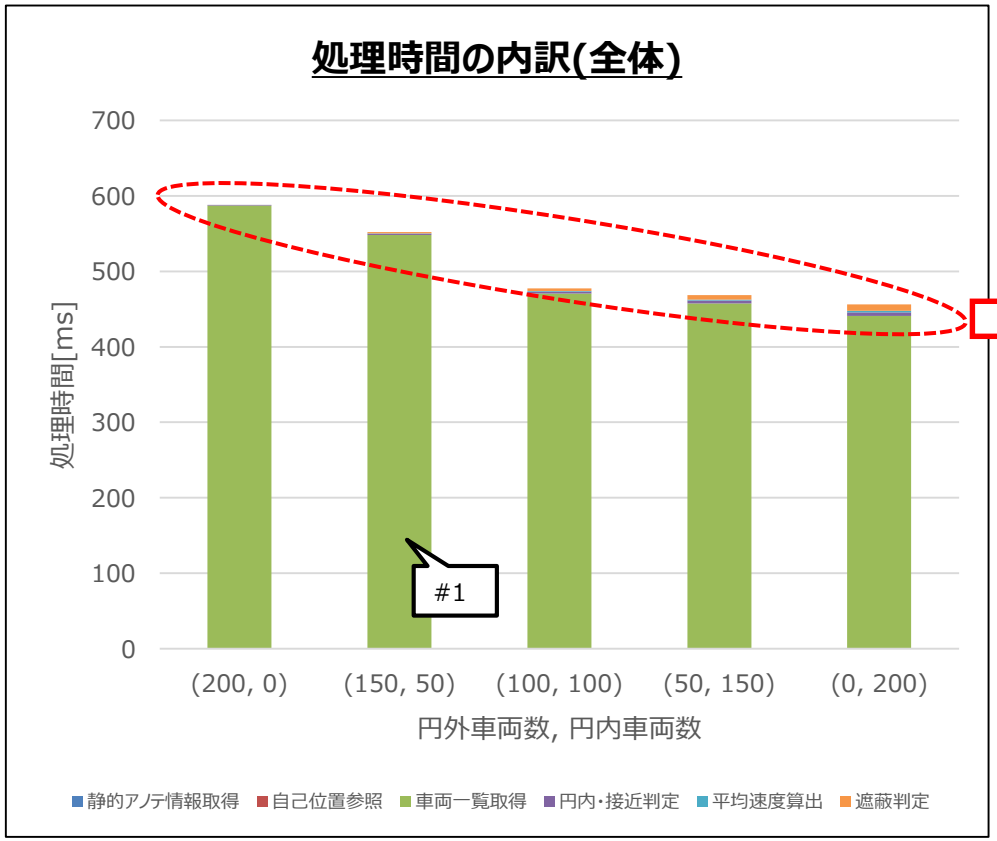


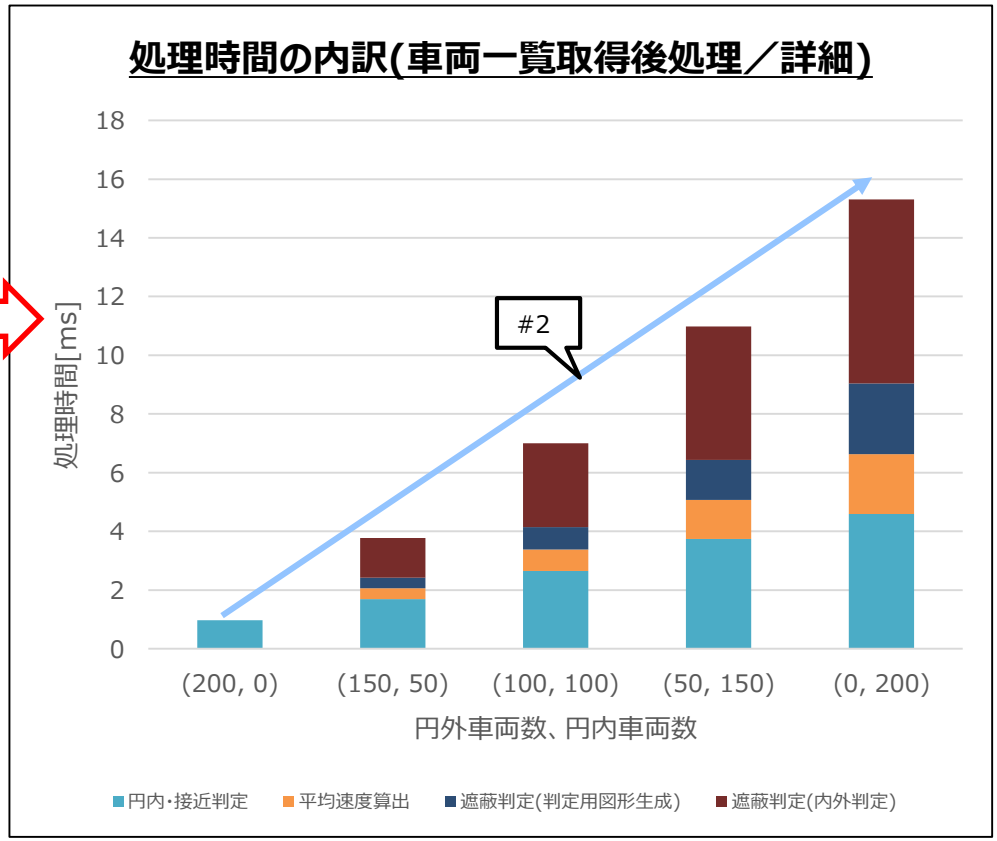
検証２-a. 結果(2/2)

処理時間については、下記2点の特徴が観測できた。

1. 全体処理時間について、ダイナミックDBから車両一覧を取得する処理が95%以上を占める。
2. 円内車両に実施される処理(平均速度算出・遮蔽判定)については、円内車両台数に線形増加する。

静的アノテ情報取得 → 自己位置参照 → 車両一覧取得 → 円内・接近判定 → 平均速度算出 → 遮蔽判定





#1 ほぼ全てが車両一覧取得処理(車両数の影響は誤差)

#2 線形増加

検証２-b.　内容

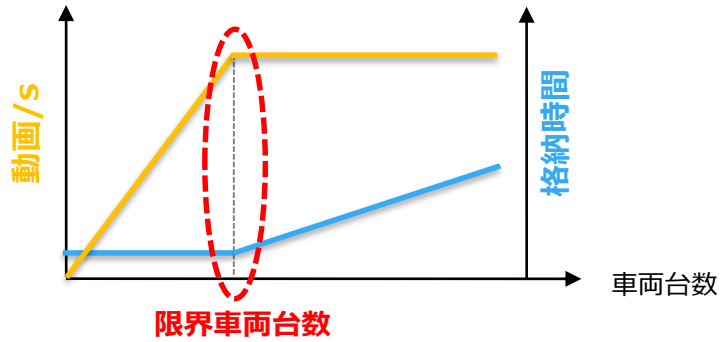
キューサーバの限界負荷の測定とリソースボトルネック箇所を確認する。

検証内容

動画データの定常的に送信し、キューサーバの限界性能値を見極める。ボトルネック箇所をキューサーバとするために、動的優先度付け処理は行わないこととする。

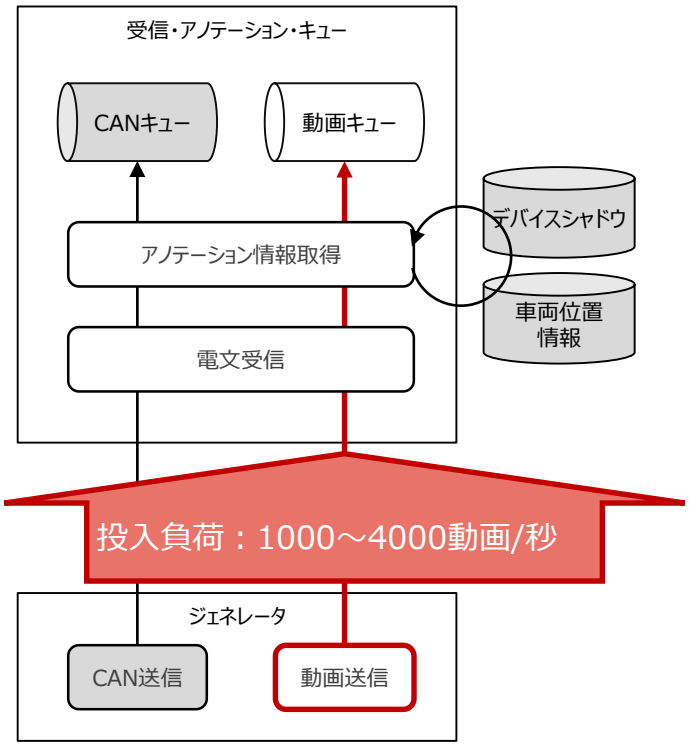
動画データのプロファイル
動画長：2秒（約2MB）
フレームレート：10fps
ビットレート：標準（6200kbps）

キューサーバの限界負荷の見極めポイント
単位時間あたりに処理可能な負荷量、およびキューへの格納時間を測定し、限界負荷量を見極める。



検証環境

秒間当たりの負荷に対する限界処理台数を測定する。



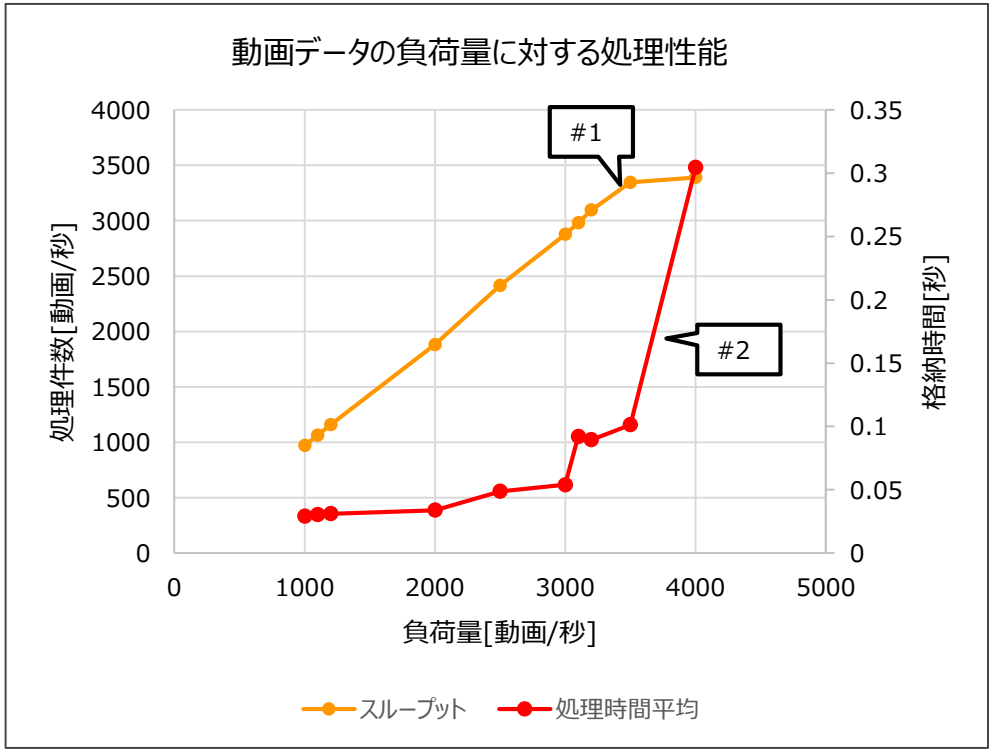
検証２-b.　結果

動画データの負荷量に対するキューサーバの限界処理性能を測定した。

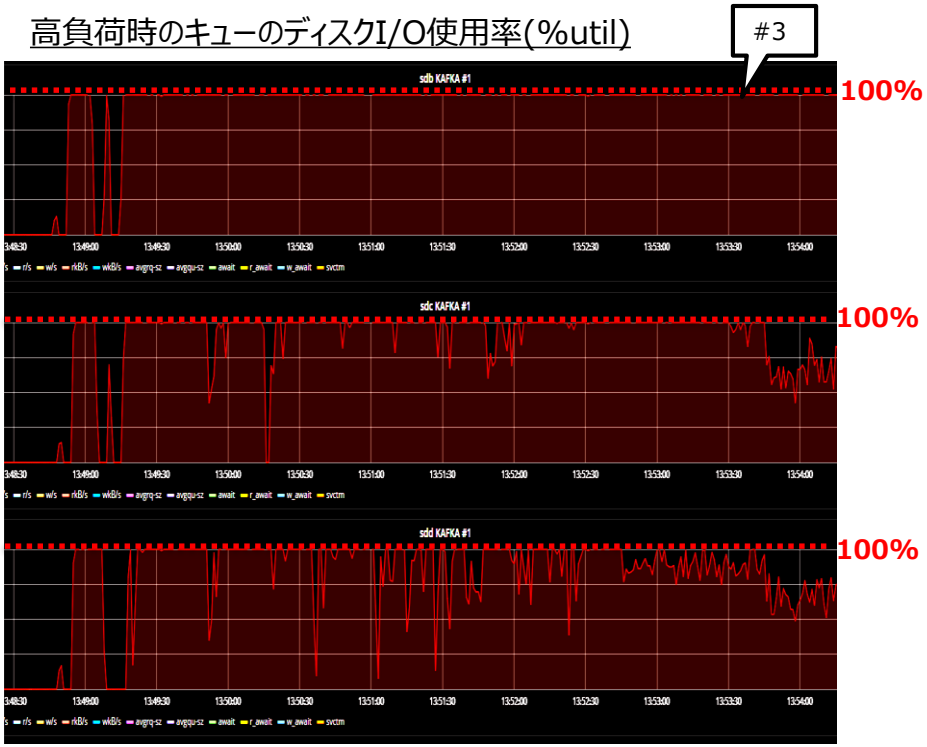
3500[動画/秒]の負荷程度までは処理し続けることができるが、これを超えると処理性能が頭打ちとなることがわかった。

負荷量[動画/秒]	前処理時間[ミリ秒]	キューへの送信待ち時間[ミリ秒]	キューへの格納完了時間[ミリ秒]
2000	0.918	2.121	30.336
3000	1.312	5.700	48.175
4000	2.653	**54.743**	**213.420**

#1　3500[動画/秒]程度で頭打ち
#2　負荷の増加に対して処理時間が長時間化
#3　ディスクI/Oがボトルネック



高負荷時のキューのディスクI/O使用率(%util)



まとめ

検証内容・観点	まとめ
1. CANデータのストリーム処理 （前処理）	電文送信間隔5秒、10秒のときに、性能目標である60万台を達成した。また、性能傾向は仮説通り、電文送信間隔によってボトルネック箇所が変化することも確認できた。 • 電文送信間隔：1秒、2秒の場合 – ボトルネック箇所は**受信・アノテーションサーバ** – 電文送信間隔を短くするほど、受信する電文数が増えるため • 電文送信間隔：5秒、10秒の場合 – ボトルネック箇所は**キューサーバ** – 電文送信間隔を変えてもトータルで処理するデータ量は変わらない
2. 画像データのストリーム処理 （前処理）	a. 動的優先度付け処理 ダイナミックDBから車両一覧を取得する処理が、全体処理時間の95%以上を占める。また、円内車両に実施される処理(平均速度算出・遮蔽判定)については、想定通り円内車両台数に線形増加する。 b. キューサーバへの格納処理 3500[動画/秒]の負荷程度までは処理できることを確認した。 また、その際のボトルネック箇所が**キューサーバの**ディスクI/Oであることも分かった。 [共通] 現状のユースケースにおいては、イベント時のみ動画データを送信する想定であり、系の限界性能は後段の画像処理であるため、本結果に問題はないと考える。 但し、定常的に動画データを収集するケースがある場合は、効率的な収集方法を検討する必要がある。

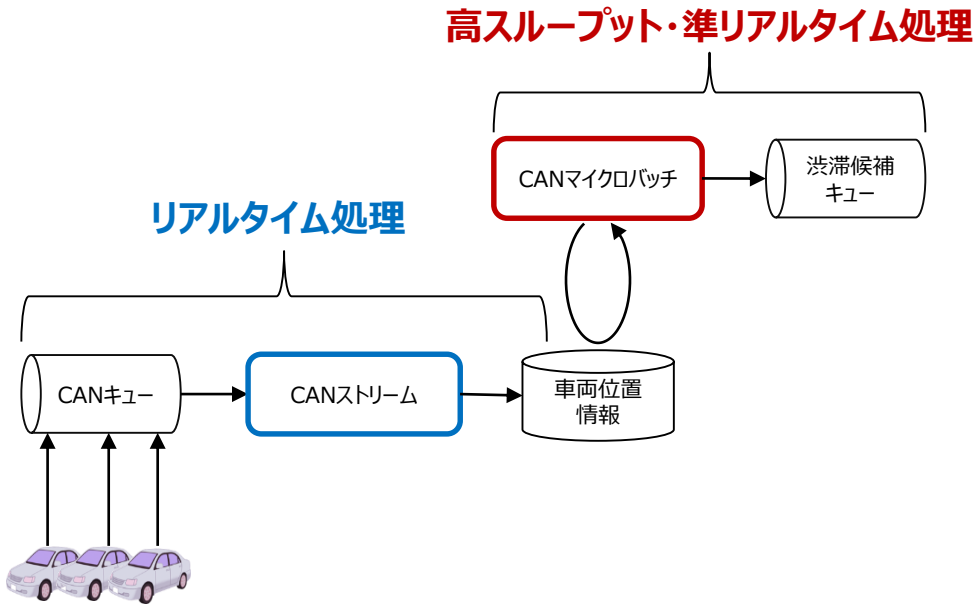
CANデータ処理

検証概要

CANデータ処理基盤について、処理方式ごと（ストリーム処理、マイクロバッチ処理）に性能測定を行う。

検証概要

車両から収集するCANデータは大容量であるが、スループットよりもリアルタイム性を重視する場合もある。よって性能要件に応じて処理方式を決定する必要がある。
本検証では、ストリーム処理とマイクロバッチ処理において、性能測定を行う。



検証内容・観点

1. CANストリーム処理
   a. 定常CANデータの物理値変換処理を例に、リアルタイム性を担保できる限界性能を確認する。

   b. 車両情報の格納処理において、目標車両台数60万台(500万xピーク稼働率12%)の達成可否を確認する。

2. CANマイクロバッチ処理
   メッシュおよびレーン別集計による渋滞絞り込みを例に、車両情報の検索処理時の絞り込み効果を確認する。

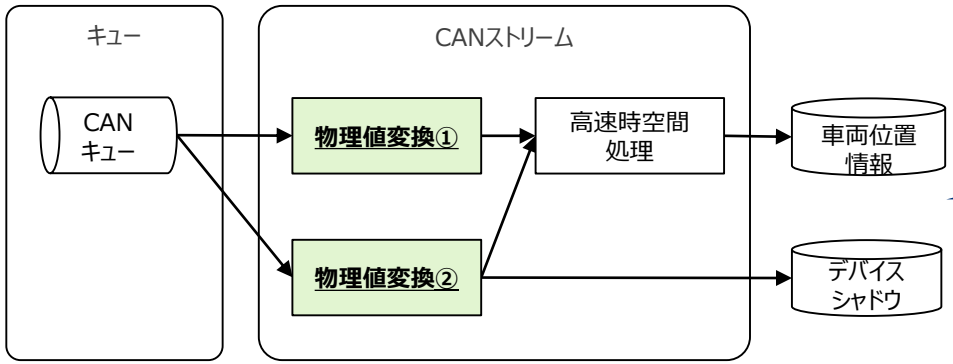
検証１-a.　内容

定常CANデータの物理値変換処理を例に、リアルタイム性を担保できる限界性能を測定する。

検証内容

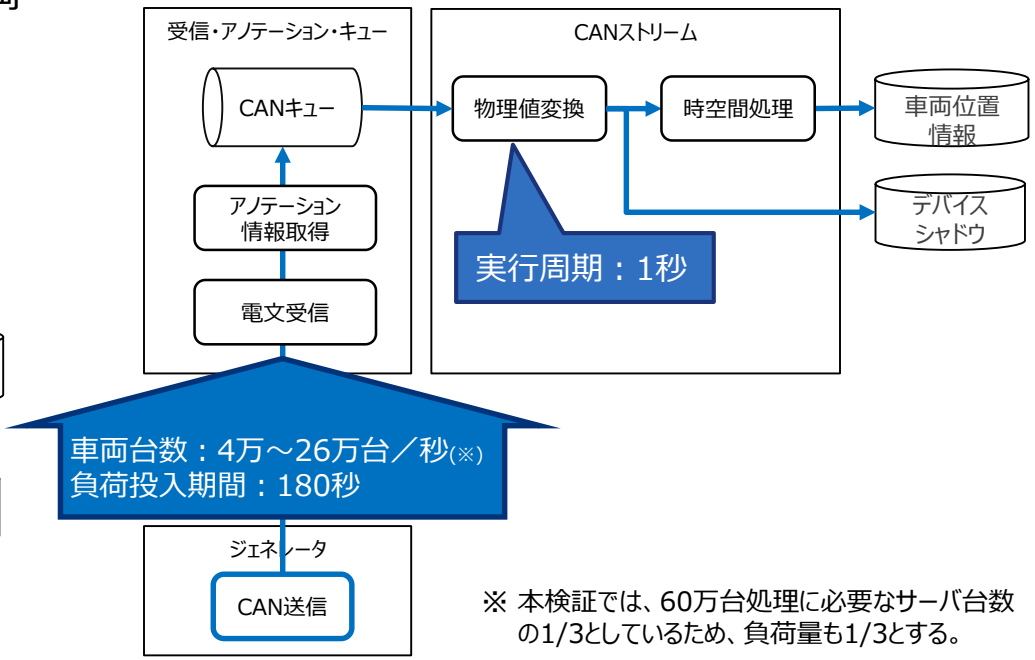
車両の最新位置情報をデバイスシャドウに格納することで、高速検索を実現する。その一方で、物理値変換時にデバイスシャドウへの格納処理が追加となるため、物理値変換全体の性能がどの程度劣化するかを検証する。
一定の処理間隔内に物理値変換処理が完了する、最大の車両台数を限界性能と定義し、以下２パターンの性能比較を行う。

① デバイスシャドウへの書き込みなし
② デバイスシャドウへの書き込みあり



検証環境

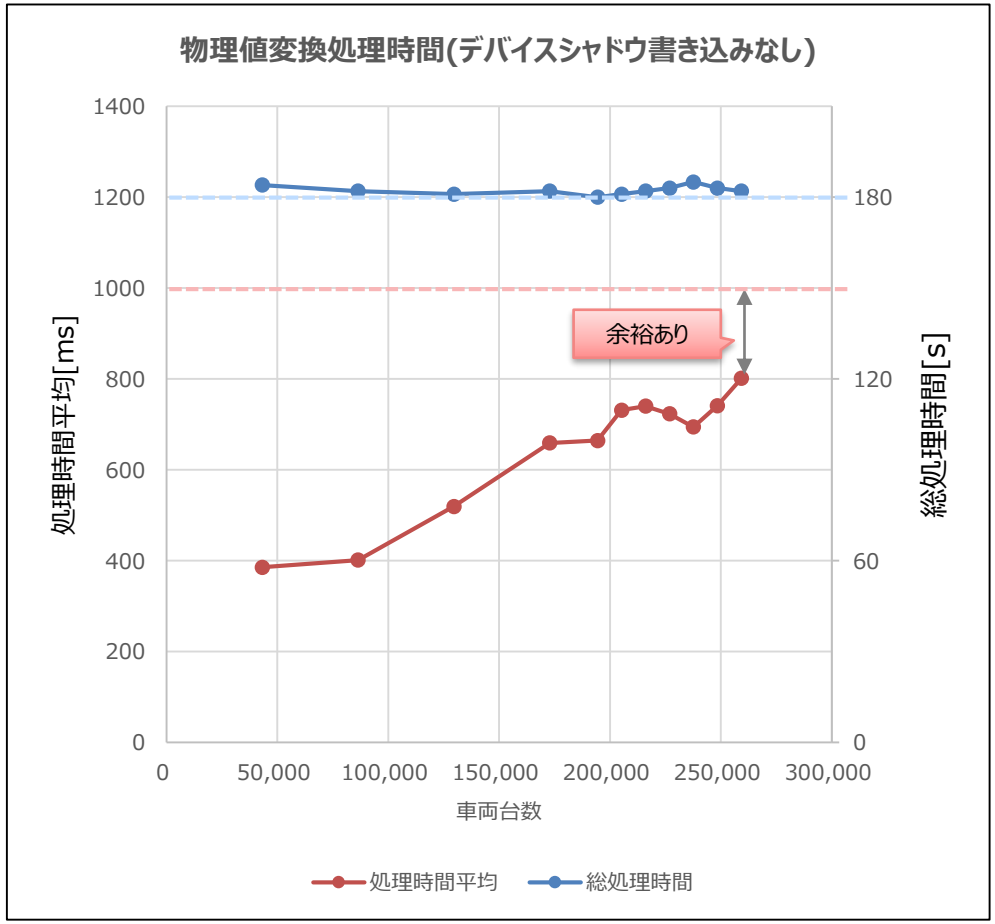
以下の条件を満たす限界車両台数を測定する。
- 1秒分データの処理時間平均が**1秒以内**であること
- 投入した負荷電文の総処理時間が、ほぼ**180秒**であること

