論　文

車両制御システムのためのセンサデータ統合管理方式の検討※

山田　真大[†]　　鎌田　浩典[†*]　　佐藤　健哉[†,††]　　手嶋　茂晴[†]
高田　広章[†]

Integrated Sensor Data Management Method for Vehicle Control System※

Masahiro YAMADA[†], Hironori KAMADA[†*], Kenya SATO[†,††], Shigeharu TESHIMA[†], and Hiroaki TAKADA[†]

あらまし　近年，プリクラッシュセーフティ技術など，車両の状態や周辺状況を判断し，ドライバへの警告や自動制御により運転の支援を行う複数の車両制御システムが登場している．これらのシステムにおいては，複数のセンサデータを電子制御ユニットにおいて個別に管理し処理を行っているため，新規のセンサを追加したり，他の電子制御ユニットと連携する場合は，アプリケーションプログラムを変更する必要が生じる．また，将来に向けて，多種多様なシステムの設計・開発を統合することが困難となる．本研究では，システムからセンサ部を切り離し，確率を利用した占有グリッドにおいてセンサから得られたデータをシーングラフ併用で統合管理し，複数のアプリケーションプログラムから共通に利用可能なセンサデータ統合管理方式の検討を行う．また，この方式を採り入れたシステムの設計，実装を行い，シミュレータを利用し，車両追従システム，自動駐車システムの車両制御システムを構築し，有用性，実現可能性を示す．

キーワード　センサデータ統合，車両制御システム，データ管理，占有グリッド，シーングラフ

1. まえがき

近年，プリクラッシュセーフティ技術など，車両の状態や周辺状況を判断し，ドライバへの警告や自動制御により運転の支援を行う車両制御システムが登場している [1], [2]．例えば，車両に搭載された複数のセンサからの情報に基づき，操舵回避の支援を行い，衝突が避けられない状況では介入ブレーキを作動させることで衝突衝撃を緩和し被害を軽減するシステムがある [3], [4]．また，衝突回避システム，車両追従システム，レーン逸脱警告システム，自動駐車システムなどもある [5]．

これらの現状のシステムの構成モデル概略を図 1 に示す．このモデルでは，複数の異種センサによりデータを取得し，一つあるいは複数の電子制御ユニット (ECU) においてアプリケーションプログラムがデータ処理を行い，その結果，ステアリングやブレーキなどのアクチュエータの操作を行う．このような車両制御システムにおいて，周辺の物体を検知するセンサは多様である．そのため，異なるセンサを利用したり，新規のセンサを追加したりする場合は，アプリケーションプログラムを変更する必要が生じ，また，複数システムの設計・開発を統合することが困難となる．

本研究では，それぞれの車両制御システムにおけるアプリケーションの設計・開発を容易にする目的で，センサ部を切り離し，センサから得られたデータを統合管理し，複数のアプリケーションプログラムから共通に統合管理されたデータを利用する方式の検討を行う．また，このセンサデータ統合管理方式の実現可能性を検討するために，プロトタイプシステムを構築し評価する．

従来，このようなデータ統合管理が車両制御システム分野において導入されてこなかった理由として，アプリケーションのそれぞれが単体システムとして開発され，複数システムを連携してコストに見合うだけの性能を発揮できなかったことが挙げられる．また，データを統合して管理するための車載ネットワークの速度

[†] 名古屋大学大学院情報科学研究科附属組込みシステム研究センター，名古屋市
Center for Embedded Computing Systems, Nagoya University, Nagoya-shi, 464–8601 Japan
[††] 同志社大学理工学部情報システムデザイン学科，京田辺市
Department of Information Systems Design, Doshisha University, Kyotanabe-shi, 610–0321 Japan
[*] 現在，（株）オクトパス
※ 本論文はシステム開発論文である．

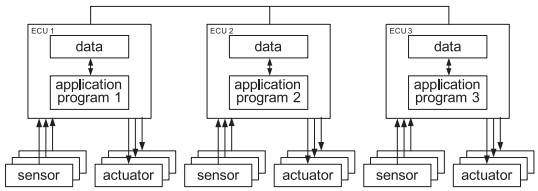

図 1　現行システム構成モデル概要
Fig. 1　Current system overview.