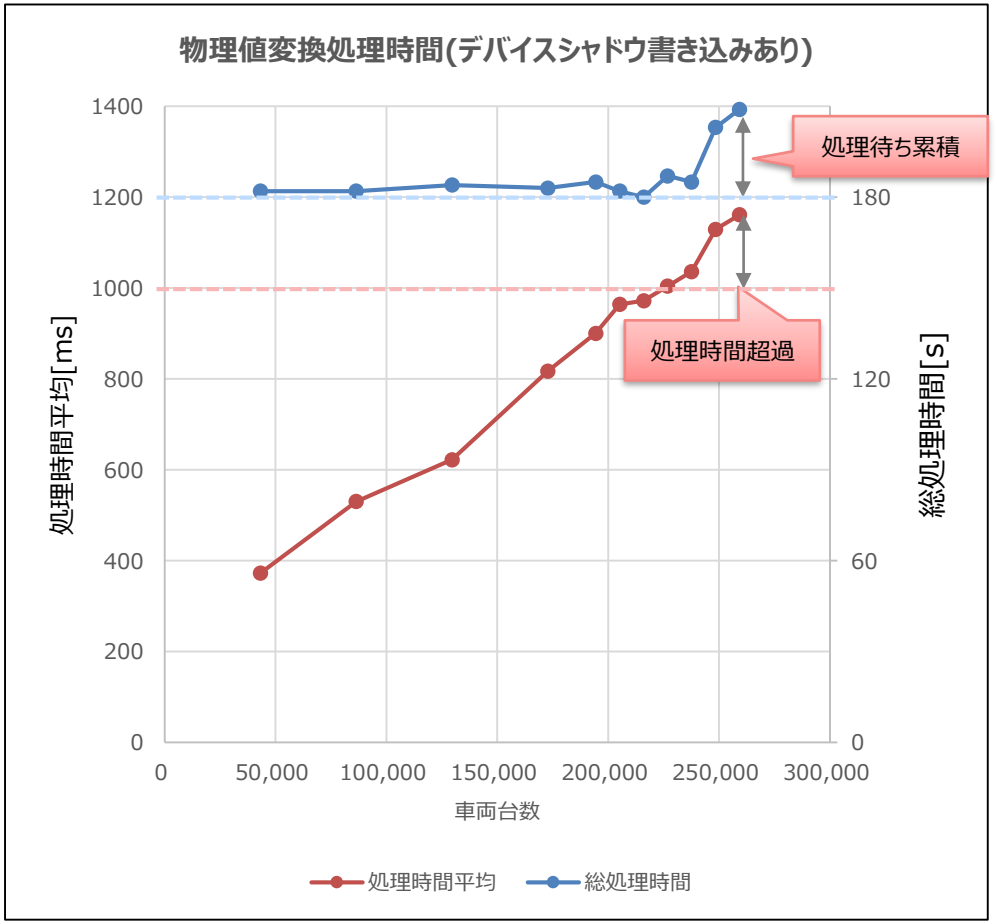


※ 本検証では、60万台処理に必要なサーバ台数の1/3としているため、負荷量も1/3とする。

検証 1 -a. 結果

車両台数を26万台まで増加させ、1秒分データの処理時間平均と投入した負荷電文の総処理時間を測定した。
デバイスシャドウへの書き込み処理を追加することで性能は劣化し、24万台／秒程度で性能限界に達することを確認した。





検証１-b.　内容

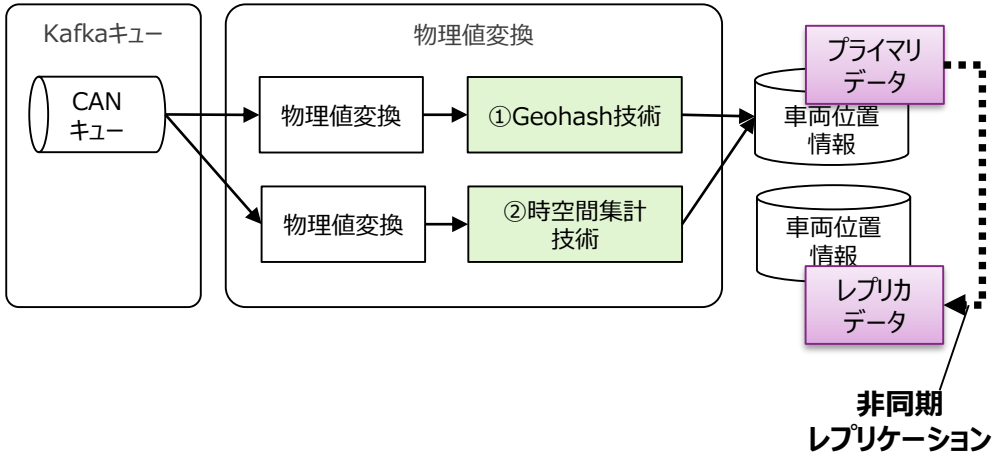
車両情報の格納処理において、目標車両台数60万台(500万x12%)の達成可否を確認する。

検証内容

車両位置情報の格納処理について、以下２つの技術を用いて性能比較を行う。

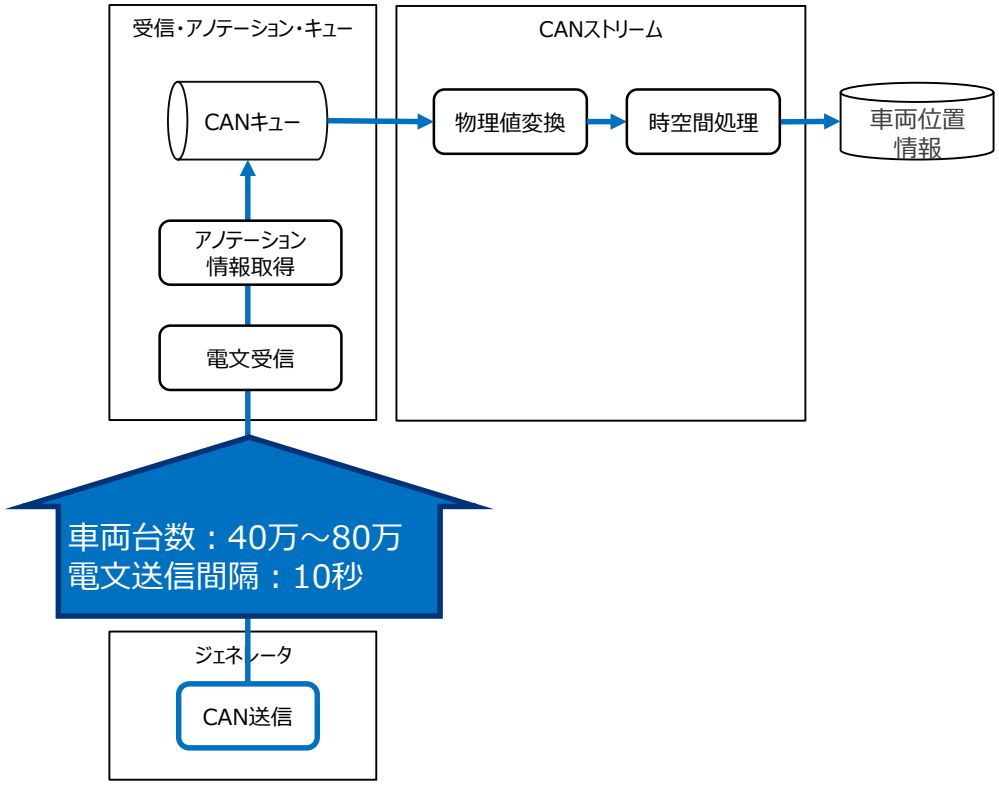
① Geohash技術
② 時空間集計技術*1

また、限界性能は車両位置情報のレプリケーションが滞留せず、かつ処理を継続できる最大負荷とする。



検証環境

秒間当たりの限界処理件数を測定する。



*1:詳細は、「Appendix① 高速時空間データ管理技術と時空間データ高速検索技術」を参照。

検証１-b.　結果

TOYOTA　NTTグループ

限界車両台数を測定し、①Geohash方式は50.4万台／秒、②時空間集計方式は68.4万台／秒まで処理できることを確認した。時空間集計方式を利用することで、 Geohash方式の1.36倍まで格納性能が向上した。

①Geohash方式

車両台数	一定の間隔内での処理 ※1	レプリケーション状況 ※2	達成状況
414,000		○	達成
504,000		○	達成
540,000	○	×	未達成
594,000		×	未達成
648,000		×	未達成
684,000		×	未達成
774,000		×	未達成
864,000		×	未達成

②時空間集計方式

車両台数	一定の間隔内での処理 ※1	レプリケーション状況 ※2	達成状況
414,000	○	○	達成
504,000		○	達成
540,000		○	達成
594,000		○	達成
648,000		○	達成
684,000		○	達成
774,000		×	未達成
864,000		×	未達成

※1：○…処理時間 ≦ 起動間隔であり、継続的に処理可能　×…処理時間 > 起動間隔で、継続的な処理は困難
※2：○…レプリケーションが滞留せずに継続的に処理可能　×…レプリケーションが滞留しており、継続的な処理は困難

検証２．内容

メッシュ集計およびレーン別集計による渋滞絞り込みを例に、車両情報の検索処理時の絞り込み効果を確認する。

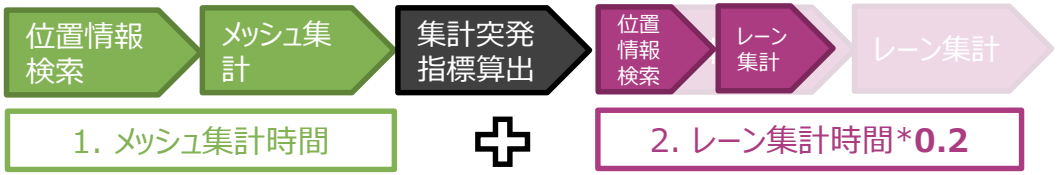
検証内容

メッシュ集計や集計突発指標算出技術を用いることで、後段のレーン別集計の負荷を下げることを目的にしている。
本検証では、以下の集計時間を測定して合計時間を算出し、前段でメッシュ集計を行うことの効果を確認する。

①レーン集計のみを実施(絞り込みなし)

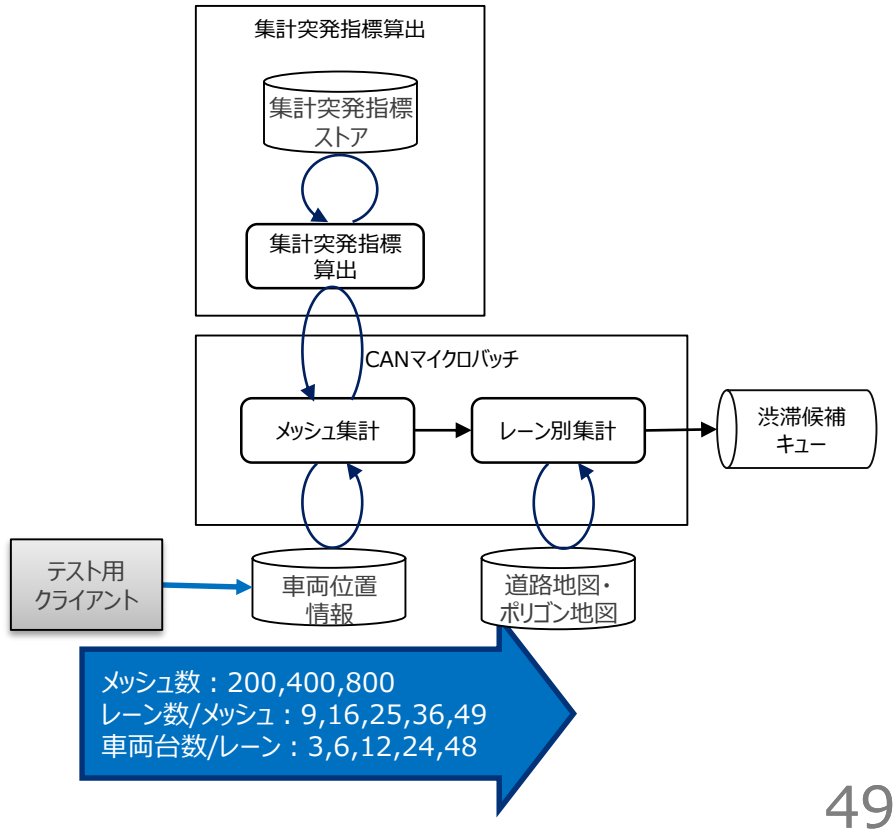
②メッシュ集計 + レーン集計を総量の5割 (メッシュ集計のみの絞り込み想定)

③メッシュ集計 + レーン集計を総量の2割 (メッシュ集計+集計突発指標の絞り込みを想定)



検証環境

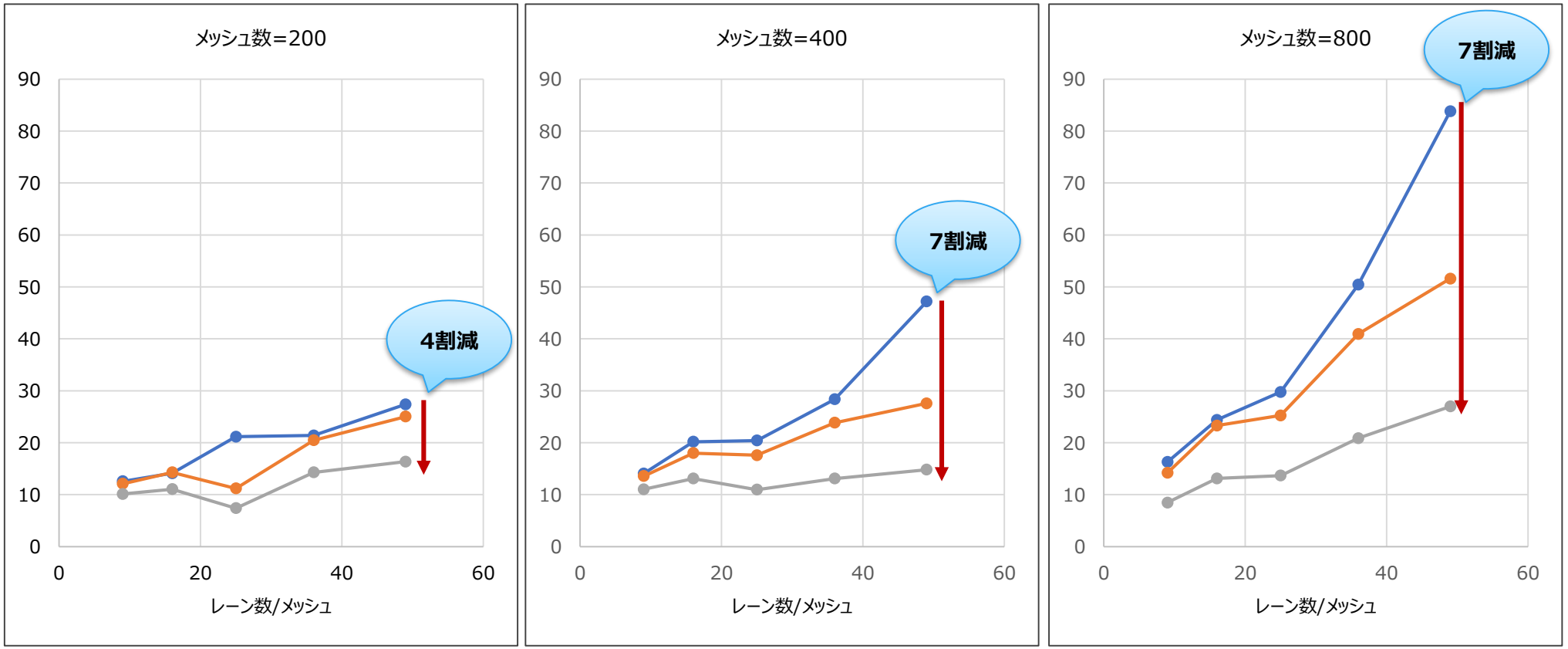
メッシュ／レーン内に存在する車両位置情報を変化させ、集計時間を測定する。



検証２．結果

メッシュ集計により事前にレーン集計対象を絞り込むことで、処理時間を減らすことができることがわかった。
レーン数や車両数が増加すると、メッシュ集計とレーン集計の時間の差が大きくなるため、絞り込みの効果がより顕著になると考えられる。

メッシュ数を変化させた際の処理時間の試算(s)



まとめ

TOYOTA NTTグループ

検証内容・観点	まとめ・考察	課題
1. CANストリーム処理	基盤性能検証で得られた結果より試算した結果、CANストリーム処理は、車両台数3,000万台に拡張可能である。 • 物理値変換処理 デバイスシャドウへの書き込みあり/なしでの性能比較を行い、書き込みなしの場合は24万TPSが性能限界であることを確認した。 物理値変換処理にデバイスシャドウ格納処理を追加することで処理時間が1.3～1.5倍増加する。 • 車両位置情報の格納 車両情報の格納処理について、Geohash方式と時空間集計方式で性能比較を行い、Geohash方式は50.4万台、時空間集計方式は68.4万台まで処理できることを確認した。 時空間集計方式を利用することで、 Geohash方式の1.36倍まで格納性能が向上した。	• サーバ故障等の異常時に対応するためのアーキテクチャ改善 • スパイク対応可能なアーキテクチャへの改善 • 利用頻度の少ないデータは単価の安い保存領域に移行する等、経済合理的なデータ管理のアーキテクチャ検討
2. CANマイクロバッチ処理	絞り込み効果を確認するために、基礎性能検証で得られた結果を用いて、以下の条件下で試算を行った。 ※基礎性能検証結果に関しては、本資料では省略している。 ① レーン集計のみを実施（絞り込みなし） ② メッシュ集計 + レーン集計を総量の5割 (メッシュ集計のみの絞り込み想定) ③ メッシュ集計 + レーン集計を総量の2割 (メッシュ集計+集計突発指標での絞り込みを想定) ③のように前段に絞り込み処理を入れることでレーン集計対象が少なくなり、全てのレーン集計を行う①に比べて、最大7割程度処理時間を減らすことができることがわかった。 レーン数や車両数が増加するとメッシュ集計とレーン集計の時間の差が大きくなるため、絞り込み効果がより顕著になると考えられる。	• 集計突発指標算出処理の並列処理化によるスループット・TATの向上 • I/F改善によるスループット・TATの向上 • 集計突発指標テーブルを分散データストアへ外出しすることによるスケーラビリティ向上 • 学習を用いた集計突発指標テーブルの更新

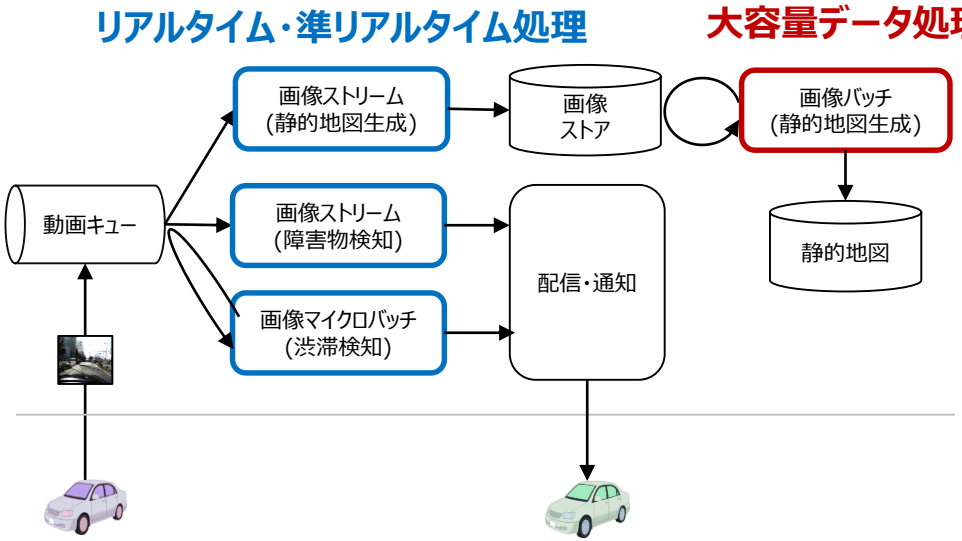
画像データ処理

検証概要

車両から送信される動画に対する基盤側の処理について、処理方式ごと（ストリーム処理、マイクロバッチ処理、バッチ処理）に性能測定を行う。

検証概要

車両から収集する動画データは大容量であるが、性能要件に応じて処理方式を決定する必要がある。
本検証では、ストリーム処理・マイクロバッチ処理・バッチ処理において、性能測定を行う。



検証内容・観点

1. 画像ストリーム処理
   a. 車両から送信された動画をストリーム処理し、リアルタイムに処理できる最大負荷を測定する。なお、システム全体のボトルネックがGPU処理であるため、稼働状況に応じて、別拠点のGPUノードに処理をオフロードする。

   b. 渋滞車列・原因特定APへの入力データを変化させたときの基礎性能を検証する。

2. 画像バッチ処理
   画像ストアに蓄積された大量の画像に対するバッチ処理を行い、GPU処理における限界性能を検証する。

検証１-a.　内容

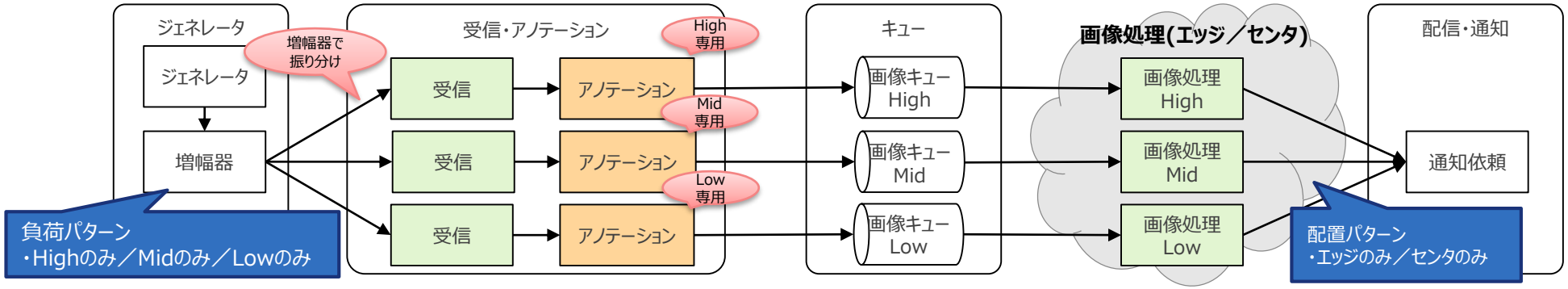
車両から送信された動画をストリーム処理し、リアルタイム性を担保できる限界点を検証する。

なお、システム全体のボトルネックがGPU処理であるため、稼働状況に応じて、別拠点のGPUノードに処理をオフロードする。オフロード有無による処理時間影響も検証する。

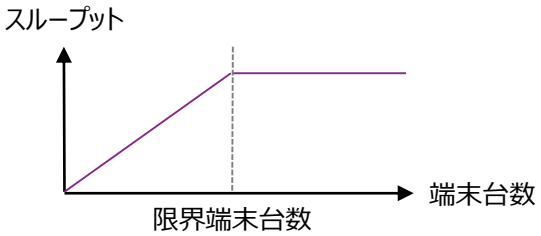
検証内容・検証環境

処理ノード動的選択技術*1を有効にした状態で、画像電文により負荷を投入する。前段の動的優先度付け処理へのインプット条件を変更させることにより優先度を変更(＝処理される拠点を変更)し、拠点毎の処理スループット①および電文処理時間②を計測する。

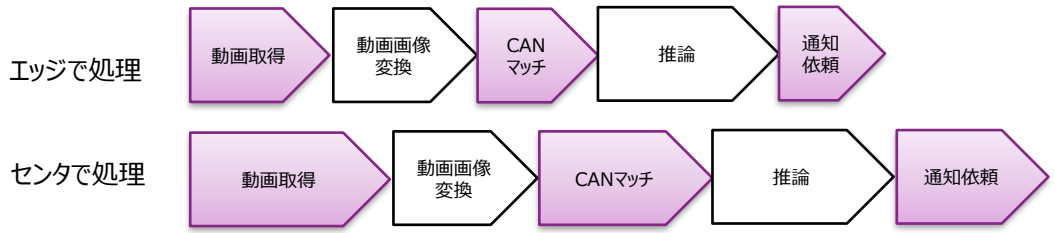
処理場所ごとの性能が適切に測定できるよう、優先度の割当ルールや負荷のパターンを調整して検証を行い、スループット・処理時間がどのように変化するか見極める。



①画像処理のスループットを測定する



②各処理の時間を分析する

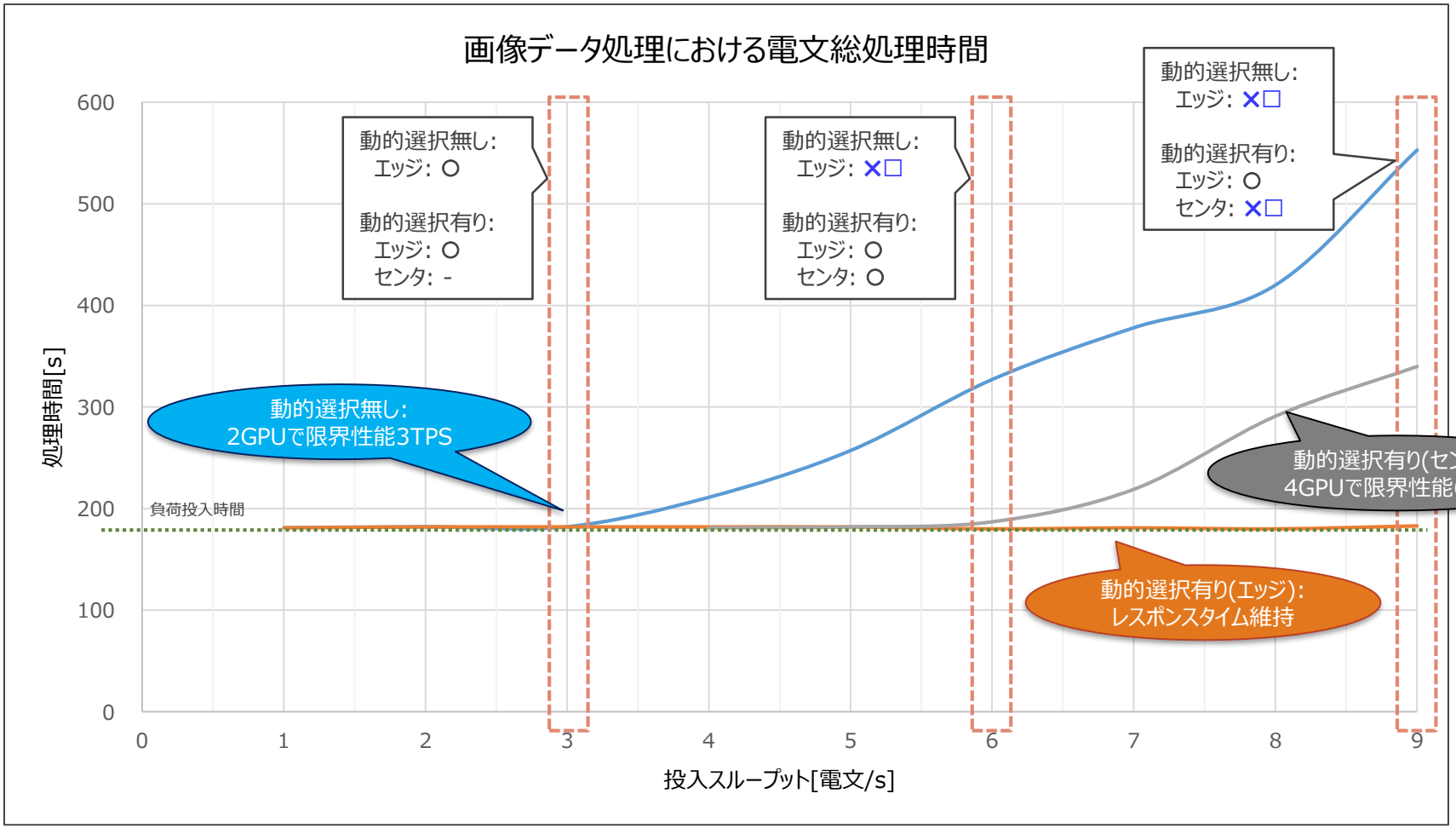


*1:詳細は、「Appendix④ 垂直分散コンピューティング技術」を参照

検証１-a. 結果(1/2)

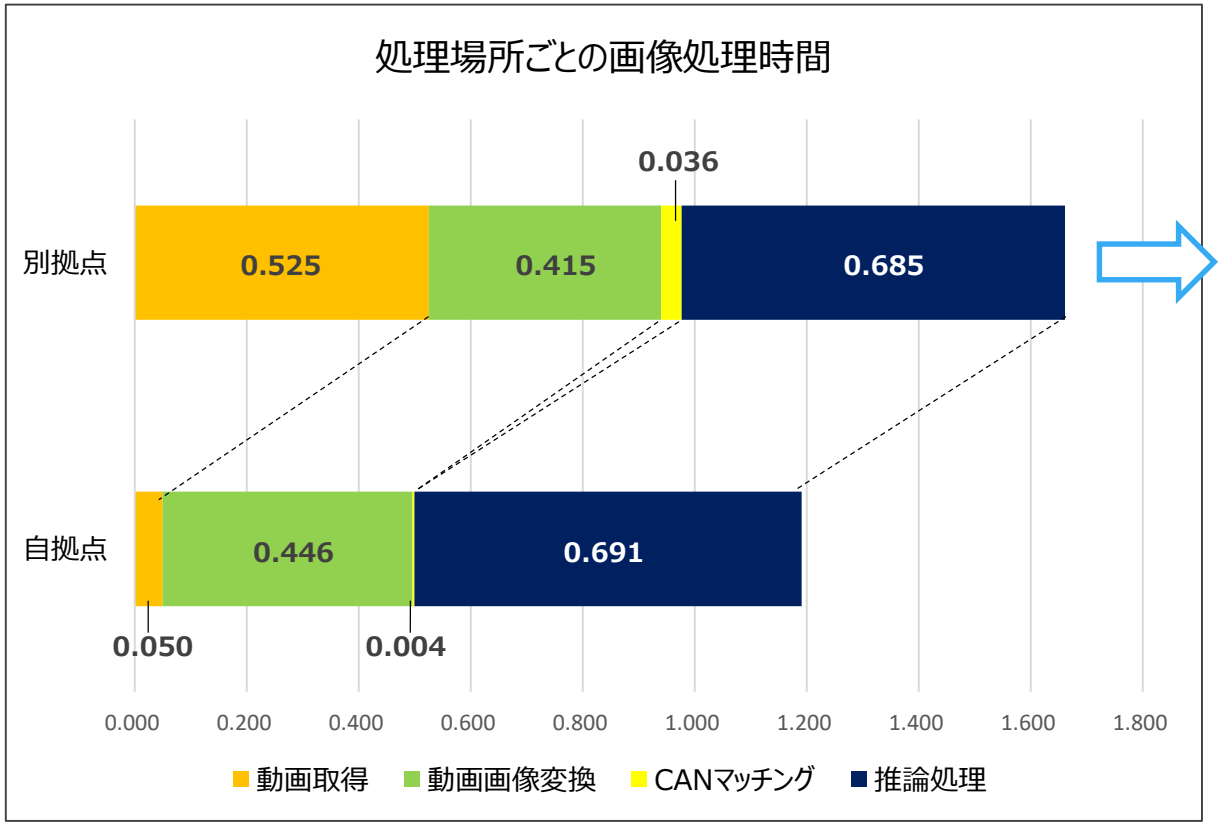
処理ノード動的選択技術の有無による性能比較を行った。

処理ノード動的選択技術を用いることで、自拠点で行う優先度が高い処理に関しては負荷量を一定にできる分、レスポンスタイムを維持できており、 負荷状況に関わらず一定時間で処理できることがわかった。また、別拠点に画像データ処理をオフロードすることで処理を担当するGPUノード数が増加するため、限界性能値が向上することを確認した。



検証 1 -a.　結果(2/2)

自拠点および別拠点で画像処理を行った際の処理時間を分析した結果、動画取得時間以外の処理は影響がないことから、別拠点での処理時にはデータ転送分の時間のみが加算されることになる。



別拠点配置による処理時間への影響

- 増加傾向：
  - 動画取得
  - CANマッチング(データサイズが小さいため転送時間は微増)
- 変化なし：
  - 動画画像変換
  - 推論処理

※処理時間に用いる値は複数回試行した結果の平均値とする。

検証１-b.　内容

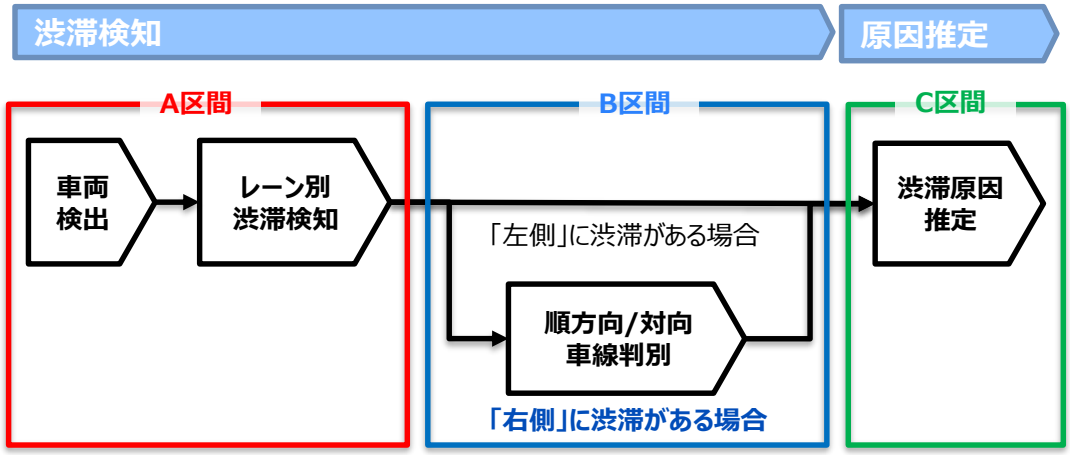
渋滞車列・原因特定APへの入力データを変化させたときの基礎性能を検証する。

検証内容・検証環境

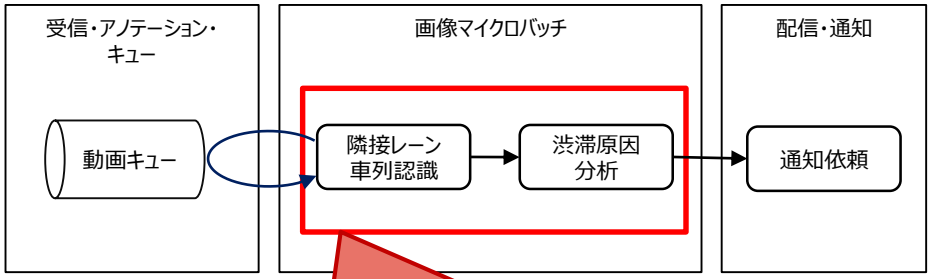
渋滞車列・原因特定技術について、以下の条件を変えた入力データに対する処理時間及びスループットを測定する。

i.　動画長
ii.　フレームレート
iii.　ビットレート（解像度）

画像処理実施中は3つの区間でGPU使用傾向がある。
処理時間については、それぞれA区間、B区間、C区間と定義し、区間ごとに分析する。



入力データのバリエーション



入力データバリエーションによる基本処理性能の確認

i.　動画長：10s、30s、60s
ii.　フレームレート：10fps、30fps、60fps
iii.　ビットレート：
- 1920x1080(1080p)
- 3840×2160(2160p)
- 7680×4320(4320p)

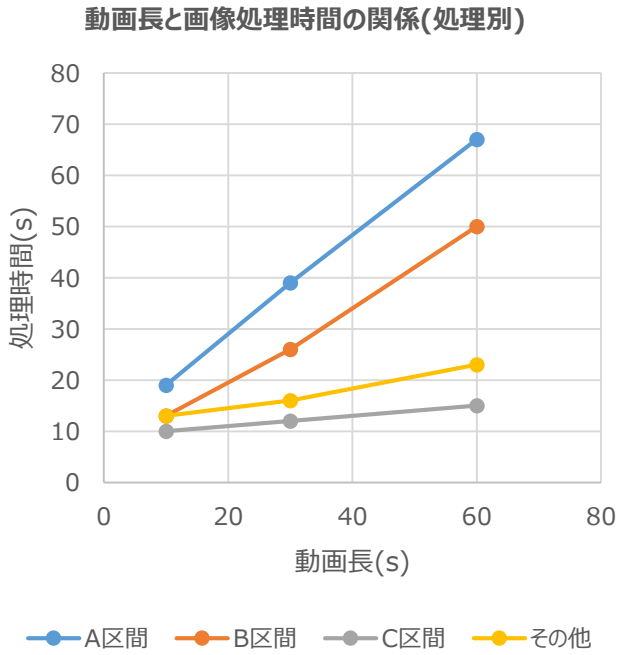
検証 1 -b. 結果

TOYOTA NTTグループ

入力データバリエーションと処理時間の関連性は以下であることがわかった。

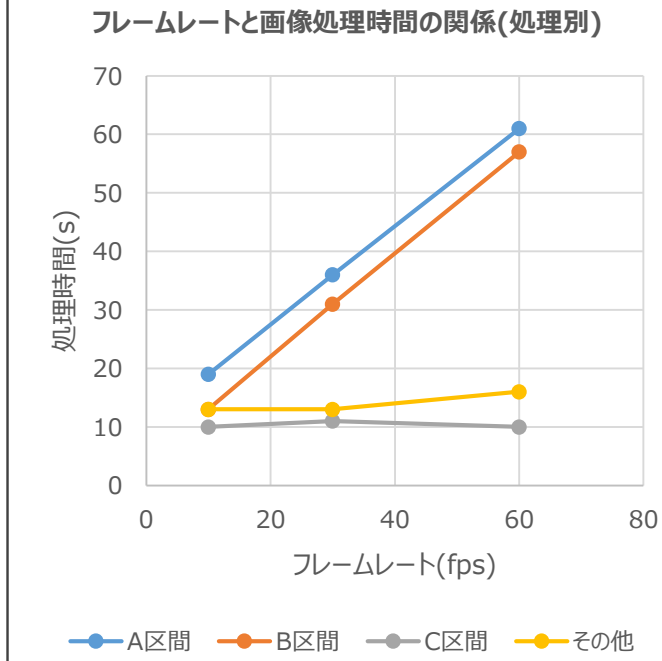
i. 動画長：A区間、B区間、C区間とも、動画長に比例して線形増加する
ii. フレームレート：A区間、B区間はフレームレートに比例して線形増加、C区間は一定である
iii. ビットレート：A区間、B区間はフレームレートに比例して線形増加、C区間は一定である

i. 動画長



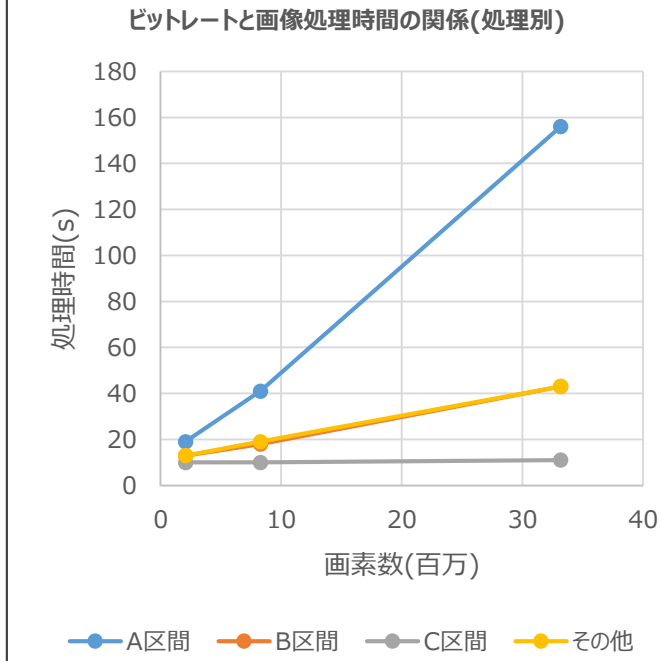
#	動画長	A区間	B区間	C区間	その他	全体
1	10.1	19	13	10	13	55
2	30.2	39	26	12	16	93
3	60.6	67	50	15	23	155

ii. フレームレート



#	フレームレート(fps)	A区間	B区間	C区間	その他	全体
1	10	19	13	10	13	55
2	30	36	31	11	13	91
3	60	61	57	10	16	144

iii. ビットレート



#	解像度	画素数(百万)	A区間	B区間	C区間	その他
1	1080p	2.07	19	13	10	13
2	2160p	8.29	41	18	10	19
3	4320p	33.17	156	43	11	43

※時間の単位はいずれも秒

検証２．内容

画像ストアに蓄積された大量の画像に対するバッチ処理を行い、GPU処理における限界性能を検証する。

検証内容・検証環境

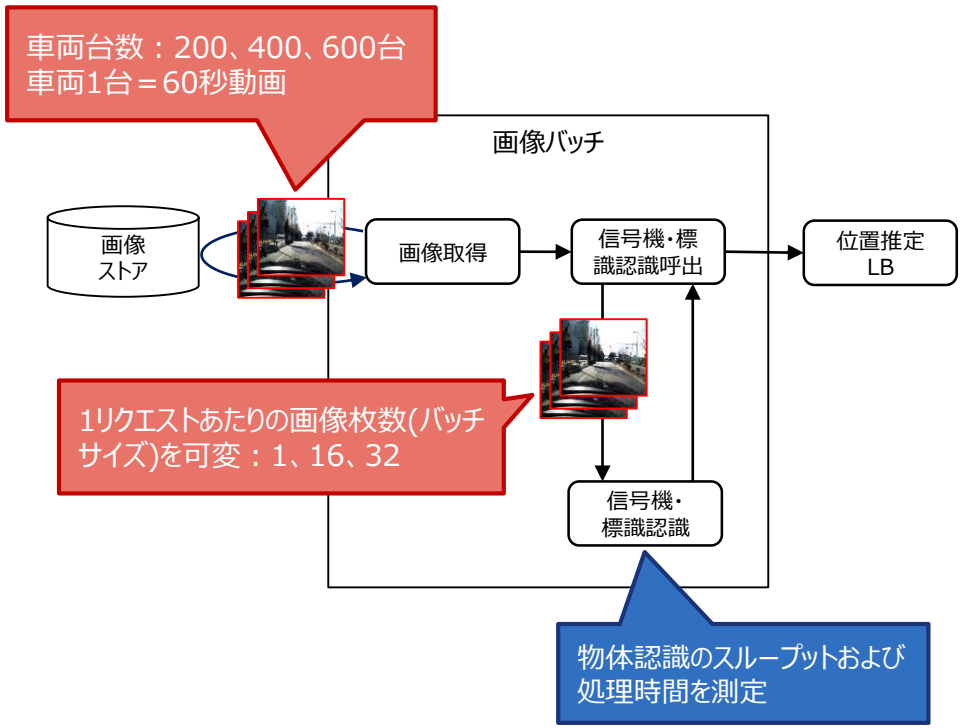
画像ストアに蓄積された大量の画像に対するバッチ処理において、GPU処理の性能限界値を取得する。以下のパラメータを可変とし、物体認識のスループットおよび処理時間を測定する。

① 1回の推論リクエストで処理する画像枚数（バッチサイズ）
② 画像データサイズ

画像データの種類

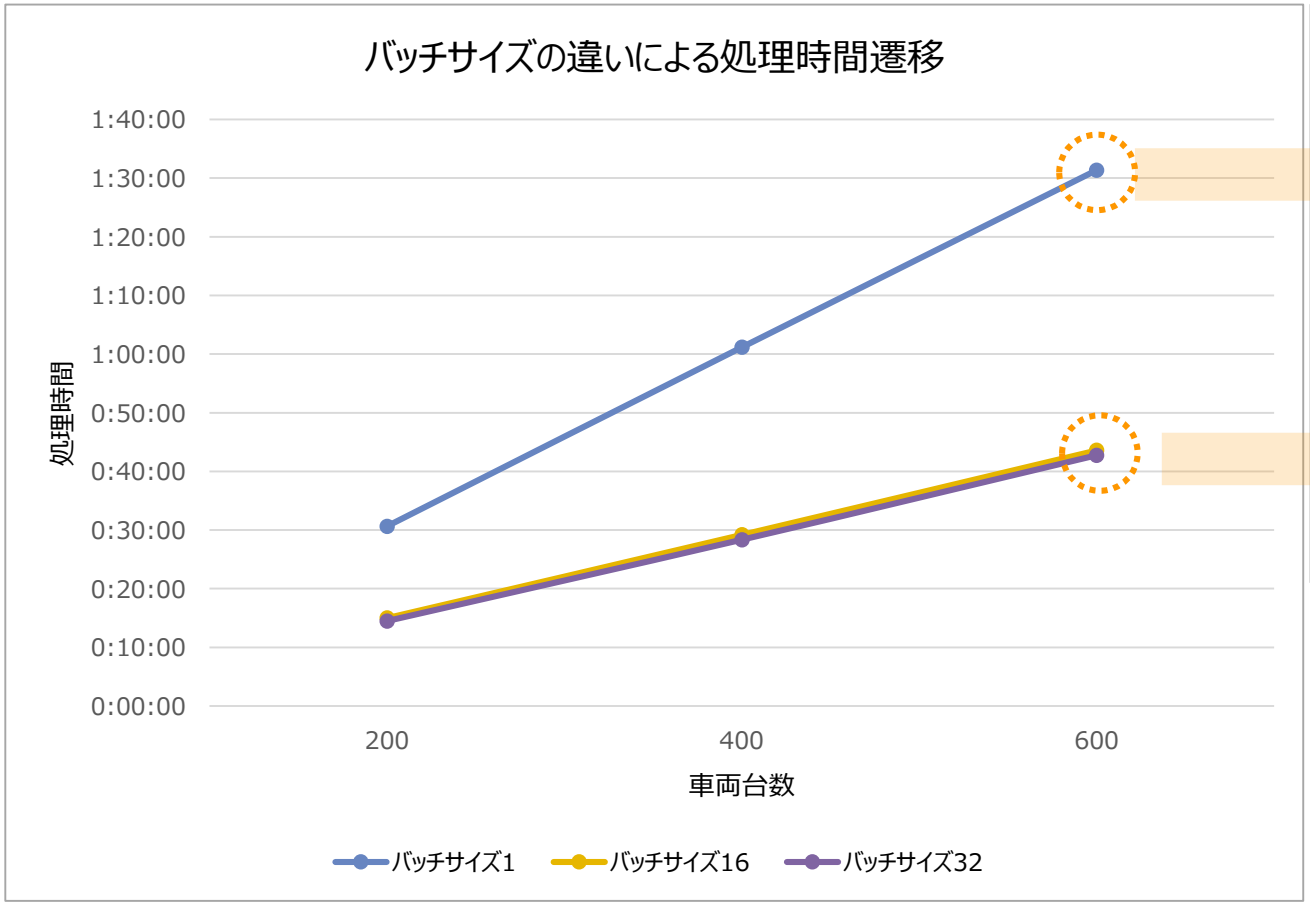
#	フレームレート	ビットレート	解像度	画像サイズ（平均）
1	10fps	6200kbps	高画質	295KB
2	10fps	6200kbps	低画質	77KB
3	30fps	6200kbps	高画質	295KB
4	10fps	4800kbps	高画質	291KB
5	10fps	8000kbps	高画質	298KB

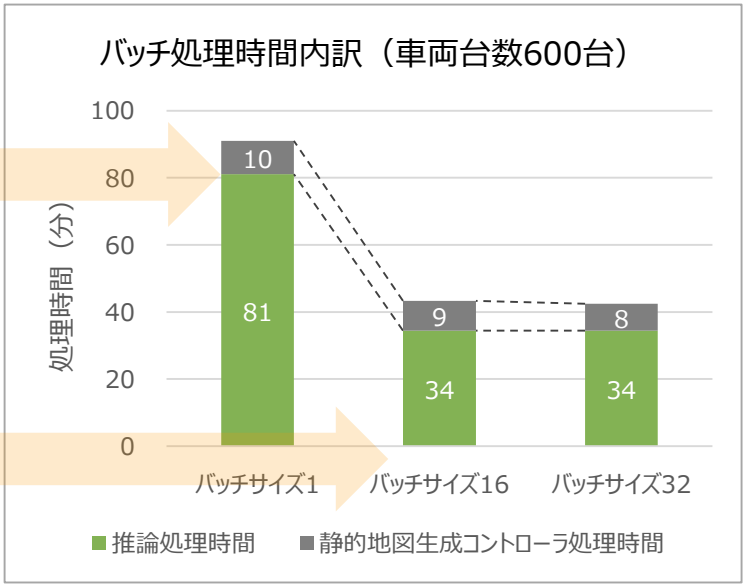
パラメータのバリエーション



検証２．結果(1/2)

バッチサイズを大きくすることで、推論処理時間が削減できることがわかった。また、車両台数とバッチ処理時間は比例増加することを確認した。なお、バッチサイズは32程度で頭打ちとなり、バッチサイズを40以上にした場合はGPUメモリ（16GB）の枯渇によりOOMエラーが発生した。





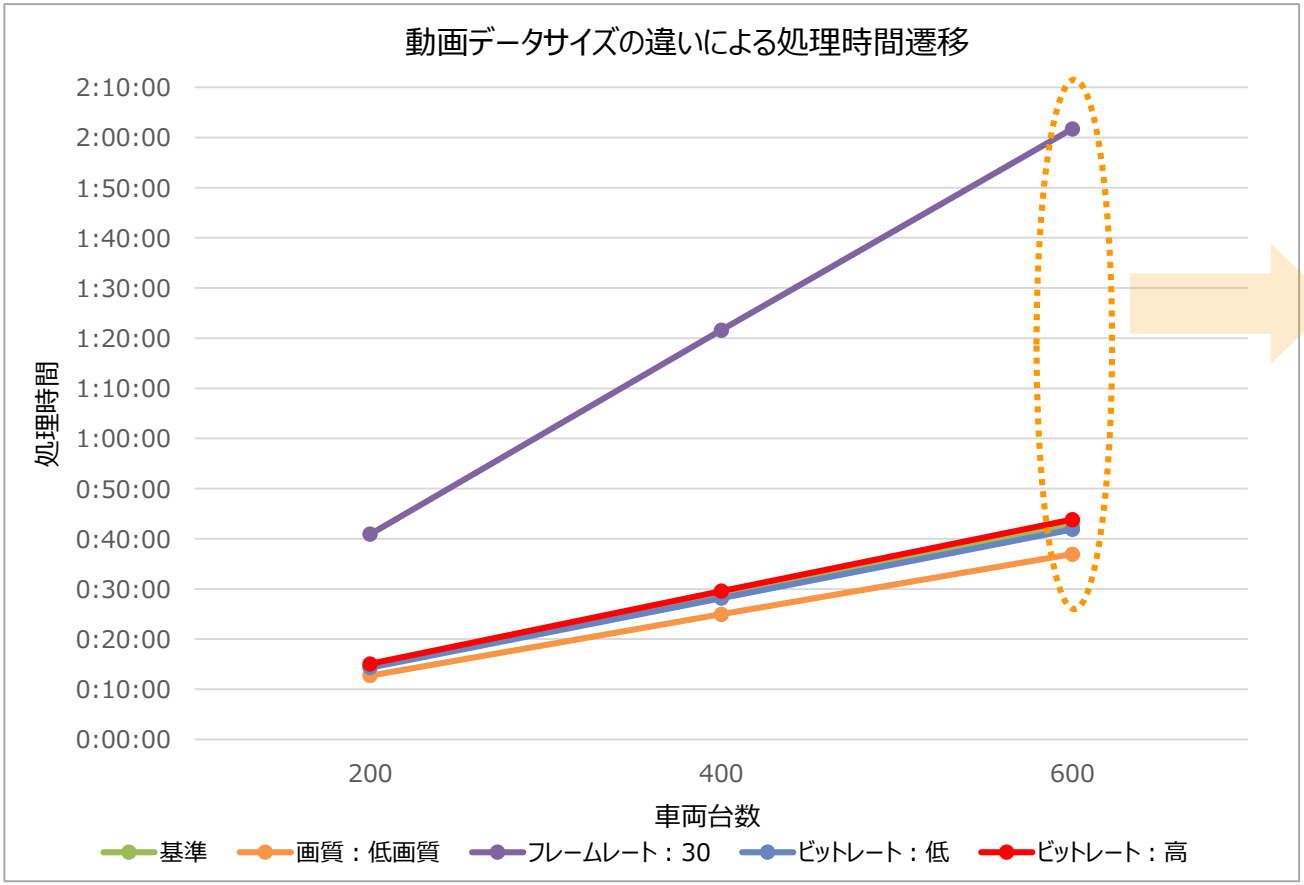
バッチサイズ	推論処理時間(平均)	画像1枚あたりの推論処理時間(平均)
1	53.9ms	53.9ms
16	363ms	22.69ms
32	705ms	22.03ms(※)