が十分でなかったことも一因である．しかし，様々な車両制御システムが車に実際に搭載され始め，高速な車載ネットワークの登場によりデータの統合管理の可能性や必要性が出てきた．データ統合管理により，複数の車両制御システムにおいて利用されているそれぞれのセンサを統一したインタフェースで共通利用できれば，センサやアプリケーションの追加・削除が容易に行え，また，アプリケーションソフトウェアの再利用性が高まり，システム開発のコスト削減にもつながる可能性がある．

2. センサデータ統合管理

2.1 システム要求

本節では，車両制御システムのデータ統合管理に対する要求事項を実時間性，演算，トランザクションの観点から述べる．

2.1.1 実 時 間 性

人が危険を検知してから行動を起こすまでの時間を空走時間（あるいは反応時間）と呼び，その平均はおよそ 660 ms といわれている [6]．今回，車両制御システムの中で時間制約が厳しく，衝突の危険性に対し動作する運転支援システムにおいて，危険な事象が発生してから車両制御システムが危険だと判断するまでの間が空走時間以内であれば有効であると定義する．この時間を，危険な事象を示すデータが発生してから車両制御システムに到達するまでの遅延時間と考え，その内訳は次のようになる．

（1） センサデバイスによるデータ取得時間

（2） 取得したデータをデータベースへ渡す際にかかるデータ伝送時間

（3） データベースが受け取ったデータを挿入する処理時間

（4） 車両制御システムがデータベースへクエリを発行する際のクエリの送信時間

（5） データベースでクエリに対する必要なデータを処理する時間

（6） データベースから車両制御システムへ必要なデータを返す送信時間

（7） 受け取ったデータを使って危険かどうかの判断を行う処理

この中で，(1) は一般に 100 ms 以内 [7] であることが求められ，この条件を満足させるためには (7) において同等の処理時間であることが求められる．したがって，本研究ではそれぞれの処理時間を 100 ms として定義する．そのため，残りの時間 460 ms 以内にデータの送受信とデータベース内でのデータ処理を完了することを実時間性の要求条件とする．

2.1.2 演　　算

一般にデータベース利用を行うアプリケーションがよく利用する選択・射影・結合・集約の演算は，車載制御システムでも必要となる．以下に車載制御システムでそれぞれの演算の利用例について述べる．

・選択，射影

基本的な問合せである選択，射影に関して，例えば，Surround をセンサにより得られた周辺物体の認識データを格納したテーブルであるとした場合，そのテーブルの中から，現在時刻からさかのぼって t 時間前の物体 ID の位置座標のみを取得する SQL 文によって，車両の軌道計画の設定が可能となる．

Surround(物体 ID，位置座標，測定時刻)

SELECT Surround. 位置座標 FROM Surround WHERE 測定時刻 > (現在時刻 − t)

・結合

衝突危険性に対して動作する車両制御システムでは，センサで得られた認識物体の形状を調べるために，事前に定義された情報（例えば建物）との結び付けを行い，その認識物体に対して，自車の制御方法選択の判断を行う材料としてクエリ結果の利用が想定ができる．Surround テーブルはセンサにより認識した周辺物体の認識テーブルで，Buildings テーブルは地図データとして管理している建物に関するデータのテーブルとする．

Surround(物体 ID，位置座標，測定時刻)

Buildings(建物 ID，位置座標，形状 ID，住所)

SELECT * FROM Surround, Buildings WHERE Surround. 位置座標 = Buildings. 位置座標

・集約

例えば，道路の同一レーン上に存在する車両の台数を検出し，渋滞の程度を計算するために利用するとい

うことが想定できる．これらのテーブルの情報は，車両単体で得られるデータと車車間通信や路車間通信によって得られるデータを蓄積したものとする．

Surround(物体 ID，位置座標，測定時刻，レーン ID)

SELECT レーン ID, SUM(物体 ID) FROM Surround GROUP BY レーン ID;

2.1.3 トランザクション

車両周辺の情報には，逐次変化するデータとほとんど変化がない二つの種類のデータがある．障害物認識によってガードレールや木を認識した場合，その位置情報や特徴情報は比較的時間が経過しても変化することがない．一方前方車を認識した場合，その情報は比較的短い時間で大きく変化する．変化の少ないデータは，保存しておくことで現在の測定値と比較したり，測定そのものを過去のデータで代用するといったことができる．このことから，車載制御システムで扱うデータを管理する場合において，永続的ストレージに保存し，その中で不要なデータは定期的に削除するといったトランザクションが必要になる．

2.2 システム構成

本研究で提案するセンサデータ統合管理方式を実現するためのシステム構成のモデル概略を図 2 に示す．複数のシステムにおいてそれぞれ管理されていたデータを統合管理し，センサを切り離す構成をとる．この構成は，データの管理において一般的にとられる手法である [8]．車速/車輪速センサやステアリングセンサ，加速度/角速度センサなどから得られる車両の運動状況を示すデータは簡単な正規化を行うことで，複数のアプリケーションから共通に利用することが可能である．しかし，ミリ波/レーザレーダ，可視光/赤外線カメラ，超音波センサなどから得られる周辺の物体のデータは多様であり，センサ設置の有無や，センサそれぞれが検知できる範囲や精度も様々である．そこで，周辺の物体のデータを統一的に管理するための方式が必要となる．

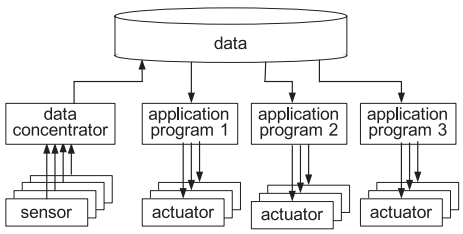

図 2　提案システム構成モデル概要
Fig. 2　Proposed system overview.