※ 1リクエストあたりで処理する画像が増えるため、バッチサイズが大きくなると推論処理時間も長くなる

検証２．結果(2/2)

動画データのバリエーションにおける性能傾向は以下であることがわかった。

- 解像度：低画質の画像サイズは高画質のおよそ0.26倍にもかかわらず、処理時間は0.86倍程度にとどまった（画質は後続の位置推定処理にて精度をあげるための1要素であるため、処理時間に影響が少ないのであれば高画質するのが望ましい）
- フレームレート：フレームレートが大きくなると推論処理対象の画像枚数が増加するため、想定通りバッチ処理時間が延びた
- ビットレート：バッチ処理時間にほぼ影響を与えていない。これはフレームを切り出すときに画質を固定しているため、ビットレートによるデータサイズに差が発生していないためであると考えられる



基準動画データと各可変パラメータの処理時間倍率
（車両台数600台）

解像度	処理時間(分)	倍率
高画質(基準)	43	-
低画質(0.26倍)	37	0.86

フレームレート	処理時間(分)	倍率
10fps(基準)	43	-
30fps(3倍)	122	3

ビットレート	処理時間(分)	倍率
6200kbps(基準)	43	-
4800kbps(0.77倍)	42	0.98
8000kbps(1.3倍)	44	1.02

まとめ

TOYOTA NTTグループ

検証内容・観点	まとめ・考察	課題
1. 画像ストリーム処理	a. 従来は1エッジの限界が系全体の性能限界であったが、垂直分散コンピューティング技術を活用することにより以下の傾向が得られることを確認した。 • 自拠点で行う優先度が高い処理に関しては負荷量を一定にできる分、レスポンスタイムを維持できており、 負荷状況に関わらず一定時間で処理できる。 • 別拠点に画像データ処理をオフロードすることで処理を担当するGPUノード数が増加するため、限界性能値が向上する。(限界性能=1.5×GPUノード数)	以下の要素も組み合わせることで、スループット向上をさらに加速させることが可能となる • 推論処理の車両へのオフロードによるトラフィック量削減 • クラウド連携によるGPUリソース増強 • アップロードする車両を選択することによる、動画送信・処理コストの削減
	b. 渋滞車列・原因特定APへの入力データのバリエーションによる基礎性能検証を行い、得られた結果を以下にまとめる。 i. 動画長 – 渋滞検知：処理時間は動画長に対し、線形増加する – 原因推定：動画長とは依存性がなく一定である(※) ※検証においては、同じ動画を繰り返したものをINPUTにしたことから、動画長に比例して渋滞数が等倍に増えるため時間が延びた ii. フレームレート – 渋滞検知：処理時間はフレームレートに対し、線形増加する – 原因推定：ビットレートとは依存性がなく一定である iii. ビットレート – 渋滞検知：処理時間は動画長に対し、線形増加する – 原因推定：ビットレートとは依存性がなく一定である	• 前処理における2段階のディスク書き込み排除によるスループット・TATの向上 • 前処理との機能統合によるチャンク処理によるスループット・TATの向上 • CANデータの欠損・解像度不足への基盤側での補完対応
2. 画像バッチ処理	• バッチサイズ可変 バッチサイズを大きくすることで、スループットが向上することが確認できた。バッチサイズを上げることでGPUの効率的な利用ができるためである。なお、バッチサイズを上げすぎるとGPUがボトルネックとなり、スループットは頭打ちになる。 • 画像データサイズ可変 フレームレートが大きくなることにより、同様の倍率で処理時間が延びることを確認できた。 また、画質をある程度高い状態にしても、処理時間に大きな差異は発生しなかったため、位置推定精度向上などの面から高画質での処理は現実的である。	–

配信・通知

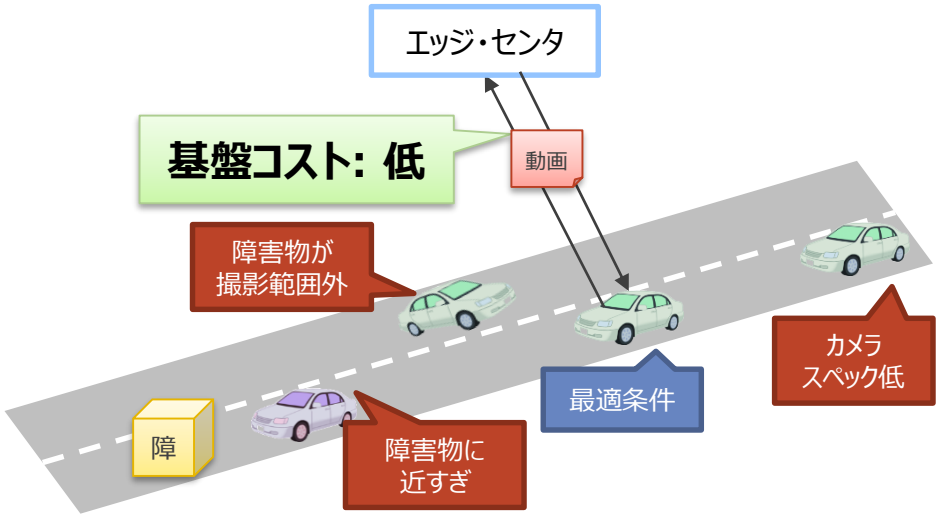
検証概要

障害物検知ユースケースのために動画データの収集依頼を通知することを想定する。その際の収集トラフィック削減のために車両と障害物の位置関係等の情報を用いて収集依頼の通知対象を絞り込む。この絞り込みの処理性能を見極める。

検証概要

本検証では、車両選択アルゴリズム*1を導入したアーキテクチャにおいて、性能検証を行う。

検証イメージ



検証内容・観点

1. 車両選択の処理時間
   検索対象となる車両データ数が多い状況を再現し、「収容する車両台数」と「車両選択の処理時間」の関係を検証する。

*1:配信・通知によるデータ収集時に、位置情報だけではなく、車両の向き、カメラ画角等を踏まえ最適な車両からのみ画像を収集することで、負荷集中に起因する基盤コストや処理時間増加の低減を期待できる。詳細は「 Appendix② 車両データ選択的収集アルゴリズム」を参照。

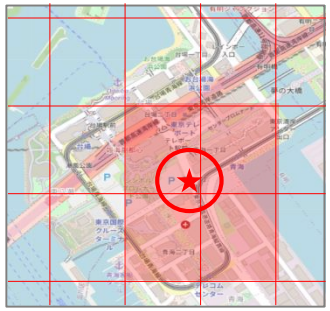
検証１．内容

検索対象となる車両データ数が多い状況を再現し、「収容する車両台数」と「車両選択の処理時間」の関係を検証する。

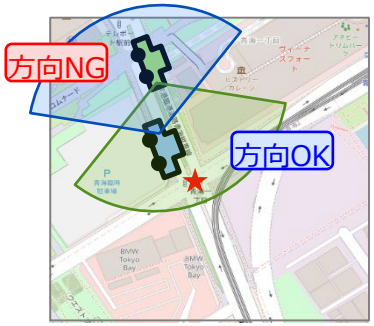
検証内容

車両選択アルゴリズムを用いた画像収集フローは以下の4ステップから成り立つ。それぞれの処理時間を測定し、「収容する車両台数」と「車両選択の処理時間」の関係を検証する。

①車両位置DBの検索



③方向の判定

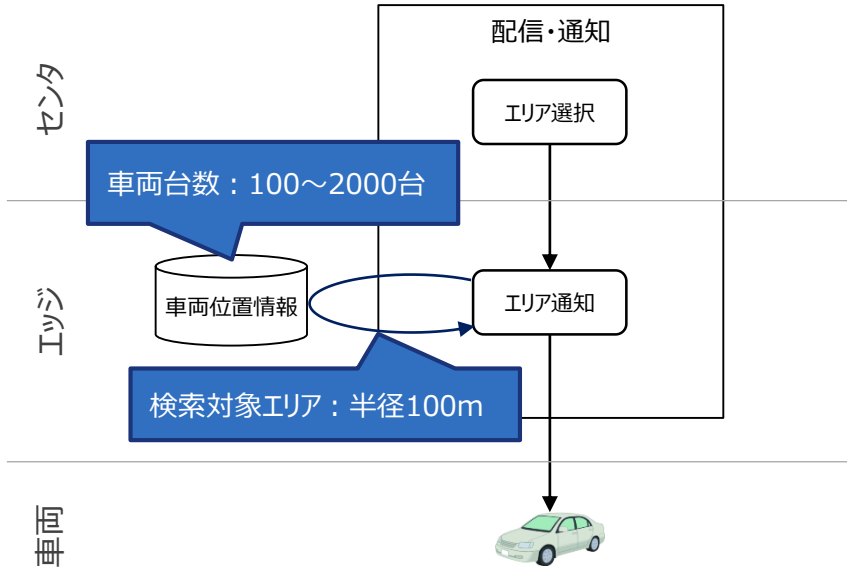


②距離の判定

④遮蔽の判定

検証環境

検索対象エリア内に存在する車両数を変化させながら処理時間の測定を行う。



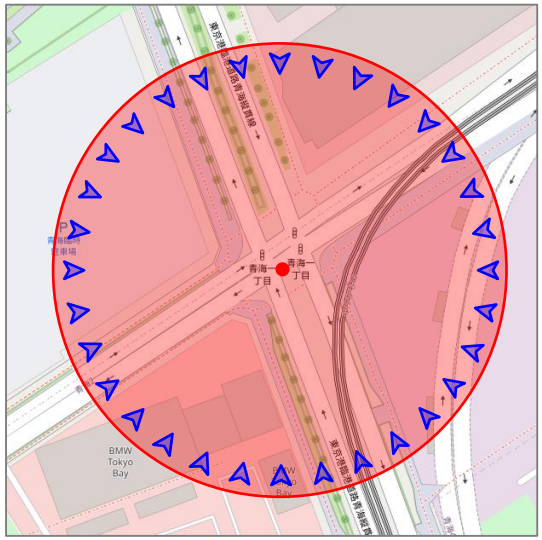
【上限を2000台とした根拠】

- 半径100mの円の面積は、100×100×3.14＝31,400m^2
- 駐車場のグリッドは車両1台あたり、3×5.5＝16.5m^2
- 円内に車両を敷き詰めると、31,400÷16.5＝1,903台

検証１．結果

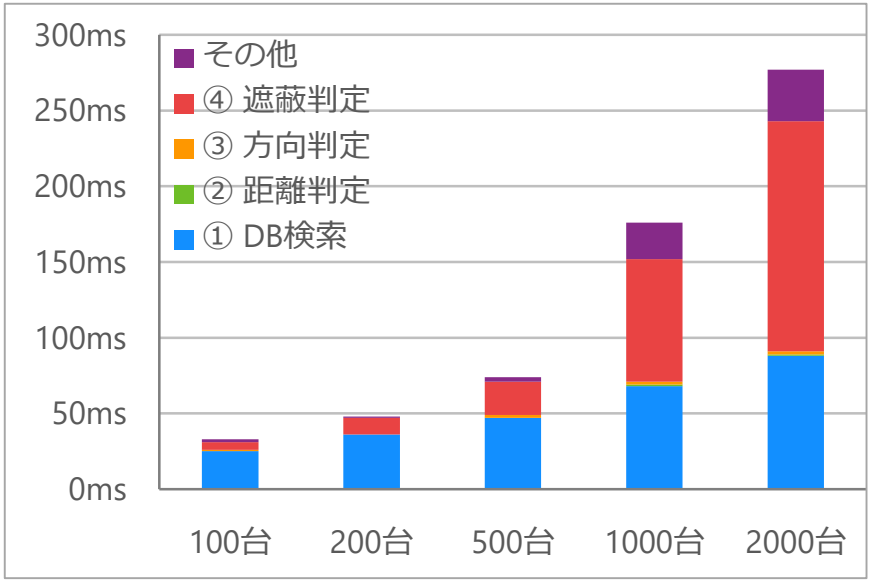
車両選択対象となる車両を増加させて検証したところ、DB検索および遮蔽判定で増加傾向が見られた。
特に遮蔽判定に関しては、対象車両が1000台を越えると支配項になる結果となった。

検証車両配置



- 検索エリア ○
  - 中心：「青海一丁目」交差点
  - 半径：100m
- 車両データ ⮝
  - 100～2000台（半径90m地点）
  - 距離OK・方向OK・遮蔽OK

車両数と処理時間の関係



- 「距離判定」および「方向判定」の処理時間はごくわずか
- 「DB検索」および「遮蔽判定」で増加傾向
- 特に遮蔽判定が必要な車両数により、処理時間に大きく影響を与える
  - 車両2000台で280ms程度
  - 1000台超で遮蔽判定が支配項となる
  - 理論値のO(n)に従う

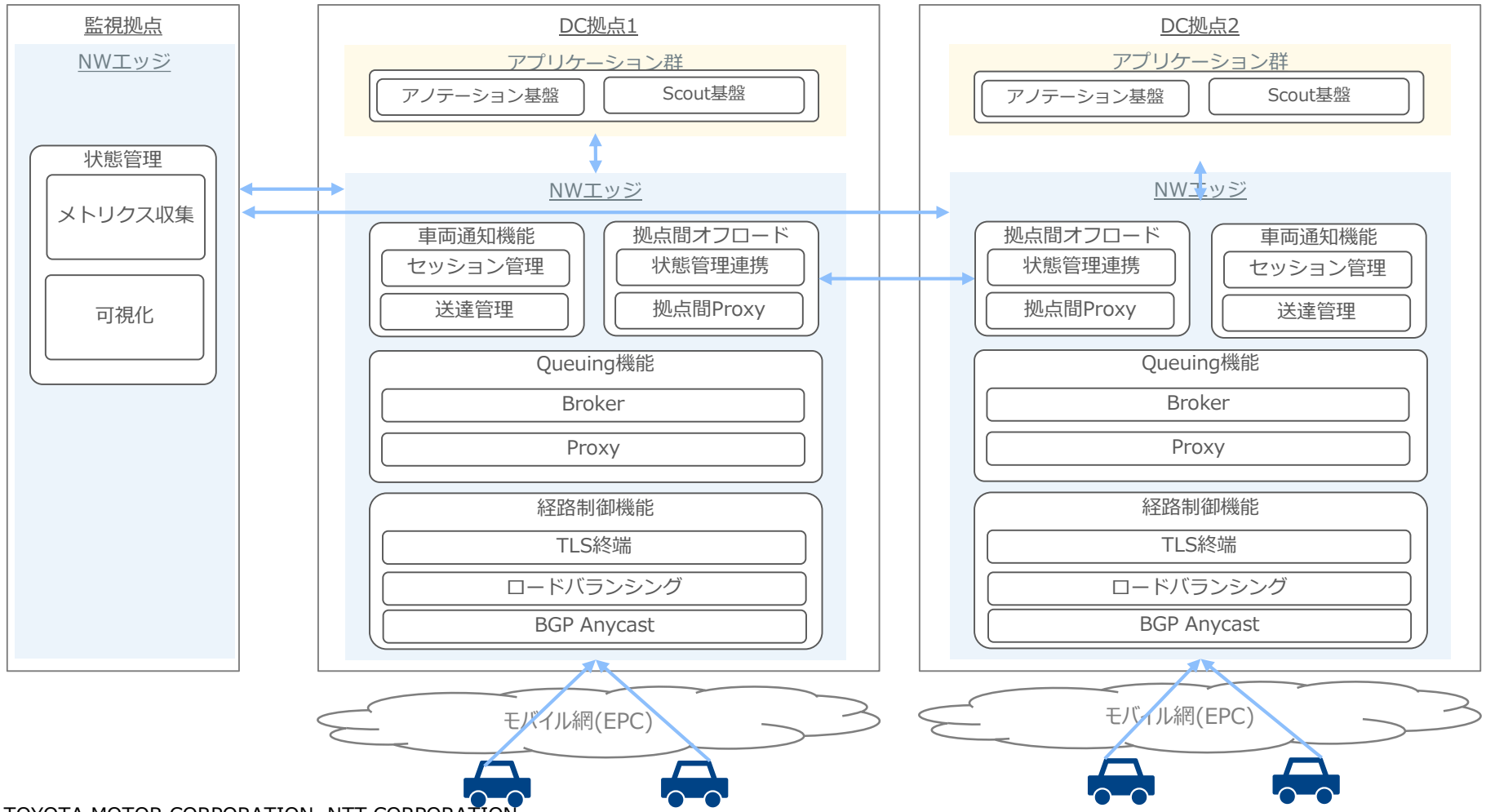
まとめ

検証内容・観点	まとめ	考察
1. 車両選択の処理時間	車両位置情報の収容台数を変化させ、車両選択アルゴリズムの各ステップの処理時間を測定した結果、以下の傾向が得られた。 • 「距離判定」と「方向判定」の処理時間はごくわずか • 「DB検索」および「遮蔽判定」で増加傾向 • 特に「遮蔽判定」に関しては1000台を越えると支配項となり、2000台収容時で280ms程度	車両選択アルゴリズムの適切な適用方針について考察を示す。 • 「距離判定」および「方向判定」は、処理時間も短く、最低限の無駄な通知を省く観点から、高速レスポンスを優先するユースケースや、通知のみで完了するユースケースに適用すべきである。 • 「遮蔽判定」については、処理の性質上、通知精度を優先する、または、通知に紐づき画像等の大容量データを収集するようなユースケースに適用すべきである。高速レスポンスを優先するユースケースで使用する際は、処理時間への影響とのトレードオフを考慮し適用を検討する必要がある。

NWエッジ基盤検証

検証概要(1/4)

拠点が広域に分散するコネクティッドカーシステム内で、CANや動画データの最適なデータフローを実現させるため、車両とアプリケーション群の間に、複数のネットワーク機能を持たせた基盤システムを含むアーキテクチャを策定し、コネクティッドカーシステムの技術課題への有効性を実験にて検証した。この基盤システムはネットワークエッジ（NWエッジ）と呼び、同じくエッジ上に配置するアプリケーション群とは区別する。

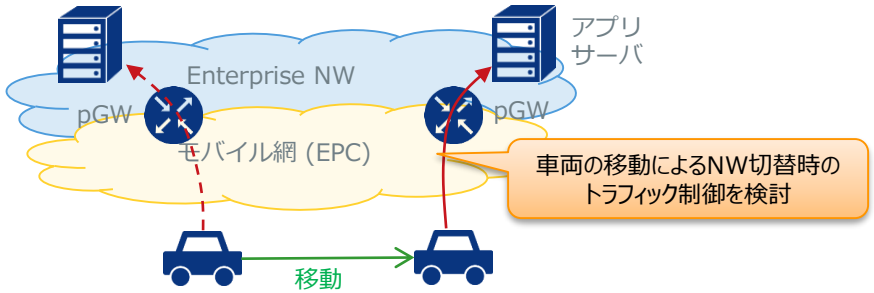


検証概要(2/4)

車両から複数拠点に分散するアプリケーション群へのデータフローとして、車両の移動に伴う接続アプリケーション設備の変更と、広域切替（局所的な負荷集中等による切替）に伴う接続アプリケーション設備の変更、を基礎ユースケースとして定めた。

車両の移動

車両移動に合わせて、接続先アプリケーション設備の切替を行う。車両の移動の基礎UCに関連して、NWエッジでの対応が求められる内容を応用ユースケースとし定めた。



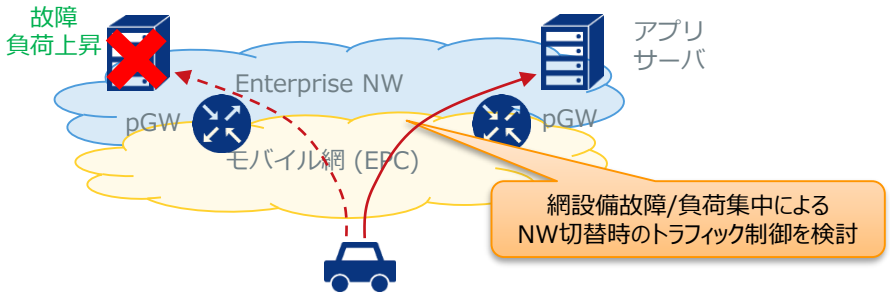
基礎UC	応用UC
車両の移動	移動追従を実現するためエッジ基盤からエッジ基盤への接続変更をサポートする
	高速に移動する車両のロケーション追従を実現しながらも高い応答速度や手動の系切替をサポートする

検証概要(3/4)

TOYOTA NTTグループ

広域切替

拠点での障害発生や負荷分散をトリガーとし接続先アプリケーション設備の切替を行う。広域切替の基礎UCに関連して、NWエッジでの対応が求められる内容を応用ユースケースとし定めた。



基礎UC	応用UC
広域切替	ネットワークインフラへの負荷削減のためエッジ基盤を活用することでデータ量の削減や事前処理を行う
	車両の局所集中が発生した際には車両からの重複/類似データの送信を制限し基盤への影響を防止する
	設備の負荷状況、時間、回線品質をもとにエッジ基盤の選択を行う
	ネットワークで輻輳を検知した場合、一部車両を別のエッジサーバへ切り替える
	各設備は増設に対応し、手動の増設だけでなく、高負荷時には自動的なリソーススケーリングを行う

検証概要(4/4)

基礎ユースケースを実現するために、複数のNWエッジ拠点が存在する環境を実装し、車両から各NWエッジ拠点までの最適な経路制御方式やNWエッジ拠点間の連携方式の検証を実施した。また、車両との通信におけるセキュリティの観点でTLSを用いた暗号化通信を想定し、NWエッジ拠点におけるTLS性能検証を実施した。

検証内容・観点	概要	基礎UC
1．車両からDC拠点、アプリケーションサーバへの最適な経路制御	経路制御の実現手段として2つの方式を検討し、車両の移動、広域切替(設備故障)による挙動と、それぞれの基本機能と性能を確認した。 ・DNS方式: 最寄りのDNSサーバまではBGP Anycastで誘導され、DNSサーバが最適なアプリケーション群(アノテーションサーバ基盤)を選択し車両に通知する。 ・LB方式: 最寄りのLBサーバまではBGP Anycastで誘導され、LBにて車両からのリクエストを最適なアノテーションサーバへ転送する。	車両の移動 広域切替
2．メトリクスを活用した経路制御	経路制御の手法としてLB方式を採用したNWエッジにて、下記を確認した。 ・疑似DCM、エッジやセンタのアプリケーションを結合し、シナリオに沿った処理完了の確認 ・基礎性能値取得 ・複数台構成によるスケールアウト対応可否確認 ・NWエッジ、アプリケーションサーバ故障時の挙動確認 ・ステータスに加えアプリケーションレベルでのメトリクスを収集及び活用した経路制御及び負荷分散	広域切替
3．TLS終端の性能検証(CPU処理)		-
4．Message Queue(MQ)を使ったエッジ拠点間データフロー	NWエッジ拠点が複数分散する環境下において、MQを用いた拠点間でのデータ連携方式を検討し、動作検証と性能検証を実施した。 ・広域分散MQを介した車両とアプリケーション群の通信の動作確認 （拠点間データ連携を含む） ・車両からのデータ送信における性能検証 ・車両へのメッセージ通知における性能検証	車両の移動 広域切替
5．TLSオフロード検証	CPU 使用率低減を目的として2つの方式を用いたTLSオフロードの挙動と、それぞれの性能を確認した。 ・TLS Accelerator: OpenSSL EVP API を利用した処理形態となる。 ・SmartNIC (ASIC): Kernel TLS を利用した処理形態となる。	-
6．5Gフィールド検証	LTE／5Gモデムを搭載した車両と基盤間において、大容量データのアップロード／ダウンロード通信を行い、その際の性能値を測定した。	-

検証１．内容

以下の検証内容にて、DNS方式とLB方式での経路制御の挙動確認及び基礎性能測定を行う。

検証内容

基礎ユースケース(車両の移動、広域切替)を想定した、DNS方式、LB方式による経路制御の挙動確認および、パラメータによる切替時間の比較を行う。また、 DNS方式、LB方式での基礎性能を測定する。

車両の移動：
①DNS/LB両方式での経路制御動作
②パラメータの違いによる切替時間への影響

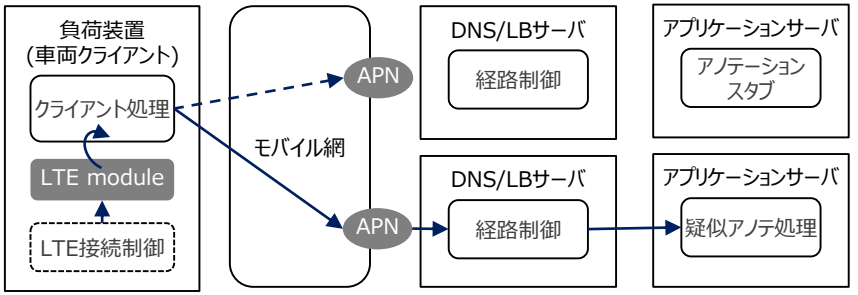
広域切替：
①障害パターンによる両方式での制御動作
-アプリケーションサーバ全故障
-アプリケーションサーバ一部故障
-制御装置故障
②パラメータの違いによる切替時間への影響

検証環境

車両の移動：
車両クライアントにて接続先のAPNの切替を行い、BGP Anycastによる経路制御の動作を確認する。

パラメータ：

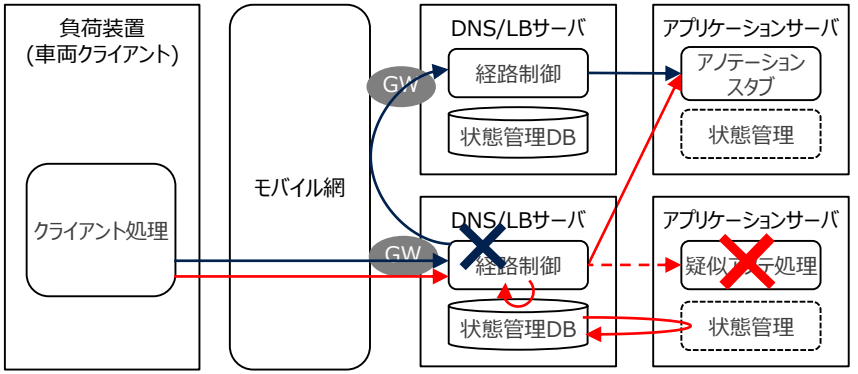
HTTP	1.1,2.0
TLS	有,無
セッション継続性	有,無
データ量(KB/s)	16,256
TTL	0,60



広域切替：
疑似的に設備障害を発生させ、状態管理DBと連携した経路制御の動作を確認する。

障害シナリオ：

パターン	想定動作
APサーバ全故障	経路制御により別拠点に切替る
APサーバ一部故障	経路制御による同拠点別APサーバに切替る
経路制御障害	BGP Anycastにより別拠点に切替る



検証１．結果

TOYOTA NTTグループ

基礎ユースケース毎の検証結果、 DNS方式、LB方式の基礎性能を以下に示す。

車両の移動

①DNS/LBの両方式での経路制御動作
両方式にて想定通りの動作となることを確認した。切替時間は両方式約2秒程度となり、経路制御方式による切替時間への影響がないことが分かった。

② DNS/LBの両方式でのパラメータの違いによる切替時間への影響
HTTPバージョン、TLS、セッション継続性、送信データによる切替時間への影響はないことが分かった。
応答時間には、各種パラメータによる影響があり、特にセッション継続性の有無による影響が大きいことが確認された。

セッション継続性	応答時間（TAT）
有	100～150ms
無	50ms～400ms

広域切替

①障害パターンによるDNS/LB両方式での制御動作
両方式にて想定通りの動作となることを確認した。
特に「故障検知閾値(ポーリング間隔やタイムアウト値)」が切替時間に大きく影響することが分かった。

処理時間	DNS	LB
切替処理	10～12秒	4~5秒
タイムアウト値	10秒	4秒

②パラメータの違いによる切替時間への影響
HTTPバージョン、TLS、セッション継続性、送信データによる切替時間への影響はないことが分かった。

基礎性能

DNS方式	LB方式
・安定稼働上限は約10,000クエリ/sであった ・UDP受信処理に伴う特定CPUの負荷がボトルネックであった思われる ・大量の負荷が途切れなく到着する場合、50,000クエリ/s程度処理可能だが現実には適用できないと想定する	・安定稼働上限は約10,000リクエスト/sであった ・TLS処理によるCPU負荷がボトルネックであったと思われる ・スケールアップ、チューニング方式は今後の課題である

検証２・３．内容

メトリクス情報に基づいた経路制御機能の検証、エッジ拠点の基礎性能検証をした。

検証内容

LB方式を用い、サーバ障害時のメトリクス情報収集、メトリクス情報による経路制御の挙動確認を行う。

基礎性能検証：
単一エッジ拠点に対し接続負荷をかけ、処理可能な接続数を確認する。
本検証では負荷装置を使用し負荷をかける。

メトリクス情報による経路制御検証：
基礎性能検証で測定した接続負荷の状態で以下障害時の経路制御動作の確認および処理性能を確認する。

①エッジノード障害
同拠点内での別ノードへの処理の分散、分散時の処理性能を確認する。
※エッジノード障害は以下の3パターン
・LBサーバ故障（電源断）
・LBサーバのロードバランシング機能障害
・LBサーバのAnycastルータとのBGP接続プロセス障害

②拠点障害
以下障害による複数拠点での処理分散を確認する。
-アプリケーションサーバの過負荷
-拠点全障害

検証環境

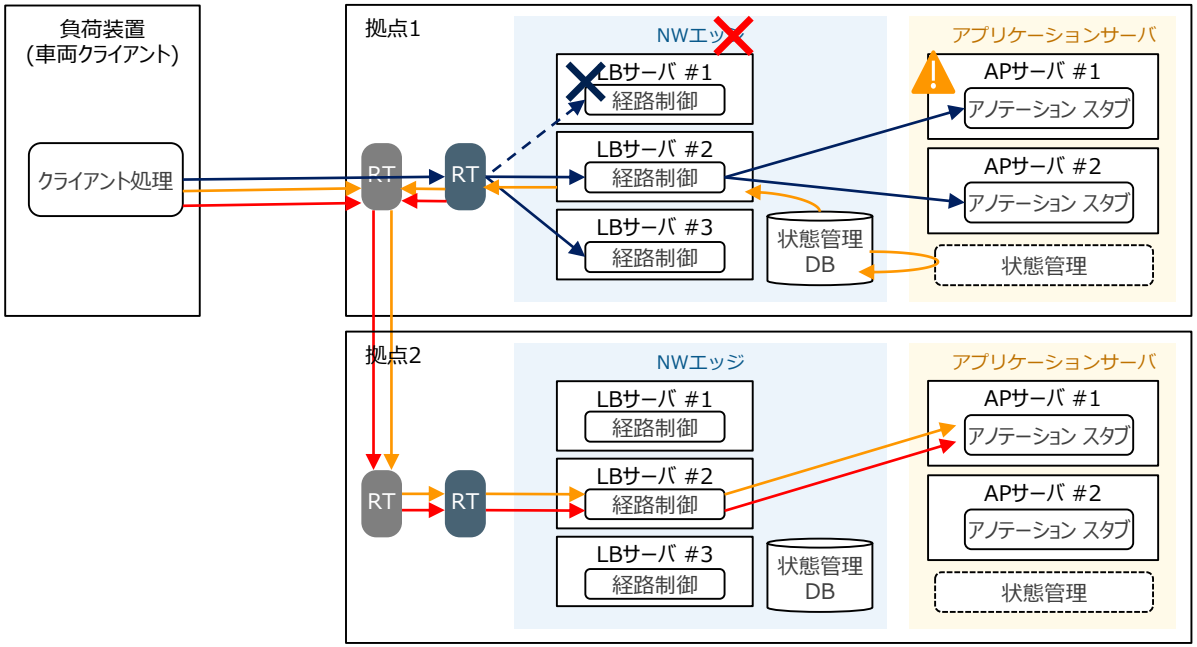
負荷条件：

同時接続数	20万TPS※
データサイズ	8KB
プロトコル	HTTPS(POST)

※単一拠点で想定する最大処理性能を同時接続数とし設定した。
基礎性能検証結果に従い、メトリクス情報による経路制御検証の同時接続数は15万TPSとする。

障害時の分散動作想定：

→	エッジノード障害	同拠点内の正常ノードにて処理が継続される
	拠点障害	別拠点へ処理が分散される
→	過負荷	一部トラフィックを別拠点に分散する
→	全断	全トラフィックを別拠点へ分散する



検証２・３．結果

基礎性能検証、ノード故障試験、複数拠点での経路制御/負荷分散における検証結果を示す。

基礎性能検証

・1拠点で、処理性能は同時接続数15万TPSとなることを確認した。
当初負荷とし20万TPSを想定したが、1つのTLSコネクションあたり、8回程度のHTTPリクエストを処理しており、負荷数が低下したものと考える。
・CPU使用率は、平均25%となっており、ボトルネックの要因とはならなかった。今回はセッション再開を使用し、新規TLSコネクション数が最大2万TPSで検証しており、新規コネクションの割合によってはCPU使用率がボトルネックとなり処理性能が低下することも考えられる。

メトリクス情報による経路制御

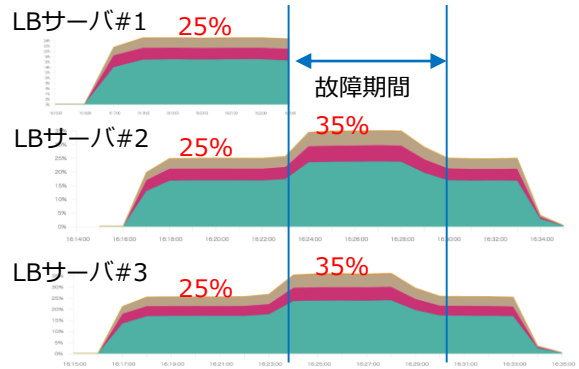
①エッジノード障害
・何れのノード障害においても、正常ノードに分散させ処理を継続できることを確認した。
ただし、NWエッジの電源断直後、数秒間処理が出来ない現象が観測された。

・平時CPU使用率が平均25%に対し、障害時(2台)になると35%であった。

・今回のHWスペックでは1GbpsのRx/TxをCPUのおよそ5%であった。

・LBサーバのプロセス障害では、クライアントからの再送が確認された。
車載器の負荷となる可能性がある点を今後考慮する必要がある。

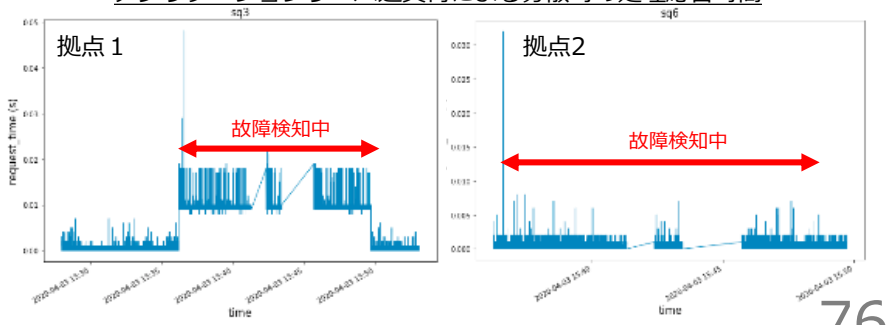
ノード障害時のCPU使用率の遷移



②拠点障害
・アプリケーションサーバの過負荷では、想定通り一部トラフィックが別拠点に振分けられることを確認した。負荷分散(故障検知)中の処理応答時間には、DC拠点間遅延(約10ms)が加算された。

・拠点障害については、想定通り全トラフィックが別拠点に切替ることを確認した。
切替直後は、処理応答時間に増加が見られた。

アプリケーションサーバ過負荷による分散時の処理応答時間



検証４．内容

アノテーション処理や車両通知等のアプリケーション群の挙動を含めて一連のシナリオが機能するか確認するための実車走行試験と、疑似的にデータアップロードまたはクライアント接続の負荷をかけて広域分散MQの性能を確認する基盤検証を行った。

検証内容

広域分散MQを導入したNWエッジを用いて、車両からのデータ送信、車両へのメッセージ通知の動作検証と基礎性能検証を行う。

動作検証：
広域分散MQを介して車両とアプリケーション群の通信が検証シナリオ通り可能であることを確認する。（実車及び実証実験基盤を使用）

基礎性能検証：
負荷試験装置を用いて車両からのデータ送信及び車両へのメッセージ通知を行う際の処理性能と拠点間オフロード時のデータ到達時間を測定する。

①車両からのデータ送信
-a 車両からみたNWエッジにおけるレスポンス性能及びリクエスト処理性能の測定
-b 拠点間オフロード時のLBサーバからアプリケーション群へのメッセージ到達時間の測定

②車両へのメッセージ通知
アプリケーション群から車両へのメッセージ到達時間の測定

検証環境

東京・大阪に広域分散MQを含むNWエッジサーバ群及びアプリケーション群を実装し検証を実施。

動作検証における検証シナリオ：
#１：東京・大阪の各拠点内設備での処理
#２：拠点障害等を想定したデータ送信・通知双方のオフロード
#３：車両移動等を想定した通知のみオフロード

※動作検証では九段下周辺を走行ルートとした

基礎性能検証における負荷条件：

①-a

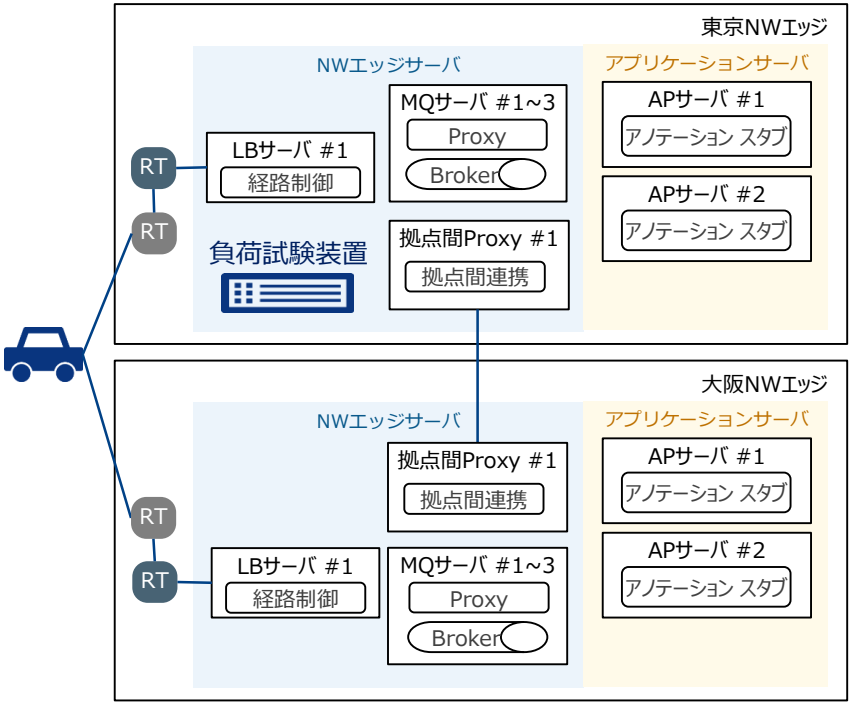
総リクエスト数	20万
クライアント数	200
メッセージサイズ	128B ~ 1MB
プロトコル	HTTP

①-b

瞬間リクエスト数	1 ~ 2000
メッセージサイズ	128B ~ 1MB
プロトコル	HTTP

②

瞬間メッセージ数	1 ~ 1000
メッセージサイズ	128B



検証４．結果

広域分散MQの動作確認、基礎性能の検証結果を示す。

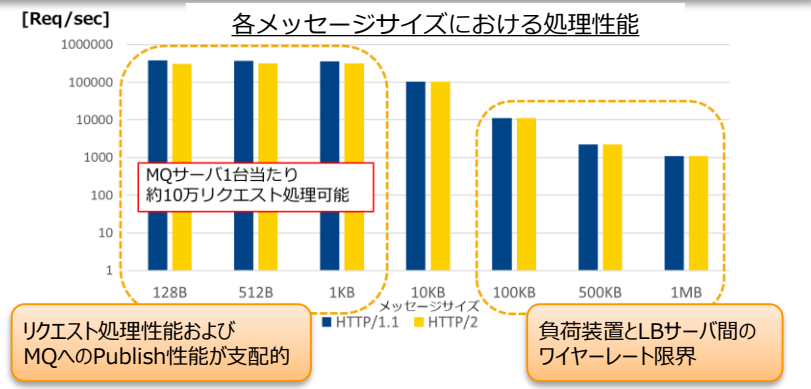
広域分散MQの動作確認

車両・東京/大阪エッジ拠点・センタ拠点の実験系において、
＃1～＃3すべての検証シナリオが正常に完了することを確認した。

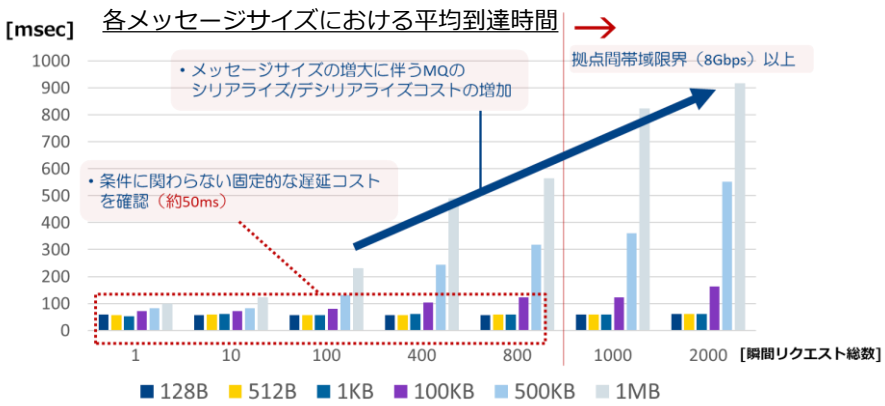
#	パターン	試験結果
1	東京・大阪の各拠点内設備での処理	OK
2	拠点障害等を想定したデータ送信・通知双方のオフロード	OK
3	車両移動等を想定した通知のみオフロード	OK

基礎性能検証

①車両からのデータ送信
①-a 車両からみたNWエッジにおけるレスポンス性能及びリクエスト処理性能の測定
・メッセージサイズが1KB以下の場合はMQサーバ1台あたり毎秒約10万リクエスト処理可能であることを確認した。
・HTTP/1.1とHTTP/2の結果に差分は観測されなかった。
・メッセージサイズが100KB以上になるとワイヤレート限界が支配的となった。



①-b 拠点間オフロード時のLBサーバからアプリケーション群へのメッセージ到達時間の測定
・オフロードによって最低50msの遅延がオーバーヘッドとして生じた。
・メッセージサイズが100KB以上の場合、瞬間リクエスト数の増大に伴い平均到達時間が総データ送信量に比較して顕著に増加した。
（ex. 100KBを1000リクエスト送る方が1MBを100リクエスト送るよりも平均到達時間は小さくなる）



②車両へのメッセージ通知
・拠点間論理キューで同時接続クライアント数を1000程度までスケールさせても通知メッセージの到達時間への影響は限定的であることを確認した。

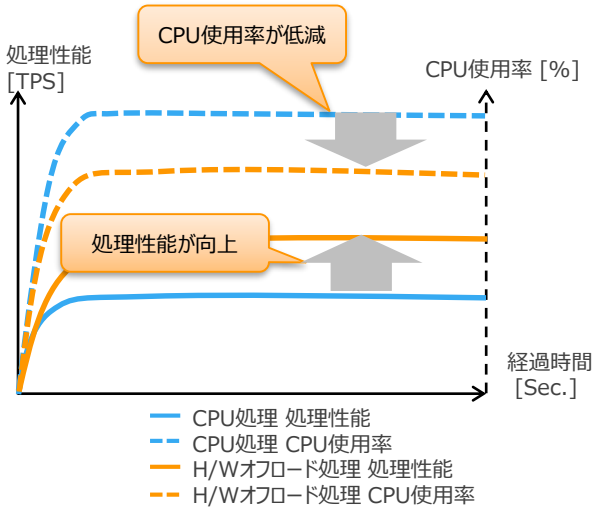
検証５．内容

検証１の基礎性能検証にも記述の通り、エッジ拠点への同時接続数が増大した場合、CPU処理がボトルネックとなり性能が低下することが考えられる。本検証では、H/Wオフロードによる処理性能向上に関する検証を行った。

検証内容

H/Wオフロード機構の導入より経路制御のどの程度の処理性能向上が実施できるか検証する。
CPU使用率についても確認する。

<u>期待する性能向上</u>
H/Wオフロードを利用することで、CPU使用率を下げながらも、処理性能が向上することを期待する。



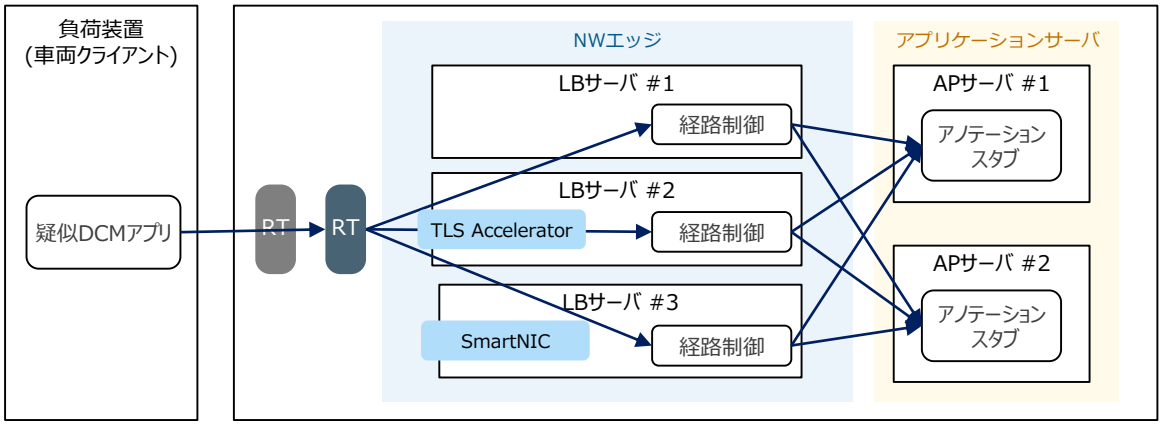
検証環境

負荷条件：　※22,000台分の車載クライアントのパケットを模擬した負荷シナリオを負荷装置にて設定

サーバ1台に対する最大同時接続数	22,000台 ※
データサイズ	8KB
APプロトコル	HTTPS(POST)
TLS	TLS：v1.2、Cipher: ECDHE-RSA-AES128-SHA256

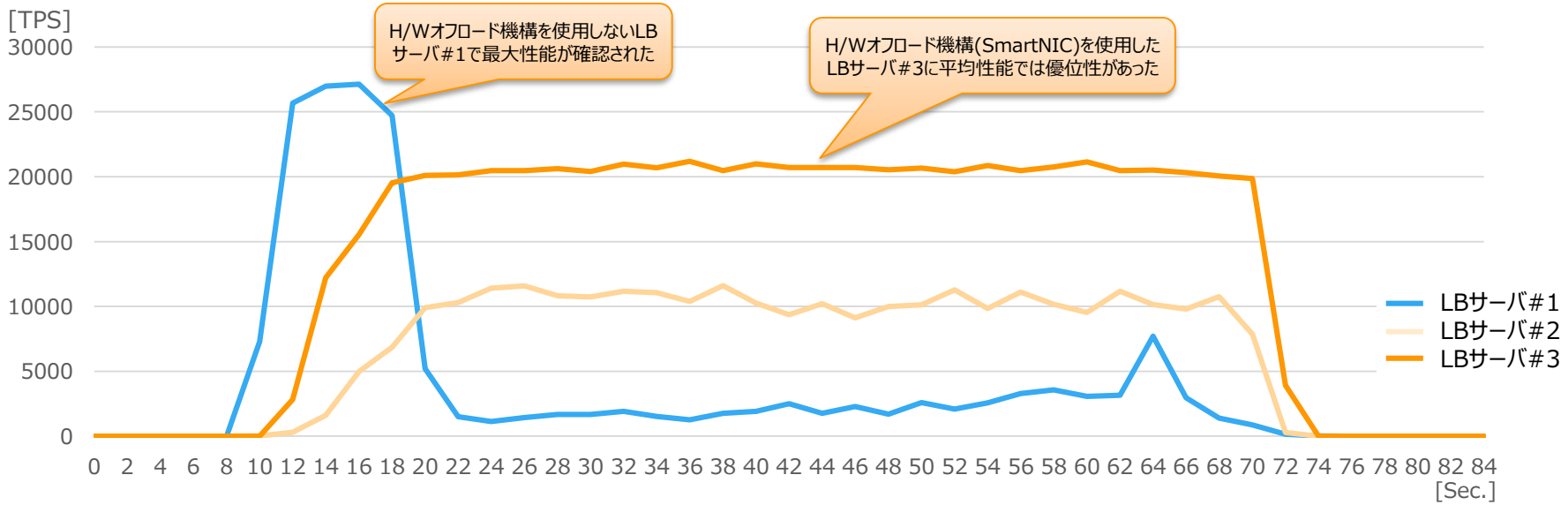
環境構成：

	H/Wオフロード機構	利用OpenSSL version
LBサーバ#1	なし	OpenSSL 1.1.1f
LBサーバ#2	TLS Accelerator 搭載	OpenSSL 1.1.1f
LBサーバ#3	SmartNIC (ASIC) 搭載	OpenSSL 3.0.0-alpha13



検証５．結果

各エッジサーバにおける処理性能とその時のサーバCPU使用率を測定した。H/Wオフロード機構を導入したLBサーバ#2,3方が安定した処理となることが確認できたが、最大性能はH/Wオフロード機構を導入していないLBサーバ#1と比較し大幅に落ちる結果となった。本検証ではデータ量の少ないパケットを条件としたため、H/Wオフロード機構が得意としない処理条件となったことにより、期待する性能向上を確認できなかったものと考える。



	LBサーバ #1	LBサーバ #2	LBサーバ #3
Avg. CPU [%]	75.87 / 11.15 ※	19.79	77.57
Avg. Transaction Rate [TPS]	5,977.57	9,152.61	**18,989.47**
Max. Transaction Rate [TPS]	**26,993**	11,599	21,182

LBサーバ#3 (オフロード機構:SmartNIC)ではTx Offloadのみで検証を行った。
Rx Offloadは完全にCPUで処理するモデルとなっており、利用することで、CPU使用率の低減も可能と想定する。

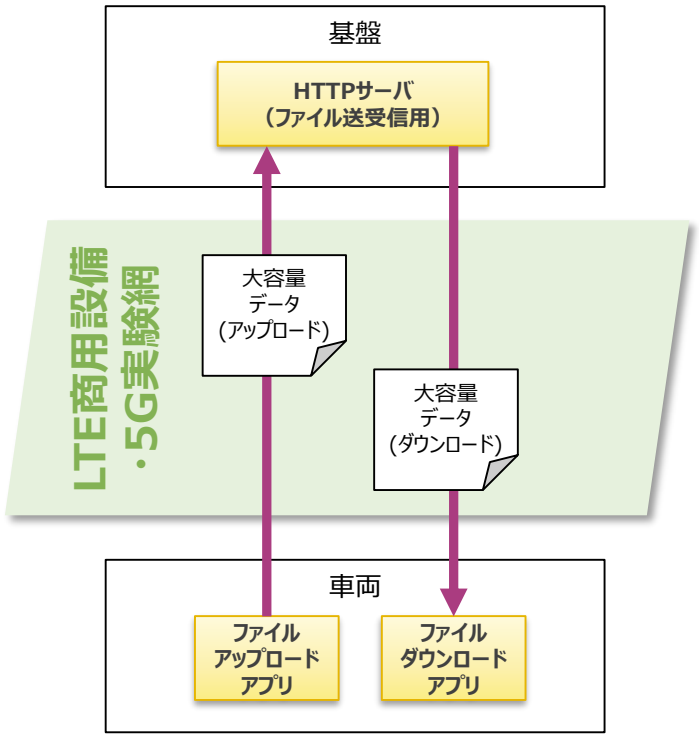
※ LBサーバ#1のCPU利用率は計測前半の山部分での平均値と、以降の平坦な部分での平均値で分けて算出している。
また、0~8s, 74s以降のTPSが0となっている区間は計算対象外としている。