2.3 データ管理方式

本研究においては，占有グリッド (occupancy grid) [9]，シーングラフ (scene graph) [10] を用い，車両の運動状況を示すデータも含めたリレーショナルデータとして統合管理を行う．一般的な占有グリッドでは，各グリッドに物体の存在する確率を割り当て，特定の位置に物体が存在するかどうかを判定するのみである．ここでは，ロボット分野で利用されている SLAM [11] の手法を取り入れ，センサの有無や，精度の違いを共通化しつつ，物体の形状，大きさ，向き，代表点の位置などの属性情報を付加し統合するためにシーングラフを利用する．シーングラフにより，占有グリッドの情報を補完することができ，周辺の物体間の相対位置関係によるトリー構造の構成が可能となり，自車両と他の車両や障害物との衝突判定や衝突予測を行うことができる．また，車両，建造物，歩行者などの物体に加え，レーンなどの情報も含めセンサから得られたデータの統合表現が可能となる．これらセンサデータの保持，及び，保持されたデータの取得のために，SQL を利用したデータアクセスインタフェースを定義し，これらの機構をすべて含めて 3D360DB と呼ぶ．

3. システム設計

3.1 システム内部構成

図 3 に本システムの内部構成を示す．アプリケーションは，3D360DB を利用する ECU 上のアプリケーションソフトウェアに該当する．センサから取得し処理を

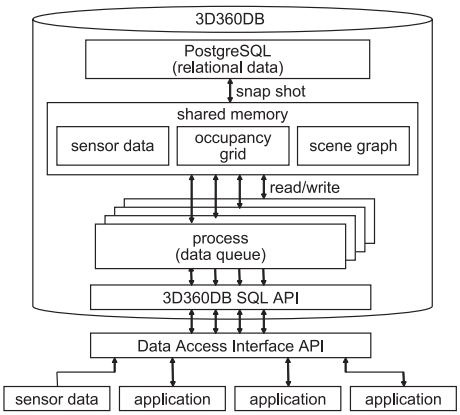

図 3　システム内部構成
Fig. 3　System design architecture.

行ったデータの書込みはセンサデータアプリケーションが実施し，その他のアプリケーションは，3D360DBからのデータの取得を行い，データ処理やアクチュエータ操作を行う．具体的には，各アプリケーションは，データアクセスインタフェースと呼ばれる 3D360DB へアクセスするための関数を利用して SQL コマンドを実行することで 3D360DB からデータ送信受信が可能になる．

3D360DB は，アプリケーションから送信された SQL コマンドを解析し，アプリケーションに対応するプロセスによって，それぞれのコマンドに応じてデータのアクセス先を変化させる．アクセス先は，SQL コマンドによるテーブルの指定に依存して，占有グリッド，シーングラフ，センサデータの三つに分かれる．また，履歴の保存のために，スナップショットを保存するためのプロセスが存在し，占有グリッド，シーングラフ，センサデータから定期的にデータを読み取り，リレーショナルデータで保存するという作業を行う．

3.2 占有グリッド

占有グリッドを示すグリッドマップの各セルには，そのセルを占有する周辺物体の存在確率を保持する．本研究においては，マップサイズを 200 m × 200 m，セルサイズを 0.5 m，セルの値は 0 から 255 の値として，利用する際に確率値に変換する．SQL による統一アクセス実現のため，SQL 文でテーブルに occupancygrid を指定することで，グリッドマップの値の参照・更新が可能となる．

> ・グリッドマップ上の位置 x，y での障害物の確率値の取得
> select cell from occupancygrid where x=xidx and y=yidx;
> xidx，yidx はグリッドマップ上の位置

グリッドマップにおける表現可能な領域には上限があり，車両周辺のすべての位置のグリッドマップを保有することができない．そこで，車両がグリッドマップ範囲外に移動した場合は，グリッドマップのスクロール機能により対応する．これは，図 4 のように，グリッドマップ範囲外に自車が移動した際などに利用することを想定し，スクロール命令を実行することでスクロール範囲外のセルは切捨てを行い，新たに出現したセルは，物体があるかどうか不明なので 0.5 として値を初期化する．

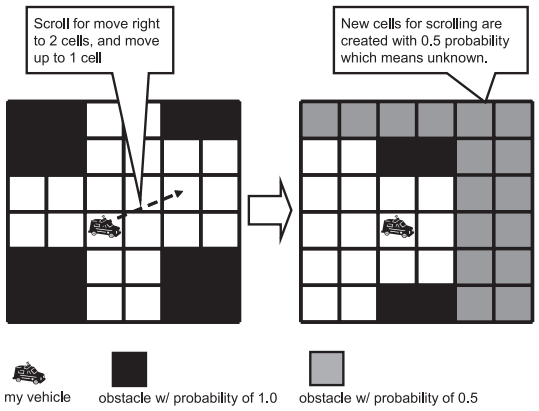

図 4 占有グリッドのスクロール機能
Fig. 4 Scroll of occupancy grid.