検証６．内容

LTE／5Gモデムを搭載した車両と基盤間において、大容量データのアップロード／ダウンロード通信時の性能値を測定する。

検証内容

大容量通信用のコネクションを車両・基盤間で確立し、大容量ファイルのアップロード／ダウンロードを行う。それぞれHTTP POST／GETが完了するまでの処理時間およびスループットを測定する。



検証環境

無線環境

- 5G実験網
  - ➢ 実験設備であり、商用環境・商用設定とは異なる
  - ➢ 28Ghz帯 400Mhz幅
- LTE通信網
  - ➢ 商用設備であり、利用場所・時間・通信環境により実効速度は異なる

制約事項

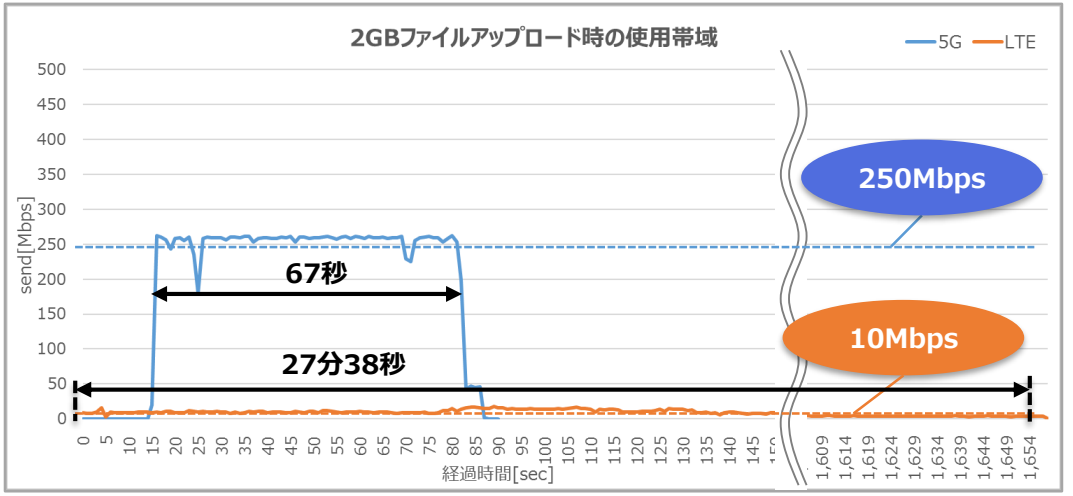
- 大容量データのファイルサイズは128MB~2GByteとする
- 車両から定常的にCANデータが送信されている状況下でアップロード／ダウンロードが行われることを想定する
- 走行エリアは、お台場の5Gトライアルサイトとし、LTEの検証も同じくお台場エリアで実施する
- 設備の制約上、通信ハンドオーバの検証は行わない

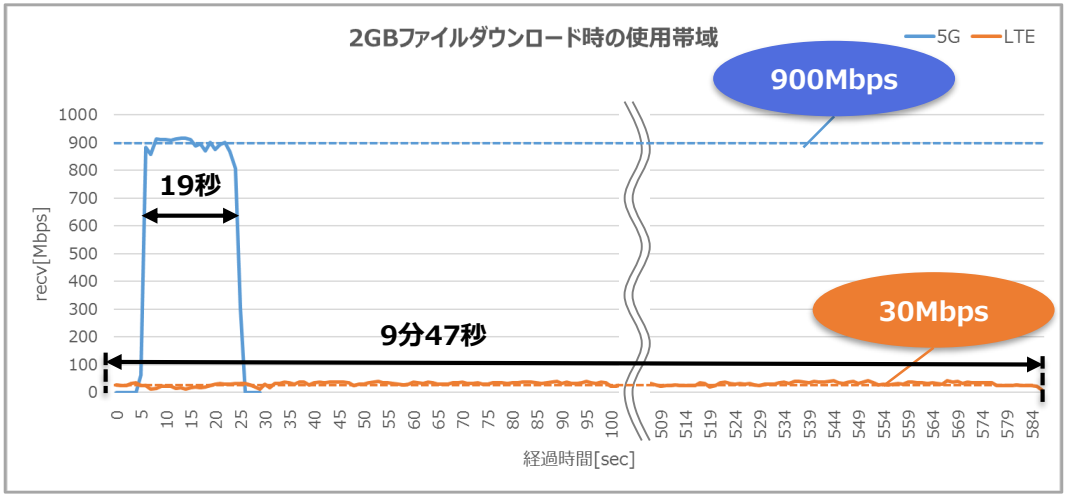
検証６．結果

車両と5G実験網を接続して、フィールド上の定点で、5G実験網の基礎性能値を取得した。その結果、広帯域の5G技術を用いることで、LTEに比べ、アップロード時間・ダウンロード時間とも大きく短縮できることを確認した。

2GB転送時の性能結果*1

	アップロード		ダウンロード	
	5G実験網	[参考] LTE商用網	5G実験網	[参考] LTE商用網
NWスループット	平均値：243.61Mbps 最大値：262.25Mbps	平均値：9.88Mbps 最大値：20.69Mbps	平均値：861.87Mbps 最大値：914.93Mbps	平均値：27.91Mbps 最大値：44.42Mbps
処理時間	平均値：67秒	平均値：27分38秒	平均値：19秒	平均値：9分47秒





*1：本実証実験の結果は実際の商用5G網とは異なる実験用に構築した環境での測定値である。

まとめ

車両とアプリケーション群の間に、複数の機能群（経路制御、プロトコル、Queuing、拠点間オフロード、車両通知管理、状態管理）を持つNWエッジを配置するアーキテクチャを実装し、実験を通して、その有効性を確認することが出来た。検証目的毎の結果と、明らかになった課題について、以下にまとめる。

検証内容・観点	まとめ・考察	課題
１．車両からDC拠点、アプリケーションサーバへの最適な経路制御	経路制御の2つの方式(DNS方式、LB方式)に対し、車両の移動、広域切替時の制御動作と、各方式の基礎性能を検証 ・何れの方式でも問題なく経路制御が実現できることを確認 ・DNS方式、LB方式で切替時間に大きな差はなかった ・CPUがボトルネックとなり性能が低下する可能性を確認した	スケールアップ、スケールアウト、チューニング方式について検討が必要と判明し、「２．メトリクスを活用した経路制御」の中で取り組んだ。
２．メトリクスを活用した経路制御 ３．TLS終端の性能検証(CPU処理)	過負荷および拠点障害時の経路制御(LB方式)による広域切替の動作と、エッジ拠点の処理性能を検証 ・単一拠点の処理性能は15万TPSとなることを確認した 本検証ではCPUボトルネックが要因の性能低下は見られなかった ・ノード障害では、何れの障害でも正常ノードでの処理が継続できた 切替時に一部再送が発生するケースがあった。 ・拠点障害(過負荷、故障)では、別拠点へ処理が振分けられた 振分け動作により、処理応答時間が増加することを確認した	メトリクス監視の高度化 • 本検証では、監視対象となるサーバのステータスやRound-Trip Time(RTT)のモニタリング状況を基に経路制御を行った。AIを使って蓄積データを分析し渋滞予測や渋滞検知時のナビゲーションを実現できる基礎技術の実環境での検証が必要となる。
４．Message Queue(MQ)を使ったエッジ拠点間データフロー	メッセージ送信・通知について、車両移動に伴う広域分散MQによる拠点間連携を含む動作と処理性能を検証 ・広域分散MQにて想定通り拠点間連携、メッセージ通知ができた ・メッセージ送信では、サーバ1台あたり約10万リクエストを処理できた アプリケーションへの到達には連携に伴う遅延(50ms) が発生した ・メッセージ通知では、到達時間への影響は限定的であった	アプリケーションサーバ群の優先制御との連携 • 主に車両の移動や設備故障といったユースケースに基づく経路制御について検証を実施した。今後はアプリケーションサーバ群で取り組んでいるデータ処理に基づく優先制御との連携を検討し、優先度の高いデータは高優先度のキューを選択する等のより高度な制御方式の検証が必要と考える。
５．TLSオフロード検証	H/Wオフロードによる CPU使用率低減と処理性能について検証 ・H/Wオフロードにより CPU使用率低減と性能向上が確認できたが、 最大性能ではH/Wオフロードの優位性が確認できなかった ・オフロード機構としては、SmartNICに性能優位性が確認できた	SmartNICのRx Offload対応とデータサイズの拡張 • 本検証では、SmartNIC試験時においてはTx Offloadのみの機能しか有していないため、今後Rx Offloadへの対応と追加検証が必要となる。また、車両からの画像・動画データを想定した大きなサイズ（500KB ～ 1MB程度）での検証が必要と考える。
６．5Gフィールド検証	LTE／5G網経由での大容量データアップロード／ダウンロードを実施し、5Gは非常に高速かつ大容量であることを確認した	–

今後の取り組み

今後の取り組み

2018年12月より、コネクティッドカー分野における代表的なサービスの実現可能性やそれを支えるICT基盤技術の評価のために実証実験を実施。

その結果、下記の2つの目的を果たすとともに、接続車両数(量)やリアルタイム性等の性能目標を達成することができた。

コネクティッドカー・自動運転車の普及時における

- クラウド基盤技術／通信インフラ技術の見極め
- 次世代車両の要件整理

今後は、さらなるコネクティッドカーの普及に備えて、下記に取り組み、事故や渋滞といった社会が直面している様々な課題の解決に資する技術開発を継続して推進していく予定である。

- コネクティッドカー向けICT基盤の高速化・効率化・精緻化
- クルマから得られるビッグデータの有効活用・付加価値向上

将来的には、現状のインフラの限界を超えた通信ならびに計算リソースを提供できるように、様々な企業・団体やサービスと連携を図りつつ技術開発を進める。
さらに、来るべき自動運転時代の安心と安全をもたらす持続可能なスマートモビリティ社会の実現をめざしていきたい。

商標について

本書に記載されている各社の会社名、製品名に関する登録商標および商標は以下のとおりです。

・AMD、AMD Arrowロゴ、ATI、およびそれらの組み合わせ、RadeonはAdvanced Micro Devices, Inc.の商標です。
・Amazon、Amazon Web Services、AWS、Amazon EC2、Amazon S3は、米国その他の諸国における、Amazon.com, Inc.またはその関連会社の商標です。
・Apache、Tomcat、Apache Hadoop、Apache Ignite、Apache Zookeeper、Apache KafkaおよびApache Sparkは、Apache Software Foundationの米国およびその他の国における登録商標または商標です。
・DockerおよびDockerのロゴはDocker,Inc.の米国およびその他の国における商標または登録商標です。
・Elasticsearch は，Elastic社の米国およびその他の国における登録商標または商標です。
・Javaは，Oracle Corporationおよびその子会社，関連会社の米国およびその他の国における登録商標である。
・Kubernetes は、米国及びその他の国における The Linux Foundation の商標または登録商標です。
・Linuxは、Linus Torvalds氏の日本およびその他の国における登録商標または商標です。
・Microsoft、Windows、Windows Aero、Internet Explorer、Office、OneNote、Outlook、Excel、PowerPoint、Windows Live、Windows MediaおよびWindowsのロゴは、米国Microsoft Corporationの米国およびその他の国における商標または登録商標です。Microsoft Corporationのガイドラインに従って画面を使用しています。
・OracleはOracle Corporationおよびその関連企業の登録商標です。
・PostgreSQLは、PostgreSQLの米国およびその他の国における商標または登録商標です。
・Redisは, Salvatore Sanfilippoの商標です。
・TensorFlowは，Google Inc.の米国およびその他の国における登録商標または商標です。
・Ubuntuは英国Canonical Ltd.の登録商標です
・インテル、Intel、Intel Inside、Intel Inside ロゴ、Intel SpeedStep、インテル Core、インテル vPro、vPro Inside、インテル Atom、CeleronおよびPentiumはアメリカ合衆国およびその他の国におけるインテルコーポレーションまたはその子会社の商標または登録商標です。

その他会社名、各製品名は、各社の商標または登録商標です。

Appendix

ICTに関する技術について

ICTに関する技術について(1/4)

TOYOTA NTTグループ

«Appendix①高速時空間データ管理技術と時空間データ高速検索技術[1][2][3][4]»

大量の車両データを瞬時に格納し、特定の範囲（時間・空間）で高速に検索可能とする技術である。
具体的には、自動車やスマートフォンなどの動くモノが生成する時刻情報・位置情報を独自方式で1次元データとして管理し、特定時間範囲内かつ特定の空間範囲内に存在した車両の検索や、道路や駐車場などの複雑な形状の空間検索を高速に行える時空間データベースである。
実空間上の大量の動的オブジェクトが生成したセンサ情報と、そのセンサ情報に紐付けられた時間情報と空間情報を格納しつつ、ある特定の時間情報と空間情報を矩形範囲で指定し、そこに含まれるセンサ情報の高速検索が可能となる。
また、時空間データ高速検索技術を土台として、道路などの矩形以外の複雑な形状による範囲検索など、より高度な時空間データ管理機能を提供する。
例えば、多角形(ポリゴン)で表現される道路や公園、学区等に含まれる自動車やスマートフォン（ヒト）のみを高速に抽出できる。

«Appendix②車両データ選択的収集アルゴリズム»

位置情報や時刻情報、センサの種別、解像度、観測範囲、周辺車両による遮蔽状況といったメタ情報に基づき、データ収集の優先順位を判断する技術である。コネクティッドカーから収集するデータは、車両データ、動画像データ、センサデータと多岐にわたり、ボリュームも大きい。これが車両台数分、さらに逐次収集するとなると、膨大な量となり、トラフィック増によるネットワークやデータ処理の高負荷が懸念されるが、観測範囲の重複が少ないデータを効率良く収集することや、通信回線と分析基盤の負荷状況に応じて収集量を調節することが可能となる。

ICTに関する技術について(2/4)

TOYOTA NTTグループ

«Appendix③レーン別渋滞検知技術»

車載カメラで撮影した映像や車速・位置情報等の走行データを分析して、レーン単位の渋滞車列を検知する技術である。レーン単位の渋滞車列が続く区間を推定することで先頭/末尾車両も特定し、その位置関係から渋滞長を算出する。なお、安価なドライブレコーダー等の単眼カメラ映像にも適用することが出来る。
現在、VICS（Vehicle Information and Communication System）や地図アプリ等で道路単位の渋滞情報は提供されているが、本技術が実用化・普及すれば、より細やかな渋滞情報の把握が行える。さらに特定した先頭車両付近の映像と位置情報を、周辺施設や信号機といった地物の位置情報と組み合わせることで渋滞車列の原因推測も可能となる。

«Appendix④垂直分散コンピューティング技術»

従来、センタなどで行っていた処理機能を、エッジサーバに分散処理させることで、高速なアプリケーション処理を実現する技術である。すべてのデバイスに対して即応答を実現しようとすると、サーバのリソース不足により、高速応答の実現が困難になる場合が発生するが、実際には車両の状態（車両の速度・移動方向・時間帯など）ごとに高速応答の必要性が異なる。そこで、車両の状態に応じて、応答処理を実行するサーバを動的に変更することで、限られたサーバリソースを有効利用しつつ、必要な車両に確実に高速応答ができるようアプリを動的に分散配置することが本技術のねらいである。これにより、大量の車両を収容し、大量のデータの収集・応答が可能となる、限られた分散コンピューティングリソースの効率的な利用を実現する。

«Appendix⑤集計突発指標算出技術»

時系列ログにおける集計値の周期性/突発性を定常状態からの乖離度合いを元に高速に指標化する技術である。車両台数などの集計値の平均や分散を複数の異なる時間軸（全期間、曜日毎、時間帯毎、曜日×時間帯毎）にて学習し、最新の集計値の突発度合いを指標化する。算出した時間軸別の突発指標値について集計回数を考慮した重み付け線形和で融合して一元化することにより、突発的な集計台数の増加を突発的渋滞候補箇所として検出する。交通流最適化に向け、③レーン列渋滞検知技術の前段で本成果を適用することで、映像取得対象とする地域の優先付けや頻度のコントロールによる通信コストとサーバ負荷の削減効果が期待される。

ICTに関する技術について(3/4)

TOYOTA NTTグループ

«Appendix⑥路上表示物位置推定技術[5]»

市販の安価な単眼ドライブレコーダーの映像とGPSを用いて、道路周辺地物の位置座標を三角測量で推定する技術である。大量に収集した映像・データから良質なデータを選別し、統計的に推定結果を自動選別することで、位置推定精度を向上させる。道路標識や看板・信号機など、道路上の立体地物を正確に認識し、高精度に位置を推定することで地図へマッピングすることが可能となる。本技術が将来的に実用化・普及すれば、最新の道路状況を反映した自動運転用の高精度地図生成や地物情報の自動更新に関わるコスト削減効果が期待される。

«Appendix⑦動画細分化送信技術»

車載カメラで撮影した映像を1秒以下の単位で送信する技術である(極論はフレーム単位で送信可能)。従来、送信される動画データの長さは最短1秒であったため、基盤側での処理開始までに必ず1秒以上の待ちが発生していた。そこで、本技術を用いることで、開始待ちを短縮できる分だけ高速化が期待される。
なお、動画データに紐づくCANデータは1秒単位で生成されるが、並列で送信することにより、各々が生成されたタイミングで送信可能となっている。

ICTに関する技術について(4/4)

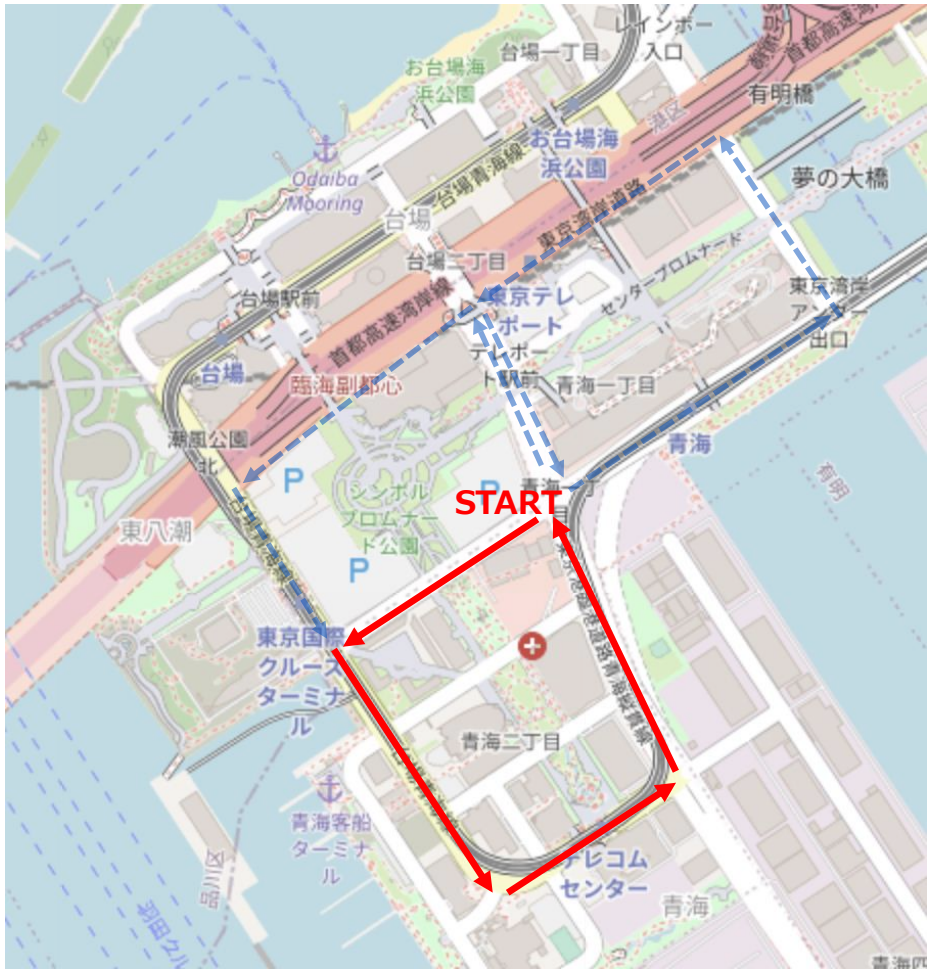
TOYOTA NTTグループ

«参考文献»
(1)A. Isomura: “Real-time spatiotemporal data utilization for future mobility services、”RedisConf19, Jun 2019
(2)RedisConf19 presentation
<https://www.slideshare.net/RedisLabs/realtime-spatiotemporal-data-utilization-for-future-mobility-services-atsushi-isomura>
(3)RedisConf19 video <https://www.youtube.com/watch?v=jVnIkwpcL3U>
(4)NTT技術ジャーナル <https://www.ntt.co.jp/journal/1911/JN20191118_h.html>
(5)Aki Hayashi, Yuki Yokohata, Takahiro Hata, Kouhei Mori, Kazuaki Obana .
The Road Environment Measurement for Automatic Road Map Generation,
2021 Volume 52 Issue 2 419-424,Published: 2021,Released: February 19, 2021

実証実験の試験走行ルート

実証実験の試験走行ルート

お台場エリアにてユースケース検証を行うにあたり検証コースを設定した。以下の赤枠ルートを基本ルートとし、必要に応じて灰色破線ルートを追加して走行した。

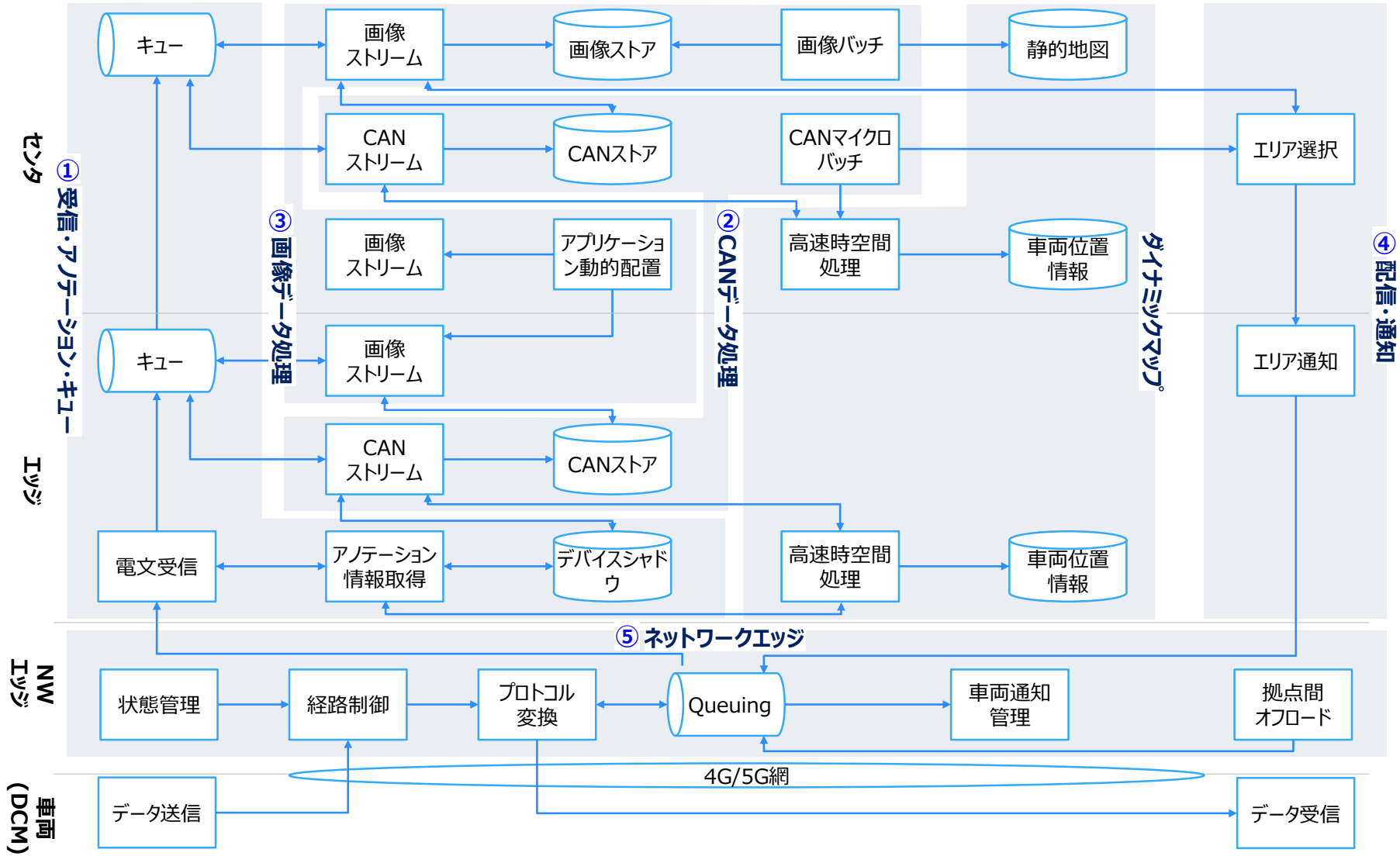


© OpenStreetMap contributors

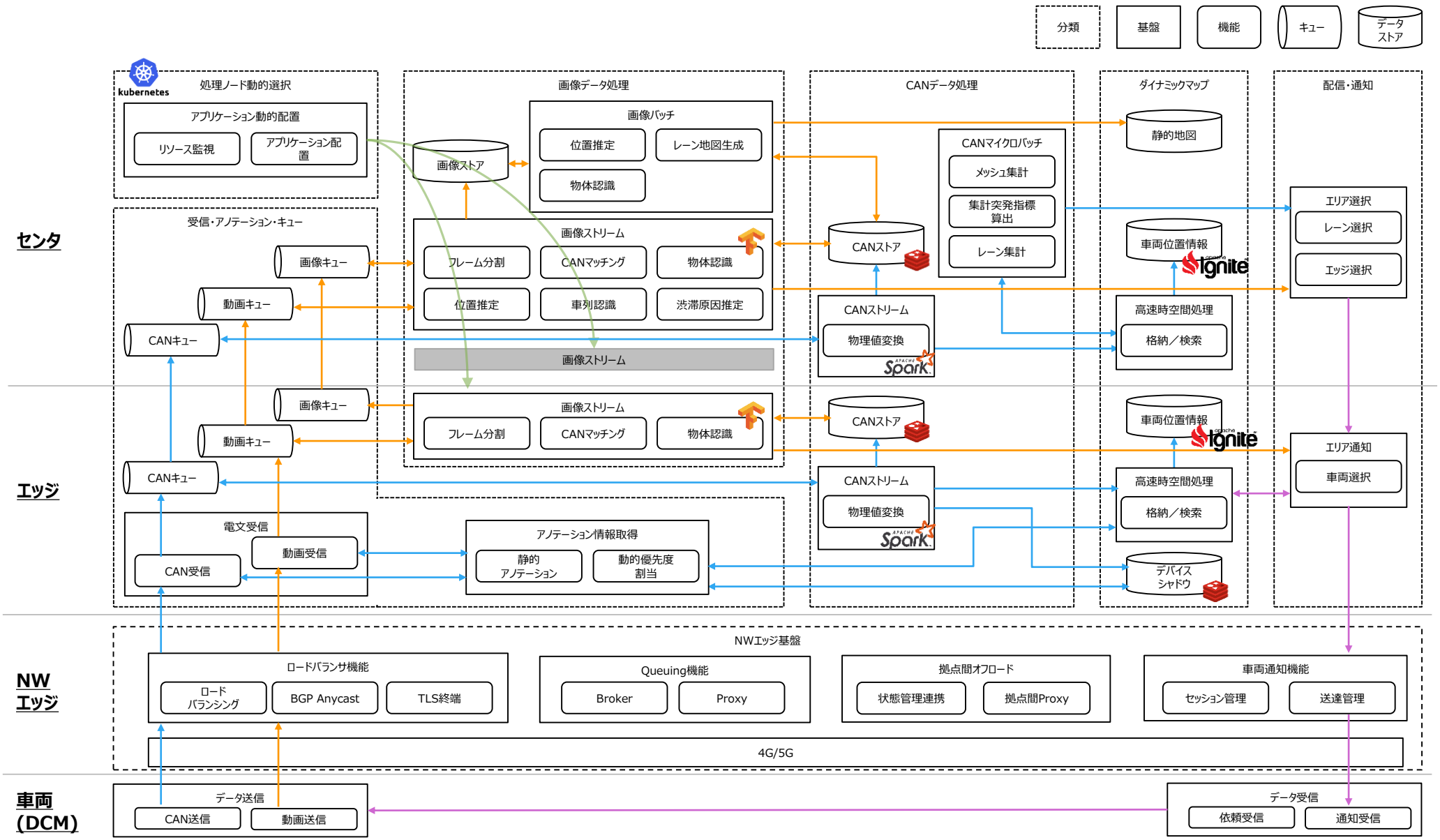
基盤検証のシステムについて

アーキテクチャ

基盤検証は、以下＃①～⑤の基盤単位で検証する。



アーキテクチャ(詳細版)



ソフトウェア一覧

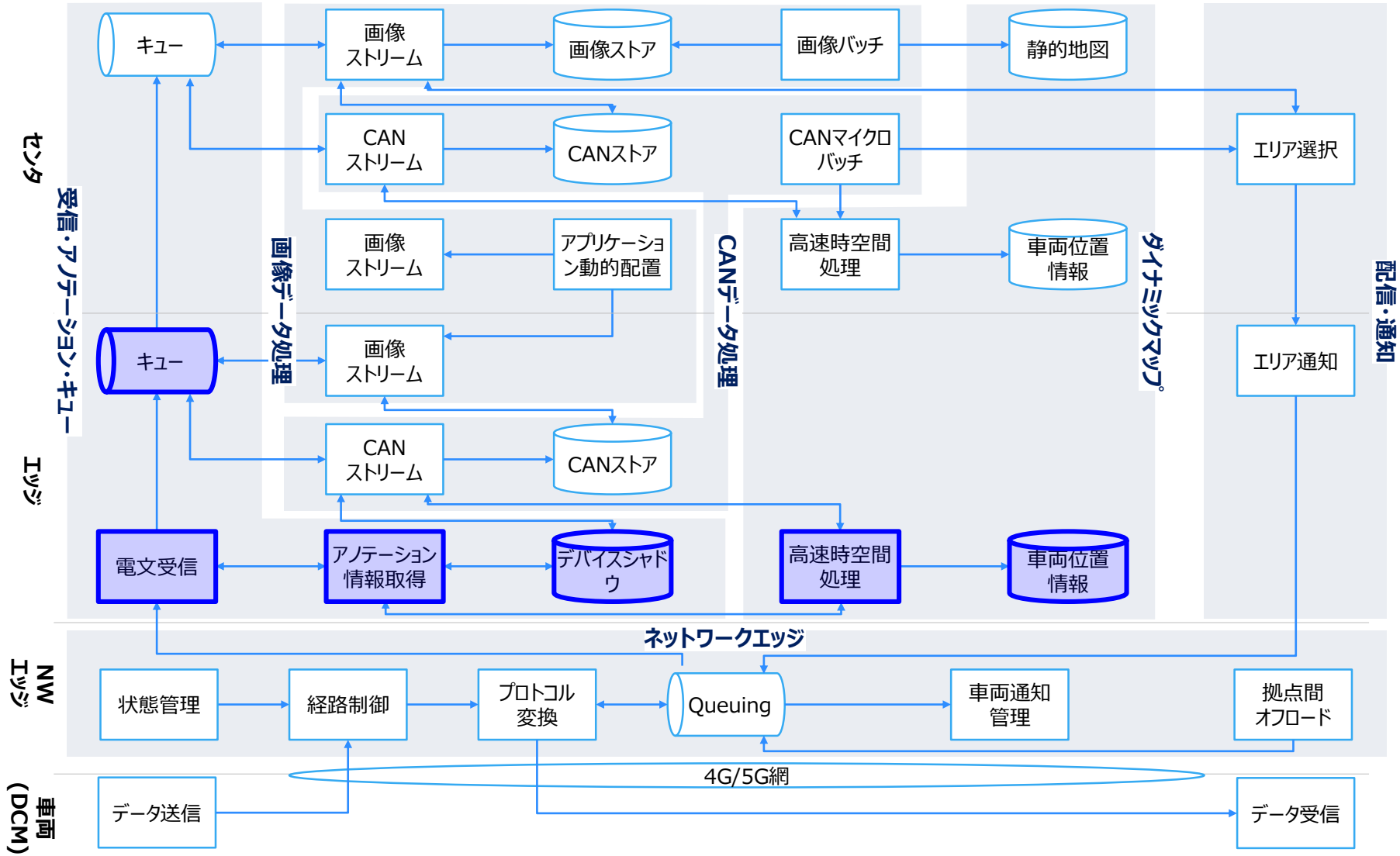
TOYOTA NTTグループ

基盤検証で用いたソフトウェア一覧を記述する。

項番	ソフトウェア名	説明	受信・アノテーション・キュー	CANデータ処理	画像データ処理	ダイナミックマップ	配信／通知	NWエッジ
1	CentOS	OS	○	○	○	○	○	–
2	OpenJDK	Java実行環境	○	○	○	○	○	–
3	Netty	JavaのNIOをラップしたフレームワーク	○	–	○	–	○	–
4	Apache Kafka	分散メッセージング処理向けのミドルウェア	○	○	○	–	–	–
5	NATS	分散メッセージング処理向けのミドルウェア	–	–	–	–	–	○
6	Apache HDFS	分散ファイルシステム	–	○	–	–	–	–
7	Apache YARN	分散リソース管理機構	–	○	–	–	–	–
8	Apache Spark	並列分散処理エンジン	–	○	–	–	–	–
9	Redis	分散インメモリKVS	–	○	–	–	○	–
10	Apache Ignite	分散インメモリデータベース	–	–	–	○	–	–
11	Elasticsearch	分散型検索／分析エンジン	–	–	–	–	○	–
12	UltraMonkey	ソフトウェアロードバランサ	○	–	○	–	–	–
13	Docker	コンテナ型仮想化環境を提供するプラットフォーム	–	–	○	–	○	–
14	Kubernetes	複数ノード上で起動するコンテナの統合管理を行なうミドルウェア	–	–	○	–	○	–
15	Hitch	TLS復号ミドルウェア	○	–	○	–	○	–
16	PostgreSQL	関係データベース管理システム（RDBMS）	–	–	–	○	–	–
17	PostGIS	PostgreSQL データベースで地理空間情報を扱うための拡張	–	–	–	○	–	–
18	Nginx	Webサーバソフトウェアでロードバランサとして使用	–	–	–	–	–	○
19	OpenSSL	SSL/TLS通信ライブラリ	–	–	–	–	–	○
20	FRRouting	ルーティングプロトコル管理ソフトウェア	–	–	–	–	–	○

アーキテクチャ

受信・アノテーション・キュー基盤検証で利用するコンポーネントを青枠で示す。



機能一覧

受信・アノテーション・キューのコンポーネントにおける機能一覧は以下の通り。

#	分類	機能名	説明
1	電文受信	CAN・動画電文受信	電文をHTTPプロトコルにより受信する。
2	アノテーション情報取得	静的アノテーション情報取得／静的・動的判定	① アノテーションマスタを参照し、アノテーション情報の初期値を取得する。 ② 電文のデータ種別を参照し、イベント動画電文であればエリア判定処理を行う。 ③ 優先度判定処理を行わない場合は取得したアノテーション情報を返却する。
3		エリア判定	イベント動画電文において、以下の流れで優先度判定処理を行う。 ① 周辺メッシュに存在する車両一覧取得 ② 平均移動速度等の計算による絞り込み ③ 遮蔽判定 ④ エリアしきい値判定
4		データキュー出力	アノテーション情報をもとに、電文をCANキュー／動画キューに送信する。
5	キュー	CANキュー、動画キュー	アノテーション済み電文を一時的に保存する。

ハードウェア一覧

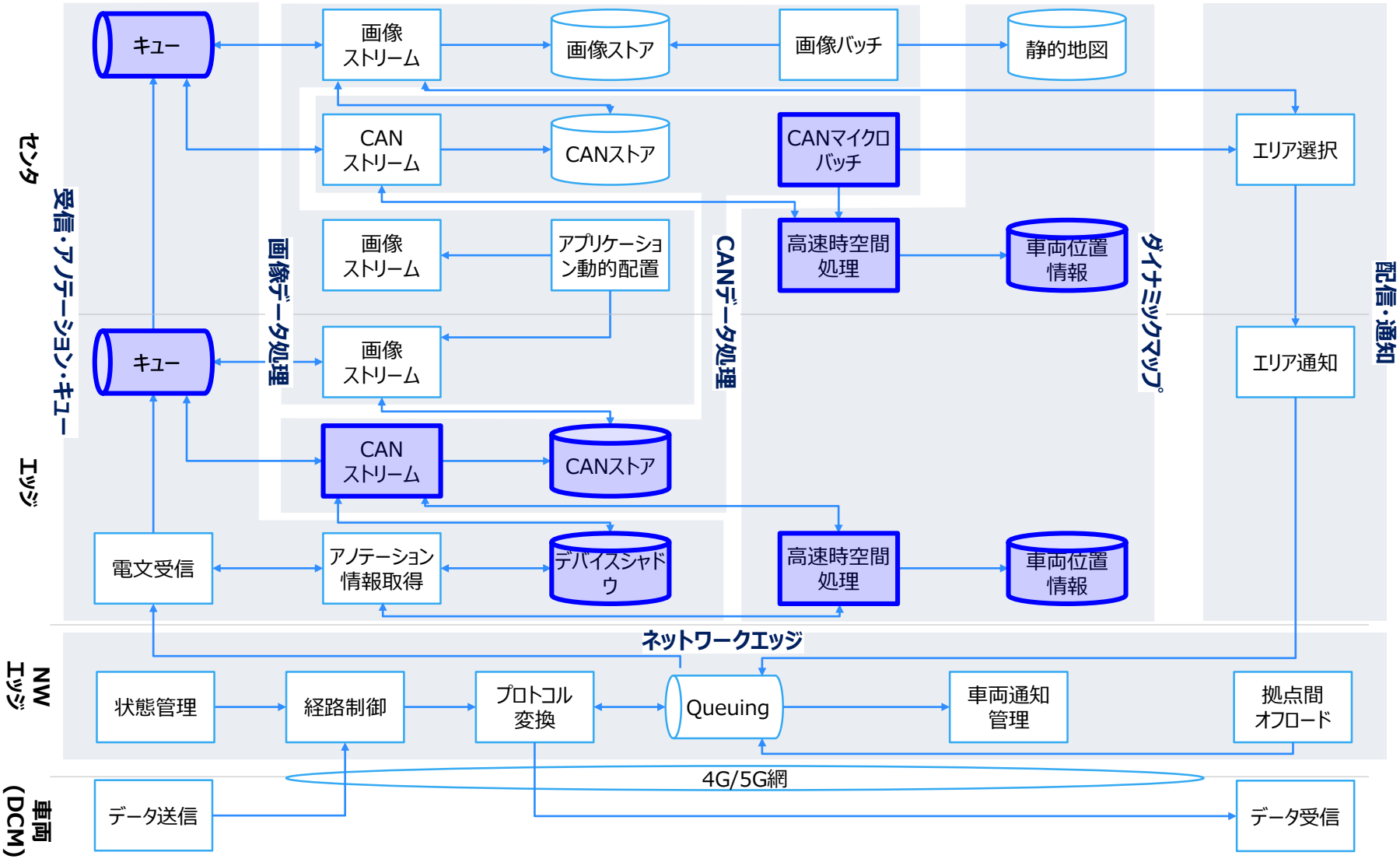
受信・アノテーション・キューのコンポーネントにおけるハードウェア一覧は以下の通り。

#	サーバ名	機能名	サーバ台数	サーバスペック
1	ジェネレータサーバ	CAN・動画電文送信	25台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 192GB ディスク(HDD) : 1,200GB
2	受信・アノテーションサーバ	CAN・動画電文受信、静的アノテーション情報取得／静的・動的判定、エリア判定、データキュー出力	6台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 192GB ディスク(HDD) : 1,200GB
3	デバイスシャドウ	-	1台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 192GB ディスク(HDD) : 1,200GB
4	車両位置情報	-	3台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 192GB ディスク(HDD) : 1,200GB
5	キューサーバ	CANキュー、動画キュー	9台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 288GB ディスク(SSD) : 1,600GB

アーキテクチャ

TOYOTA NTTグループ

CANデータ処理基盤検証で利用するコンポーネントを青枠で示す。



機能一覧

CANデータ処理のコンポーネントにおける機能一覧は以下の通り。

#	分類	機能名	説明
1	CANストリーム	物理値変換	①データキューに格納された電文を取得 ②フィルタリング ③電文中のバイナリを解釈して可読性の高いデータ形式に変換
2		車両位置登録	物理値変換後のデータを車両位置情報およびデバイスシャドウへ出力する
3		CANストア出力	イベント発生時に送信されるCANから物理値変換後のデータをEventストアに格納する
4	CANマイクロバッチ（集計処理）	メッシュ集計	指定されたメッシュ内の車両情報を取得し、存在する車両台数を算出する
5		集計突発指標算出	メッシュ集計によって算出された車両台数が、そのメッシュにおける平常状態よりも突発的に増加しているかを算出する
6		レーン別集計	集計突発指標算出によって算出されたメッシュに含まれるレーンに存在する車両の平均速度を算出し、しきい値以下となったものを渋滞候補レーンとして挙げる
7	キュー	CANキュー、渋滞候補キュー	アノテーション済みCAN電文、渋滞絞り込み結果を一時的に保存する。

ハードウェア一覧

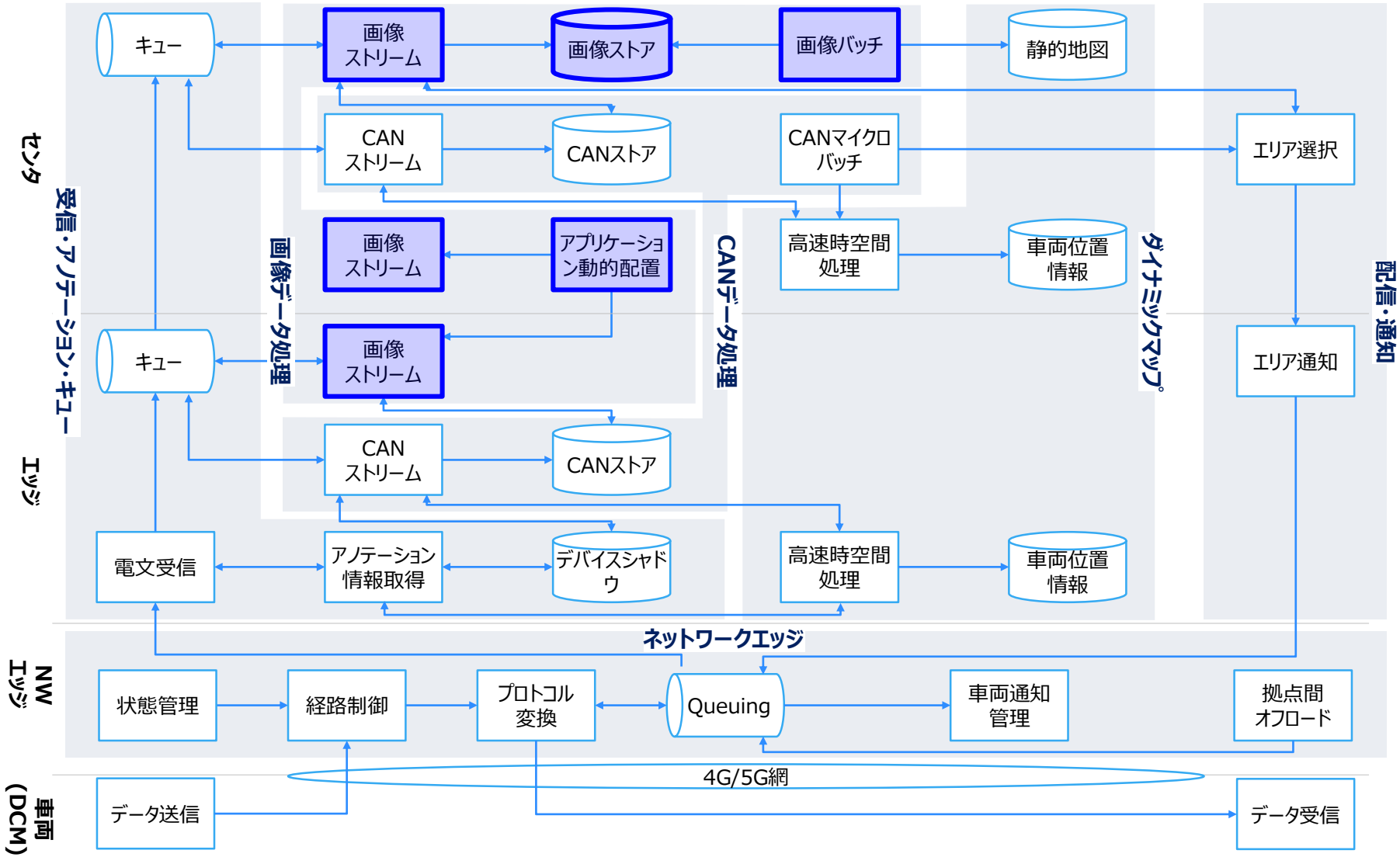
CANデータ処理のコンポーネントにおけるハードウェア一覧は以下の通り。

#	サーバ名	機能名	サーバ台数	サーバスペック
1	キューサーバ	CANキュー、渋滞候補キュー	8台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 288GB ディスク(SSD) : 1,600GB
2	CANストリームサーバ	物理値変換、車両位置登録、CANストア出力	a : 5台 b : 15台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 192GB ディスク : 1,200GB
3	デバイスシャドウ	-	1台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 192GB ディスク(HDD) : 1,200GB
4	車両位置情報	-	3台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 192GB ディスク(HDD) : 1,200GB
5	CANストア	-	1台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 192GB ディスク : 1,200GB
6	CANマイクロバッチサーバ	メッシュ集計、レーン別集計	3台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 192GB ディスク : 1,200GB
7	集計突発指標算出サーバ	集計突発指標算出	1台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 288GB ディスク : 1,600GB

アーキテクチャ

TOYOTA NTTグループ

画像データ処理基盤検証で利用するコンポーネントを青枠で示す。



機能一覧

TOYOTA NTTグループ

画像データ処理のコンポーネントにおける機能一覧は以下の通り。

#	分類	機能名	説明
1	画像ストリーム	障害物検知・通知	①動画データを画像データに分割する ②画像データに紐づくCANデータをCANストアから取得する ③障害物特定コンテナに画像データを送信し、重複して出現する障害物の同定判定を行う ④障害物情報を障害物キューに格納し、配信処理に障害物情報を引き渡す
2		障害物特定	画像に映っている障害物を検出する。
3		位置推定呼出・障害物詳細通知	①画像データに紐づくCANデータをCANストアから取得する ②位置推定コンテナに画像データとCANデータを送信し、位置推定結果を受信する ③配信処理に障害物情報を引き渡す
4		位置推定	位置推定を行い物体ラベルと位置座標（緯度経度）を出力する。
5		隣接レーン車列認識・渋滞原因分析	①車両を撮影した画像から物体検出手法と撮影状況の解析を行い、撮影者の隣接レーンに渋滞が発生していることを検知する ②渋滞の発生を検知した地点から、その周辺の地物情報等を元に渋滞原因を推定する
6	アプリケーション動的配置	メトリクス収集・集計・しきい値判定	①各拠点に配備しているGPUノードのメトリクス情報を収集する ②単位時間あたりのGPU使用率の平均値を算出する ③メトリクス集計にて算出した平均値がしきい値を超えているか判定する
7		アプリケーション配置先計画・選択	①拠点内でボトルネックを検知した際の配置する拠点を予め計画しておく ②拠点内でボトルネックを検知した場合、計画されている配置拠点先の状況を踏まえ、最終的な配置先を決定し、デプロイする
8	画像バッチ	信号機／標識認識・位置推定	①画像に映っている信号機・標識を検出する ②位置推定を行い物体ラベルと位置座標（緯度経度）を出力する
9		レーンNW地図生成	他拠点にあるDCに動画・CANデータを連携し、レーンNW地図を生成する
10		静的地図生成	レーンNW地図と位置推定の出力結果を統合し、静的地図に登録する

ハードウェア一覧

TOYOTA NTTグループ

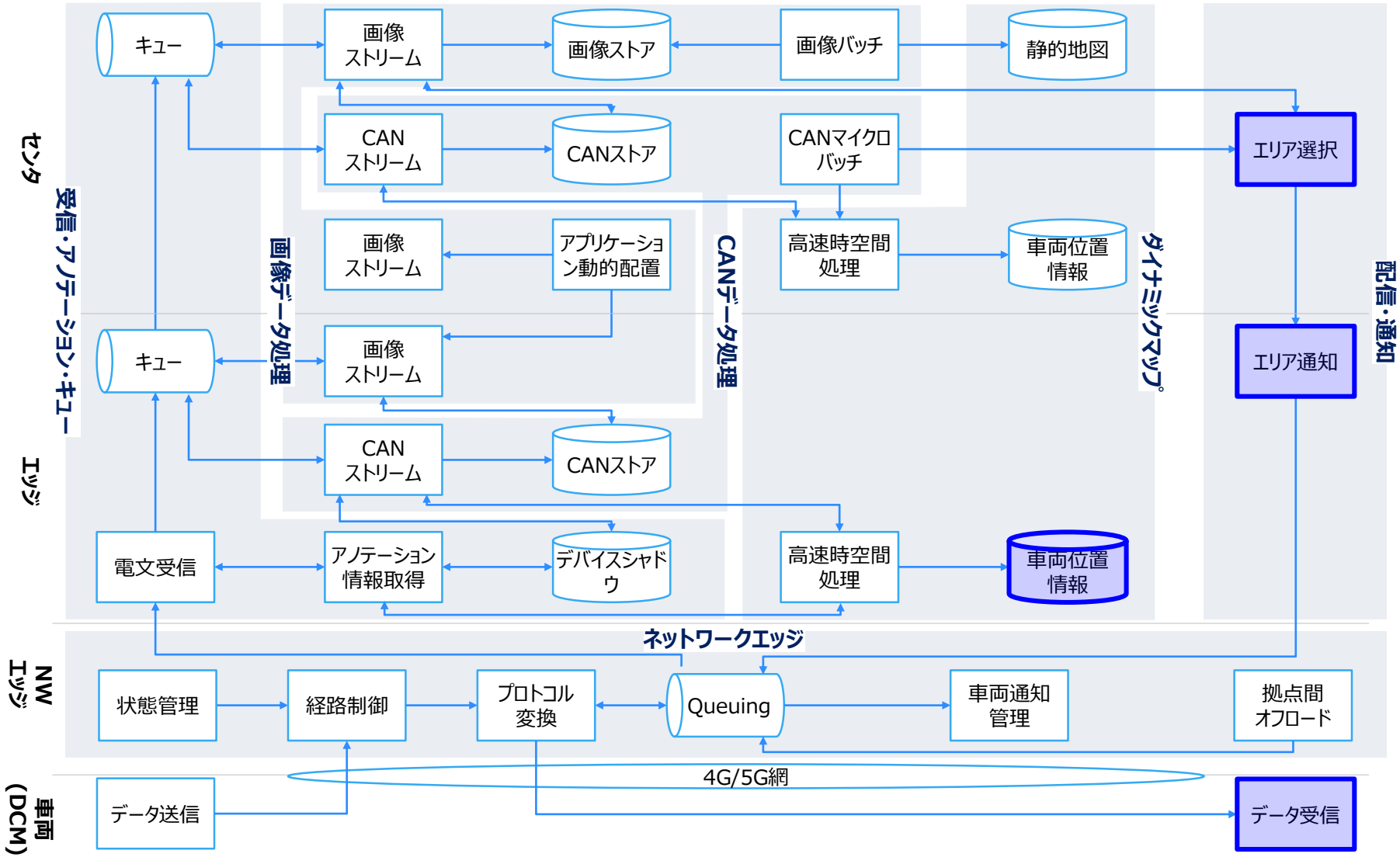
画像データ処理のコンポーネントにおけるハードウェア一覧は以下の通り。

#	サーバ名	拠点	機能名	サーバ台数	サーバスペック
1	キューサーバ	エッジ	動画キュー	1台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 288GB ディスク(SSD) : 1,600GB
2	画像ストリームサーバ （GPU搭載）	エッジ	障害物検知・通知、障害物特定	2台	GPU : Nvidia V100-PCIe CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 288GB ディスク(SSD) : 1,600GB
3		センタ	隣接レーン車列認識・渋滞原因分析	2台	
			信号機／標識認識・位置推定、レーンNW地図生成、静的地図生成	4台	
4	画像ストリームサーバ （GPU非搭載）	センタ	位置推定呼出・障害物詳細通知、位置推定	9台	CPU(Xeon Silver 4110) : 2P,16C,32T メモリ : 192GB ディスク(HDD) : 1,200GB
5	画像ストリームLBサーバ	センタ	位置推定呼出	1台	CPU(Xeon E5-2630v4) : 2P,20C,40T メモリ : 128GB ディスク(HDD) : 300GB
6	コンテナ管理サーバ	センタ	メトリクス収集・集計・しきい値判定、アプリケーション配置先計画・選択	1台	CPU(Xeon E5-2630v4) : 2P,20C,40T メモリ : 128GB ディスク(HDD) : 300GB
7	CANストア	エッジ・センタ	-	各1台ずつ	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 192GB ディスク(HDD) : 1,200GB
8	画像ストア	センタ	メッシュ集計、レーン別集計	4台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 288GB ディスク(SSD) : 1,600GB