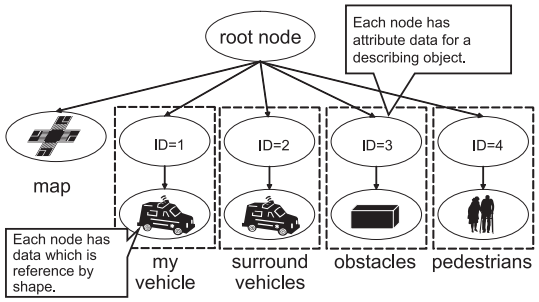

図 5 シーングラフのトリー構造
Fig. 5 Tree structure of scene graph.

また，占有グリッドは車両周辺の障害物に対して統一したデータ表現が可能であり，これを利用して自車の移動可能距離を車両周辺全体の情報から推定する機能も保持する．これは車両が特定の方向に対しどれだけ移動できるかという推定を行う機能で，次の SQL 文を実行することで障害物までの距離を返すものである．

> ・占有グリッドによる移動可能距離推定
> select shiftability(vx, vy, vz) from occupancygrid;
> vx，vy，vz は自車の向き情報

3.3 シーングラフ

シーングラフは，図 5 に記述するように，root ノードを頂点としてその下に物体のデータを表現するノードが連なる構成をとる．それぞれのノードは物体の情報をもち，その情報を二つのノード Transform ノー

ドと Drawable ノードで表現する．Transform ノードは物体の位置，向き，大きさをもち，Drawable ノードは物体の形状をもつ．

SQL による統一アクセスのため，SQL 文によりシーングラフにアクセスし，ノードに対して参照・更新を行うことができるように，ノードの作成，ノードの削除，ノードの移動，ノードの回転の四つの操作を実現できるように設計する．シーングラフにアクセスする際にはテーブル ID として scenegraph_node を指定する構成としている．

・ノードの作成
insert into scenegraph_node (geode_id, nodefile_id, p_x,p_y,p_z,roll,pitch,yaw,width, height,depth,m_time) values;
geode_id:物体 ID，nodefile_id:物体名称，p_x,p_y,p_z:中心位置，roll, pitch, yaw:向き，width:幅，height:高さ，depth:奥行 m_time:測定時刻

3.4 アプリケーション対応プロセス

3D360DB では，リレーショナルモデル，占有グリッド，シーングラフがもつそれぞれのデータに対して SQL 形式でアクセスすることができるようにアプリケーションそれぞれに対応するプロセスとして実現を行う．これらアプリケーション対応プロセスとは，3D360DB を利用するアプリケーションがデータアクセスインタフェースを利用して送信した SQL 命令を解釈した後に行う最初のプロセスである．これは，SQL による統一したデータアクセスを実現するためにあり，三つの車載データの管理方式によるアクセス手段に依存することなく，車両制御システムの開発者は SQL コマンドの利用のみでデータの送受信の要求を行うことを可能にしている．今回の方式では，SQL 形式によりアクセスを統一しており，標準化された手法に基づく操作でき，機能的で使いやすいという利点がある．

3.5 アプリケーション設計

本研究において，センサデータ統合管理方式を実現したシステムの有効性を検証するため，車両追従システム [12] と自動駐車システム [13] の二つの車両制御システムを設計する．

車両追従システムは，車の前方向に自動車が存在した場合，その自動車と一定の車間距離を保ちつつ追従を行う制御を自動で行うシステムである．一定の車間距離を保つために，前方車との車間距離や自車との速度差を必要とし，そのデータから自車の目標速度を計算し，ブレーキやアクセルを操作する．

自動駐車システムは，自車が停止している状態において，目標とする停車位置と停車方向が与えられた場合，その停車位置に合うようにステアリングとトルクを操作するシステムである．目標位置にたどり着くと停車を行う．また駐車している最中に，3D360DB の移動推定機能を利用して車両周辺の障害物と衝突しないかどうかを常にチェックしながら動作をさせている．

二つの制御システムを検証に用意することで，車両追従システムのように自車両と前方車両が高速に動いているときに動作するシステムと，駐車支援システムのように自車両は低速で動き周辺の物体は静止している状況で動作するシステムと双方の検証を目的とした．それぞれ動作するシステムの状況は異なり，利用するデータも共通部分と非共通部分がある．これにより，新たに車両制御システムを追加で実装する際の指針とすることが可能となる．

4. 実　　装

4.1 実装環境

本センサデータ統合管理方式を実現するための実装は，二つの環境を利用して実施する．3D360DB は Linux 上で実装を行い，3D360DB を利用するアプリケーションは，シミュレータである CarSim [14] と MATLAB/Simulink [15] を使い実装を行う．

CarSim は，多様な車両制御システムの制御ロジックを，実車両で評価を行う前に検証することが可能なシミュレーションソフトであり，複数の走行支援システムに利用されたという実績がある [16]．CarSim で再現された環境から得られる入力データは，そのデータの変数名，意味，単位，車両のどこの部位から取得できるかを定めており，シミュレーション実行中に逐次取得することが可能である．

3D360DB とアプリケーションがデータのやり取りをする際には，MATLAB/Simulink 側から C 言語相当のプログラムの実行機能が必要になるが，MATLAB/Simulink の S-Function Builder の機能を用いることで実現する．また 3D360DB が管理するデータは，CarSim から得られるデータをもとにしている．

4.2 3D360DB

本システム内の占有グリッド部の実装は MRPT ラ

イブラリ [17] を利用する．占有グリッドは車両周辺情報をグリッドマップという形式で共有メモリ上に保持する構成をとる．占有グリッドは，固定サイズの 1 枚のグリッドマップを共有メモリ上に作成し，複数のアプリケーションでデータ共有を行うことができるようになっている．そのため各アプリケーション間で排他制御が必要になる．今回，書込み用セマフォ，読出し用セマフォの 2 種類を用意し，書込み時には exclusive lock，読出し時には shard lock を利用することで排他制御を実現した．ロックの対象はグリッドマップ全体としている．シーングラフは，OpenSceneGraph ライブラリ [18] を利用して実装を行う．リレーショナルデータベースは PostgreSQL を利用する．

リレーショナルデータベースでは，車両制御システムがデータに対してアクセスを行うためにスキーマ定義をあらかじめ行っており，必要なデータにアクセスする際には，次のテーブル ID を指定する必要がある．

- platform_globalpose
- platform_velocity
- platform_powertrain_engine
- feature_surround
- feature_lane

platform_globalpose は自車の位置と向きに関する情報として自車の緯度，経度，高度，ロール，ピッチ，ヨーをもち，platform_velocity は自車の速度情報として自車の 3 方向に対応する速度，加速度，角速度，角加速度をもち，platform_powertrain_engine は自車のエンジンに関する情報としてクランク軸の角速度，角加速度，クランク軸の軸出力をもち，feature_surround は車周辺の物体に関する情報として周辺物体の ID，自車との 3 方向に対応する距離，相対速度，幅，高さ，奥行をもち，feature_lane は自車周辺のレーンに関する情報として自車とレーンとの位置偏差とヨー角，ピッチ角，レーン曲率，レーン幅をもっている．SQL によるアクセスを統一化するために，占有グリッドやシーングラフについても同様にテーブル ID を用意し，SQL 文の中でテーブル ID を指定することでアクセス可能とする．

4.3 センサデータ入力モジュール

3D360DB において管理を行うセンサデータの入力は，図 6 に示したセンサデータ入力モジュールが行う．入力モジュールは，CarSim を通して取得したデータを 3D360DB へ入力する役割とし，センサから取得したデータを ECU で処理した後，3D360DB へデータを提供する機能のエミュレータとして動作する．提供するデータは，大きく分類して自車，前方車，レーン，障害物の四つになる．自車データはテーブル ID として platform_globalpose，platform_velocity，platform_powertrain_engine を指定し，それぞれ自車位置，自車速度，エンジンの出力値の入力を行う．前方車データはテーブル ID として feature_surround を指定し，車間距離や速度の入力を行う．レーンはテーブル ID として feature_lane を指定し，自車との位置偏差，レーン曲率の入力を行う．

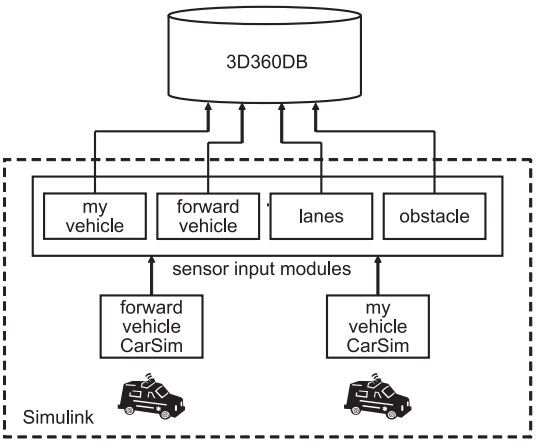

図 6　入力モジュールのデータフロー
Fig. 6　Dataflow of input modules.