アーキテクチャ

TOYOTA NTTグループ

配信・通知基盤検証で利用するコンポーネントを青枠で示す。



機能一覧

配信・通知処理のコンポーネントにおける機能一覧は以下の通り。

#	分類	機能名	説明
1	エリア選択	対象エッジサーバ検索	通知依頼を転送すべきエッジを特定する。
2	エリア通知	対象車両検索	画像データ送信依頼の送信対象となる車両の絞り込みを実施する。本開発においては、以下の2パターンの絞り込みロジックを想定している。 ①車両の位置情報のみを利用した対象車両の検索 ②車両に搭載されるカメラスペックや画角を考慮した、より精度の高い検索
3		通知依頼	「対象車両検索」にて特定された車両一覧と、データ送信依頼の内容を、「情報通知」に送信する。

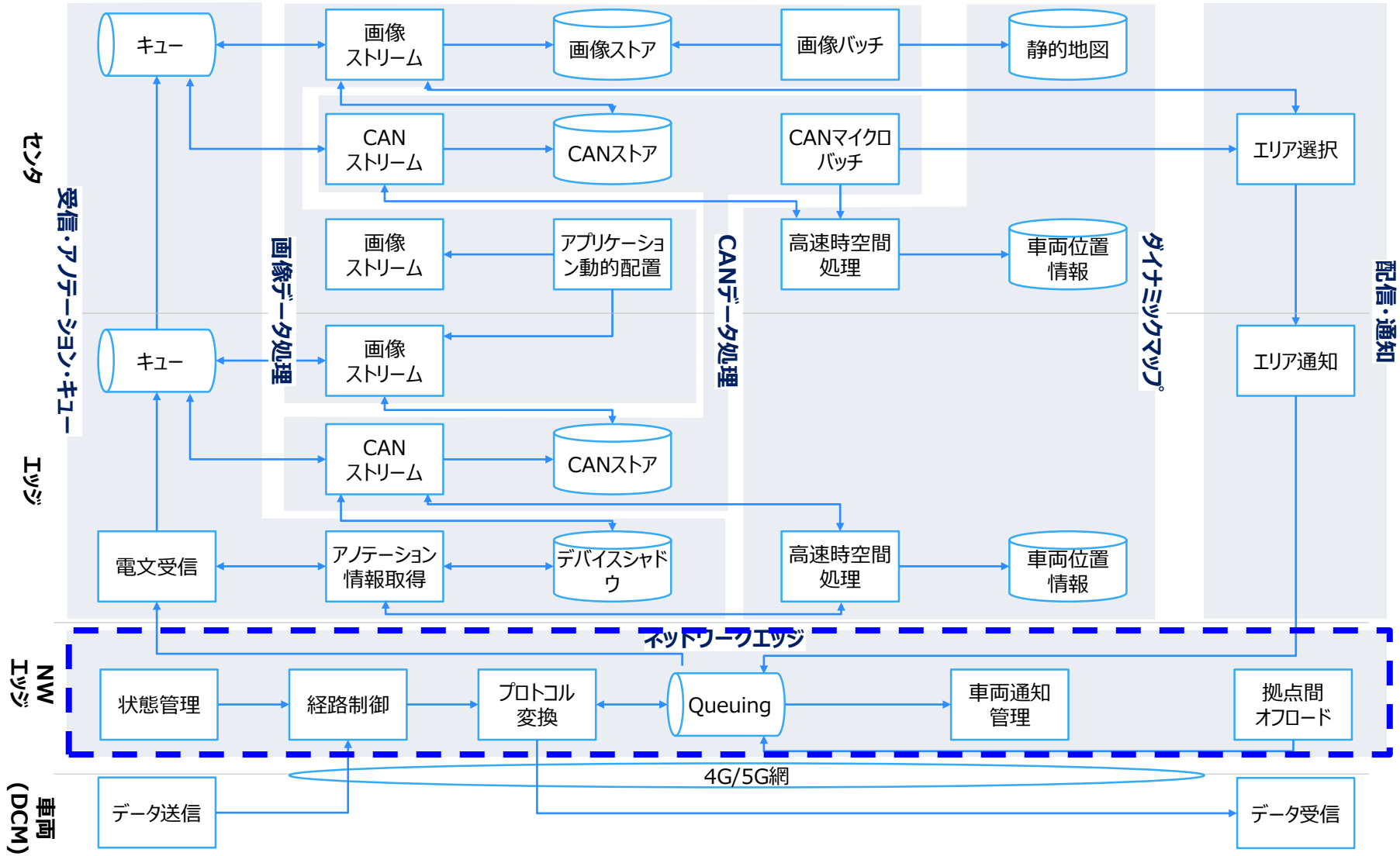
ハードウェア一覧

配信・通知処理のコンポーネントにおけるハードウェア一覧は以下の通り。

#	サーバ名	機能名	サーバ台数	サーバスペック
1	エリア選択サーバ	対象エッジサーバ検索	1台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 192GB ディスク(HDD) : 1,200GB
2	エリア通知サーバ	対象車両検索、通知依頼	1台	CPU(Xeon E5-2630 v4) : 2P,40C,80T メモリ : 128GB ディスク(HDD) : 1TB
3	車両位置情報	-	3台	CPU(Xeon Gold 5118) : 2P,24C,48T メモリ : 192GB ディスク(HDD) : 1,200GB

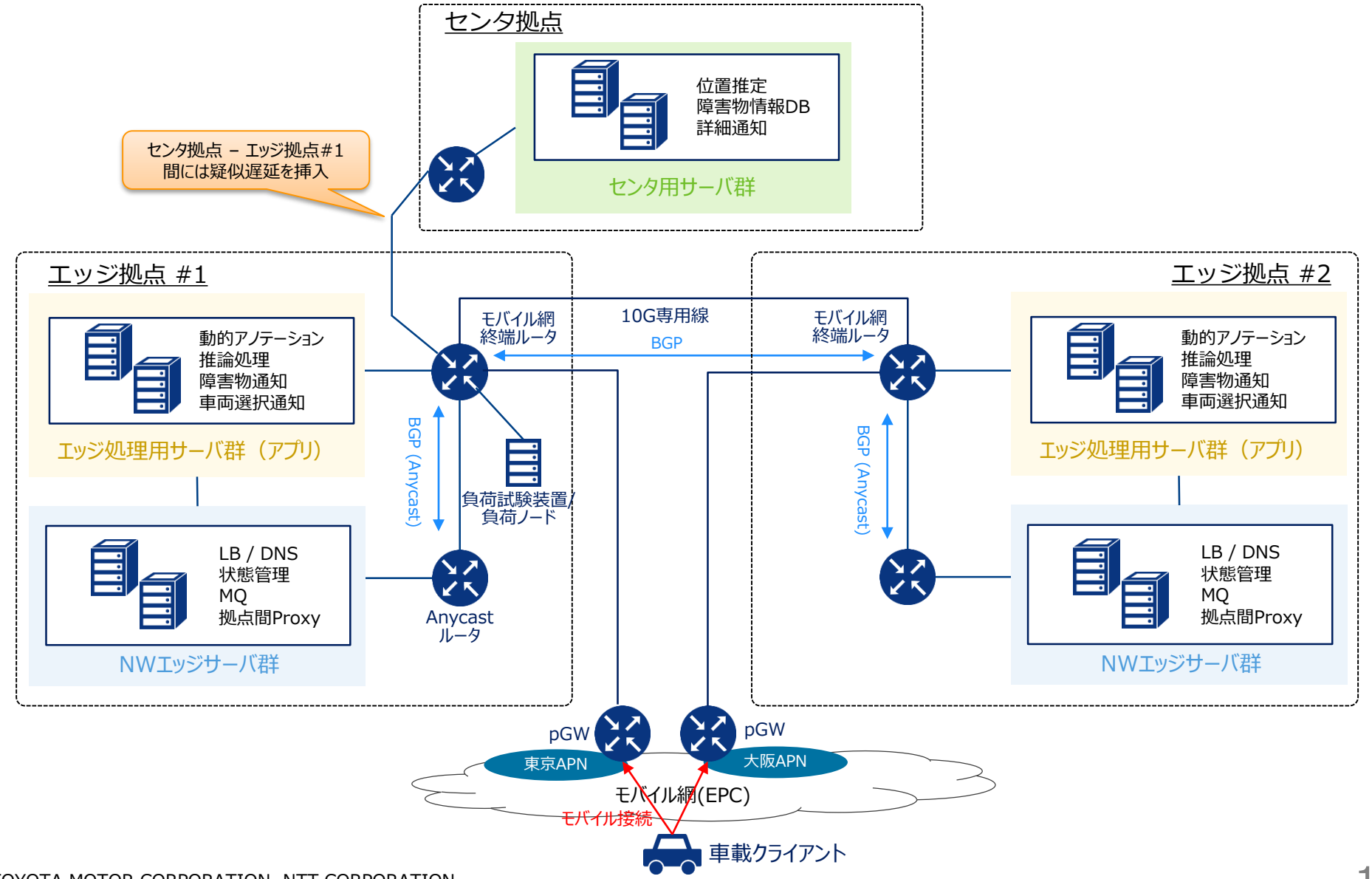
アーキテクチャ

ネットワークエッジ基盤検証は以下の青枠部分が検証対象である。



システム構成

NWエッジ基盤検証に用いたシステム構成を示す。



機能一覧

ネットワークエッジ基盤の機能を示す。
本実証実験ではこれら機能を主にIAサーバに実装し、台数を増やすことでスケールアウト可能とした。

#	機能名		機能説明
1	経路制御	BGP Anycast	電文送信宛先IPアドレスをBGP Anycastで伝搬させることにより、車両からのデータを最寄りのNWエッジへの誘導する。
		ロードバランシング	設定したポリシーに基づいて電文の振り分け・負荷分散を行う。
		TLS終端	認証・TLS通信の復号を行う。
2	プロトコル変換	-	通信プロトコル（HTTP/MQTT/WebSocket等）の変換を行う。
3	Queuing	-	受信した電文を一定期間保持する。
4	拠点間オフロード	NWエッジ状態管理・連携	エッジ拠点間連携のため、他のNWエッジの状態を管理、連携拠点の選定を行う。
		拠点間Proxy	エッジ拠点間の電文の送受信を行う。
5	車両通知管理	セッション管理	車両とエッジ拠点間の接続状態を管理する。
		送達管理	車両への通知メッセージの送達状態を管理する。
6	状態管理	メトリクス収集	NWエッジのメトリクス情報を取得する。
		可視化	収集したメトリクス情報の可視化を行う。

ハードウェア構成一覧

検証１～４におけるNWエッジサーバのハード構成、ソフトウェア構成を示す。

検証１　DNS, LB, アノテーションサーバ	
ハイパーバイザスペック	
CPU	Intel Xeon E5-2650 2.00GHz 8C/16T
RAM	64GB
SAS	1.67TB (300G x8本 RAID50)
ハイパーバイザ	ESXi-6.5.20180502001-standard
仮想マシンスペック	
CPU	8vCPUs
RAM	32GB
HDD	64GB
OS	Ubuntu 18.04.1.0 LTS amd64
Kernel	Linux Kernel 4.15.0-45-generic
Nginx(LB)	1.15.6
dnsdist(DNS)	1.3.3 (Lua 5.1.4)
Knot DNS(DNS)	2.6.9

検証２・３　NWエッジ, アノテーションサーバ	
ハードウェアスペック	
CPU	Intel Xeon Gold 6140 2.3GHz 18C/36T x2
RAM	384GB
SSD	480GB (480G x2本 RAID1)
ソフトウェアスペック	
OS	Ubuntu 18.04.3.0 LTS
Kernel	Linux Kernel 4.15.0-135-generic
Nginx(LB)	1.14.0

検証４　NWエッジ, アノテーションサーバ	
ハードウェアスペック	
CPU	Intel Xeon Gold 6140 2.3GHz 18C/36T x2
RAM	384GB
SSD	480GB (480G x2本 RAID1)
ソフトウェアスペック	
OS	Ubuntu 20.04.1 LTS
Kernel	Linux Kernel 5.4.0-65-generic
Nginx(LB)	1.19.3

ハードウェア構成一覧

検証５におけるNWエッジサーバのハード構成、ソフトウェア構成を示す。

検証５　NWエッジ 共通	
ハードウェアスペック	
CPU	Intel Xeon Gold 6140 (2.3GHz, 18C) x2
RAM	384GB DDR4
SSD	480GB (RAID1)
ソフトウェアスペック	
OS	Ubuntu 20.04.1 LTS
nginx (LB)	1.18.0
TLS (Cipher)	v1.2 (ECDHE-RSA-AES128-SHA256)

検証５　NWエッジ #1 (CPU)	
ハードウェアスペック	
CPU w/ HT	Yes
NIC	Intel X710 x2
ソフトウェアスペック	
Kernel	Linux Kernel 5.4.0-42-generic
OpenSSL	1.1.1f

検証５　NWエッジ #2 (TLS Accelerator; QuickAssist Adapter)	
ハードウェアスペック	
CPU w/ HT	Yes
NIC	Intel X710
QAT board	QuickAssist Adapter 8950 x3
ソフトウェアスペック	
Kernel	Linux Kernel 5.4.0-65-generic
OpenSSL	1.1.1f

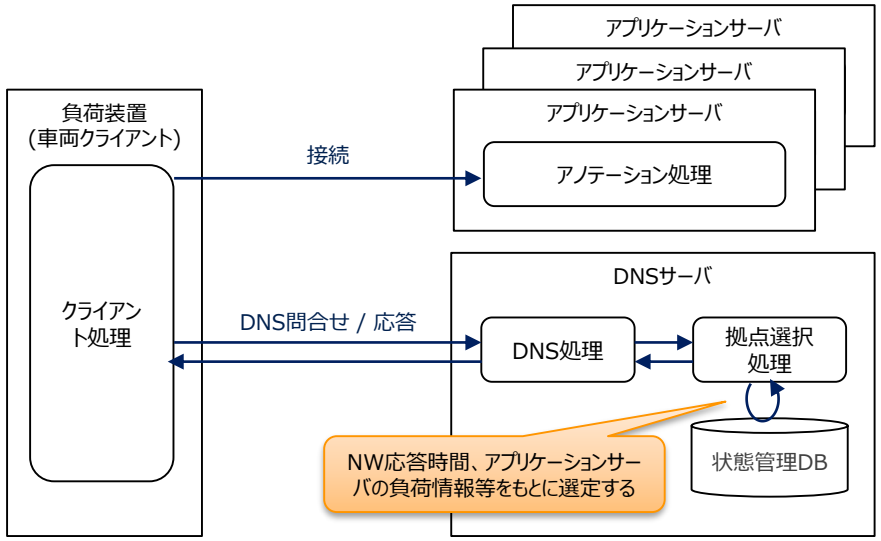
検証５　NWエッジ #3 (ASIC based SmartNIC)	
ハードウェアスペック	
CPU w/ HT	No
NIC	Intel X710
SmartNIC	Mellanox ConnectX-6 Dx
ソフトウェアスペック	
Kernel	Linux Kernel 5.4.0-65-generic (ktls enabled)
OpenSSL	3.0.0-alpha13

検証１．検証アーキテクチャ

車両に最も近いエッジ拠点への接続や拠点障害や負荷増大に伴う広域切替を実現するために、車両トラフィックの適切な経路制御が必要となる。本検証では経路制御方式としてDNS方式とLB方式を考案し、各方式の基本機能と性能を検証した。DNS方式とLB方式の検証アーキテクチャを以下に示す。

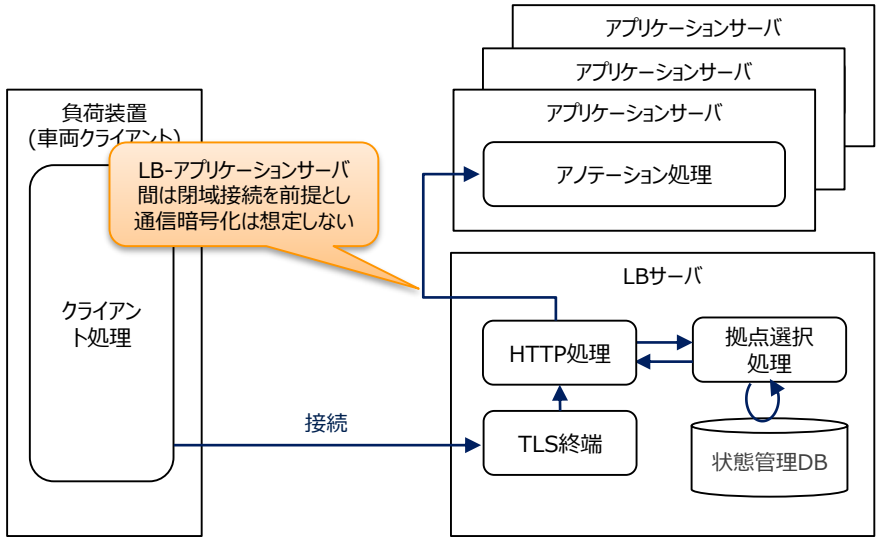
DNS方式

状態管理DBにてアプリケーションサーバ、NW等の負荷情報を保持する。
DNS処理にて状態管理DBの情報を参照し、最適なアプリケーションサーバを選定し、DNS応答することにより経路制御を行う。



LB方式

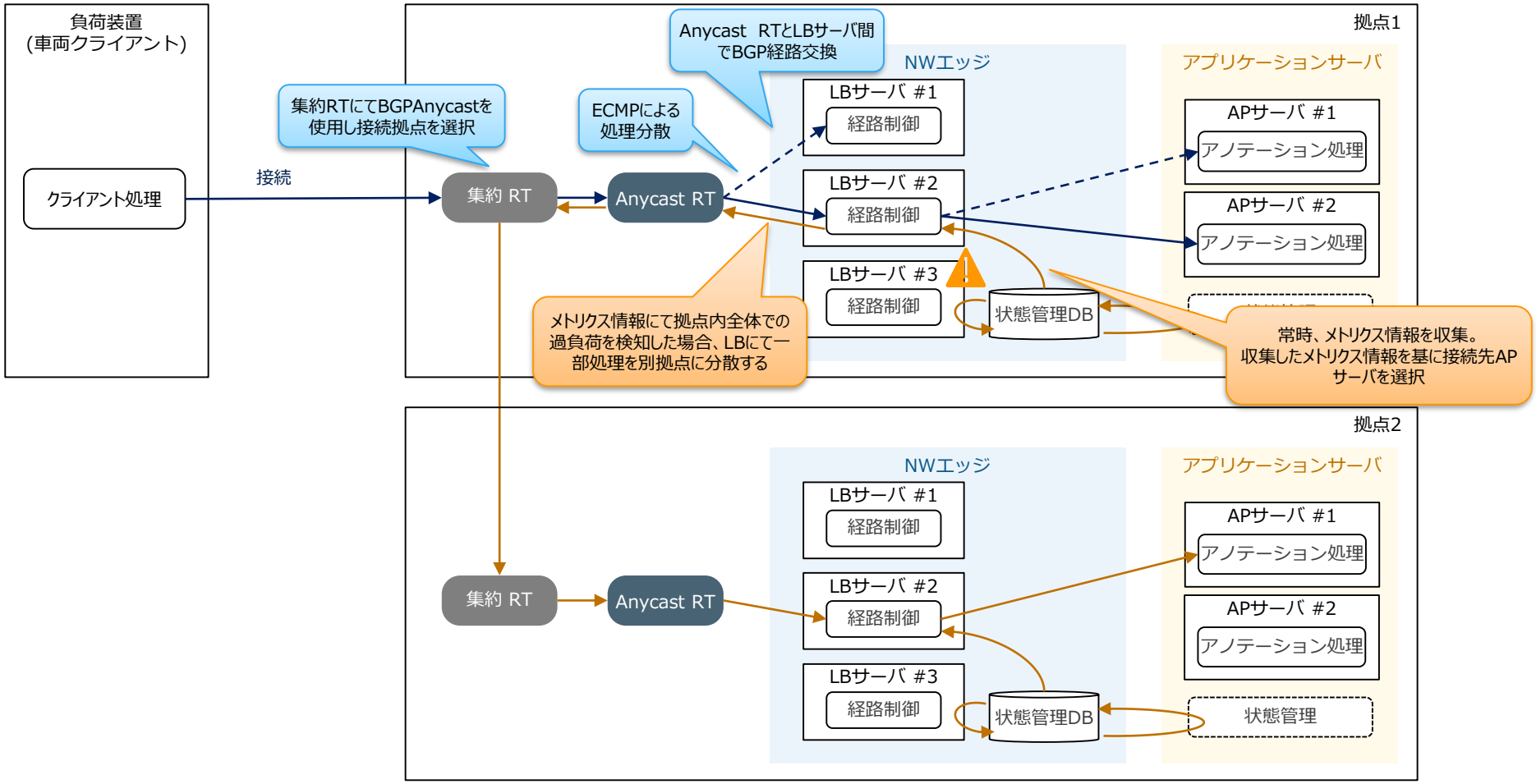
状態管理DBにてアプリケーションサーバ、NW等の負荷情報を保持する。
LBでの処理時に、状態管理DBを参照し、最適なアプリケーションサーバを選定し後続処理へ連携することにより経路制御を行う。



※ 接続するエッジ拠点(DNSサーバ、LBサーバ)の選定には、BGP Anycastの使用を想定する。
BGP Anycastでは、GW間の経路情報の広告により接続する拠点を選択する。
車両の移動により、接続モバイル網内のpGWが変更した時に再接続を実現するために導入する。

検証２・３． 検証アーキテクチャ

エッジ拠点の故障や負荷状況に応じた広域切替を実現するためには、メトリクス情報に基づいた経路制御が必要となる。本検証では、LB方式を前提とし、メトリクス情報に基づいた経路制御機能を検証した。1エッジ拠点の基礎性能検証を合わせて実施した。検証アーキテクチャを以下に示す。

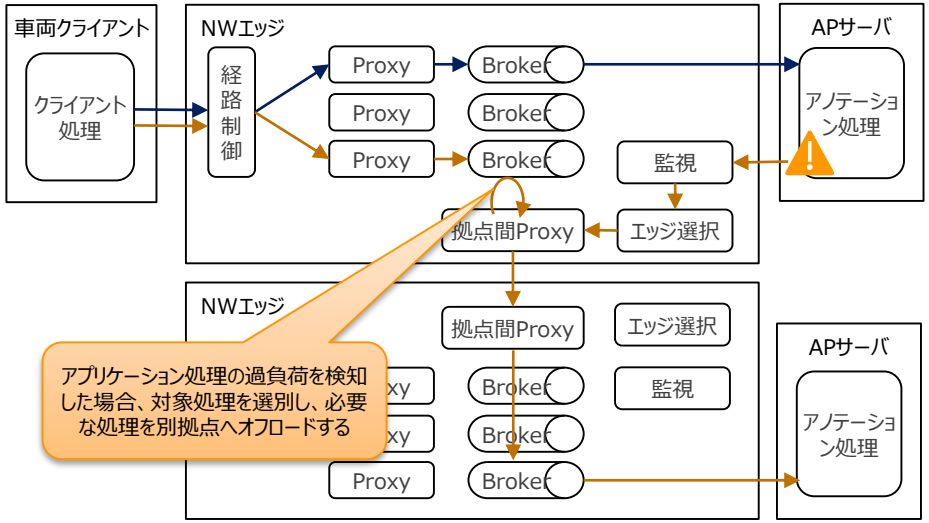


検証４．検証アーキテクチャ

特定のNWエッジ拠点が障害等の影響で可用性が失われた場合でも、車両との通信は継続させるため、NWエッジが複数分散する環境下において分散する拠点間でのデータ連携が重要となる。これに対し、NWエッジにMessage Queue(MQ)を導入することでNWエッジ全体でのデータ連携・可用性向上を実現する。本検証では、広域分散MQを利用したエッジ拠点間でのデータ連携に関する検証を行う。検証アーキテクチャを以下に示す。

車両からのデータ送信

特定拠点へ車両接続が集中しアプリケーション処理が過負荷となることが想定されるため、エッジ拠点間での処理のオフロードが必要となる。
アプリケーション処理の負荷状況を監視し過負荷を検知した場合、必要なデータのみ、拠点間Proxyにて別拠点へ処理をオフロードする。



車両へのメッセージ通知

車両の移動により、通知対象の車両の接続エッジが遷移することが想定されるため、エッジ間で通知メッセージを連携し、送達管理を行う必要がある。
複数拠点を跨った論理キューを構成し、拠点間のデータ連携を行う。
各拠点で通知メッセージのメタデータを共有し、車両からの接続要求等をトリガーに論理キューよりメッセージを取得し車両へ通知する。