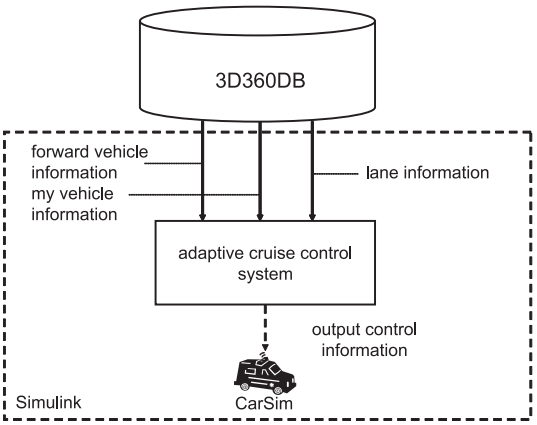

図 7　車両追従システムのデータフロー
Fig. 7　Dataflow of adaptive cruise control.

4.4 アプリケーション設計

車両追従システムのデータフローを図 7 に示す．車両追従システムを動作させるため，3D360DB は実行時，自車や周辺の情報を車載データとして保有し，提

図 8　車両追従システム
Fig. 8　Adaptive cruise controll system.

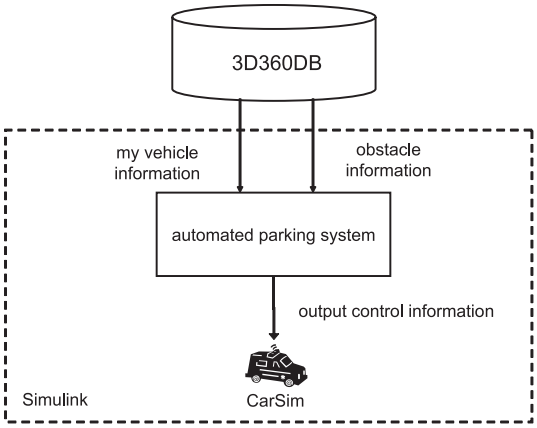

図 9　自動駐車システムのデータフロー
Fig. 9　Dataflow of automated parking system.

供する必要がある．今回の実装においては，CarSim をシミュレータとして利用し，CarSim から得られるデータをセンサから得られたデータと見立てて 3D360DB へ提供を行う ECU の役割を担うソフトウェアとして入力モジュールの実装する．

3D360DB より Simulink モデルで実現した車両追従システムがレーン情報，自車情報，前方車情報のデータを取得し，車間距離と自車速度をもとに，制御すべき目標速度，ステアリング値の計算を行い制御情報を CarSim へ出力する．車両追従システムの表示例を図 8 に示す．

次に，自動駐車システムのデータフローを図 9 に示す．自動駐車システムにおいても，車両追従システムと同様に 3D360DB は実行時，自車や周辺の情報を車載データとして保有し，提供する必要がある．同じく，CarSim をシミュレータとして扱っており，CarSim から得られるデータをセンサから得られたデータと見立てて 3D360DB へ提供を行う ECU の役割を担うソフトウェアとして入力モジュールの実装を行う．

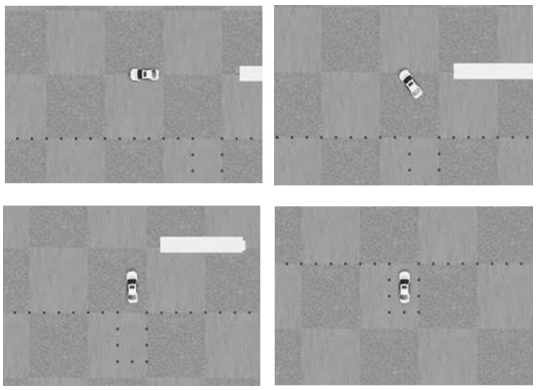

図 10　自動駐車システム
Fig. 10　Automated parking system.

車両追従システムと同様，Simulink モデルで実現した自動駐車システムが，3D360DB から自車情報と障害物情報を取得し動作する．自動駐車システムは起動時，ドライバから駐車位置を指定され，自車位置と駐車位置をもとに軌道を計算し，軌道に従い制御を行う．軌道上を移動する過程で，障害物情報を定期的に取得し，障害物と衝突の危険がある場合には停止を行う．自動駐車システムの表示例を図 10 に示す．

5. 評　　価

5.1 評価環境

性能評価は PC (CPU: PentiumD CoreDuo 2.8 GHz, RAM: 2 GB, OS: Ubuntu 8.04) の環境で実施した．3D360DB と CarSim を利用して作成した車両制御システムの間で，データの送受信にかかったデータ量とクエリ実行の処理時間の測定を行った．データ量の測定は，車両制御システムが 3D360DB を利用する際のデータ量の見積りを目的とし，クエリ実行の処理時間は，3D360DB を利用することで生じるオーバヘッドを調べることで実現性の検証を行うことが目的である．

5.2 車両追従システム

車両追従システムからのデータ取得要求により，3D360DB が SQL コマンドを実行する際の入出力のバイト数と実行時間の計測を行った．また，そのときの入力モジュールからのデータ更新要求で 3D360DB が SQL コマンドを実行した際の入出力のバイト数と実行時間の計測も実施した．それぞれの評価結果を

表 1　車両追従システム実行時のデータサイズ
Table 1　Data size of adaptive cruise control system.

状況	アプリ	データ	実行数	平均入力バイト / 平均出力バイト
1	車両追従システム	前方車	122	92.50 byte / 94.37 byte
2	車両追従システム	自車	183	51.33 byte / 87.64 byte
3	車両追従システム	レーン	61	46.00 byte / 99.21 byte
4	入力モジュール	前方車	61	127.77 byte / 13.00 byte
5	入力モジュール	自車	183	208.81 byte / 13.00 byte
6	入力モジュール	レーン	61	75.21 byte / 13.00 byte

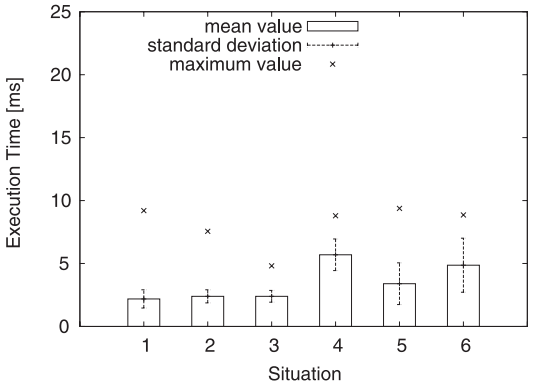

図 11　車両追従システム実行時のクエリ実行時間
Fig. 11　Execution time for query by adaptive cruise control system.

表 1，図 11 に示す．

表 1 において，状況 1 は車両追従システムが前方車情報をデータとして取得する場合であり，この際の SQL の実行数は 122 回，平均入力バイトは 92.50 バイトであり，平均出力バイトは 94.37 バイトである．状況 2 から状況 6 も同様である．図 11 は，表 1 の状況 1 から状況 6 に対応した実行時間の平均値，標準偏差，最大値を示す．

5.3　自動駐車システム

自動駐車システムからのデータ取得要求で，3D360DB が SQL コマンドを実行した際の入出力のバイト数と実行時間の計測を行った．また，そのときの入力モジュールからの書込み要求で 3D360DB が SQL コマンドを実行した際の入出力のバイト数と実行時間の計測も行った．それぞれの評価結果を表 2 と図 12 に示す．

表 2 において，状況 1 は自動駐車システムが自車情報をデータとして取得する場合であり，この際の SQL の実行数は 7218 回，平均入力バイトは 50.00 バイトであり，平均出力バイトは 12.96 バイトである．状況 2 から状況 4 も同様である．図 12 は，表 2 の状況 1 から状況 4 に対応したクエリの実行時間の平均値，標準偏差，最大値を示す．

表 2　自動駐車システム実行時のデータサイズ
Table 2　Data size of automated parking system.

状況	アプリ	データ	実行数	平均入力バイト / 平均出力バイト
1	自動駐車システム	自車	7218	50.00 byte / 112.96 byte
2	自動駐車システム	障害物	2407	74.00 byte / 64.00 byte
3	入力モジュール	自車	7221	219.82 byte / 13.00 byte
4	入力モジュール	障害物	722	90.36 byte / 36.37 byte

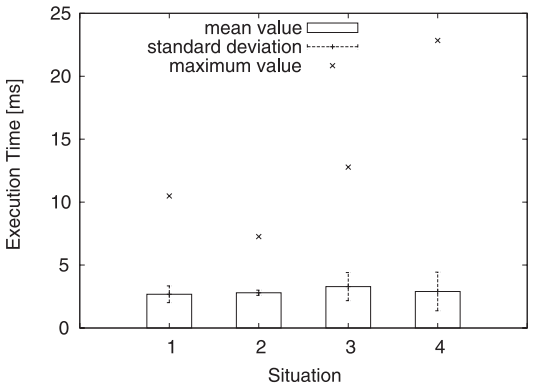

図 12　自動駐車システム実行時のクエリ実行時間
Fig. 12　Execution time for query by automated parking system.

5.4　3D360DB

3D360DB の占有グリッドの機能について，セル値読出し，セル値更新，セル値初期化，グリッドマップスクロール機能のそれぞれの実行時間と入出力のバイト数の測定を行った評価結果を表 3，図 13 に示す．

表 3 において，状況 1 は占有グリッドマップからある特定のセルの値を読み出す場合であり，この際の SQL の実行数は 2500 回，平均入力バイトは 57.00 バイトであり，平均出力バイトは 56.00 バイトである．状況 2 から状況 4 も同様である．図 13 は，表 3 の状況 1 から状況 4 に対応したクエリの実行時間の平均値，標準偏差，最大値を示す．

3D360DB のシーングラフ機能について，ノード属性読出し，ノード移動・回転，ノード作成，ノード削除のそれぞれの実行時間と入出力のバイト数の測定を行った評価結果を表 4，図 14 に示す．

表 4 において，状況 1 はシーングラフに存在してい

表 3　占有グリッドのデータサイズ
Table 3　Data size of occupancy grid.

状況	操作	実行数	平均入力バイト / 平均出力バイト
1	セル値読出し	2500	57.00 byte / 56.00 byte
2	セル値更新	2500	60.00 byte / 13.00 byte
3	セル値初期化	2500	61.00 byte / 13.00 byte
4	スクロール機能実行	800	44.52 byte / 5.00 byte

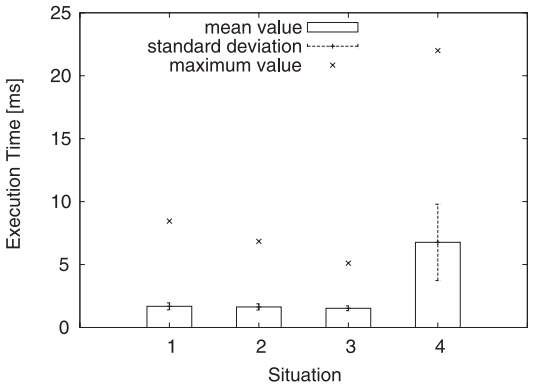

図 13　占有グリッドの実行時間
Fig. 13　Execution time for query occupancy grid.

表 4　シーングラフのデータサイズ
Table 4　Data size of scene graph.

状況	操作	実行数	平均入力バイト / 平均出力バイト
1	ノード属性値読出し	1000	72.89 byte / 138.00 byte
2	ノード移動・回転	1000	73.89 byte / 13.00 byte
3	ノード作成	1000	182.89 byte / 13.00 byte
4	ノード削除	1000	50.89 byte / 13.00 byte

るノードからその属性の値を読み出す場合であり，この際の SQL の実行数は 1000 回，平均入力バイトは 72.89 バイトであり，平均出力バイトは 138.00 バイトである．状況 2 から状況 4 も同様である．図 14 は，表 4 の状況 1 から状況 4 に対応したクエリの実行時間の平均値，標準偏差，最大値を示す．

5.5 考　　察

車両追従システム実行時のクエリ実行時間の評価結果（図 11）と自動駐車システム実行時のクエリ実行時間の評価結果（図 12）では，実行時間の平均と最大の差が大きくなっている．この原因は 3D360DB において実行される SQL 命令を PostgreSQL へ処理依頼する際の，書込み側と読込み側のロック等によるデータ

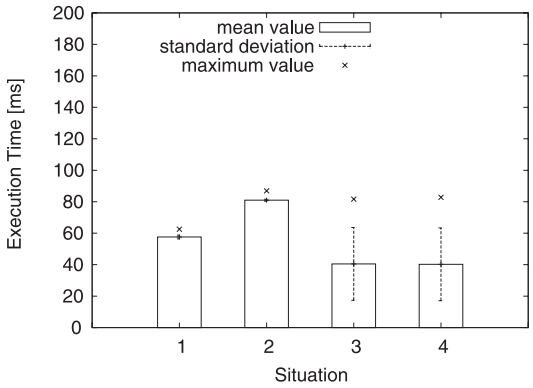

図 14　シーングラフの実行時間
Fig. 14　Execution time for query scene graph.

ベースに起因するものである．

占有グリッドの実行時間の結果（図 13）では，全体的に実行時間の平均と最大の差が大きくなっている．これは占有グリッドに対する SQL 命令を受け取った 3D360DB がその命令文をパースする際にまれに発生するメモリ不足によって発生する遅延である．またスクロール命令では，標準偏差が大きく，実行時間の平均と最大の差も大きくなっている．スクロール命令は現在のグリッドマップのデータの一部と新たにスクロールした領域とを重ね合わせて，新たなグリッドマップのデータを生成する．そのため，X 座標方面にスクロールするか Y 座標方面にスクロールするかで，現在のグリッドマップのデータ（配列）のうち残しておくデータの配置が異なるため，処理時間にばらつきが発生する．

シーングラフの実行時間の結果（図 14）では，insert 処理と delete 処理の標準偏差が大きく，実行時間の平均と最大の差も大きくなっている．これは，プロセス間のシーングラフのデータ共有をファイルを介して行っており，insert 処理，delete 処理はともに最初にそのファイルをインポートし，変更が完了したらエクスポートを行っている．そのため insert や delete によりデータの書込みや削除を行うことでファイルサイズが増減し，それに比例してファイルのインポート処理，エクスポート処理の時間の増減が発生している．

5.6 実 時 間 性

今回の車両追従システム及び自動駐車システムの評価結果が **2.1.1** で定義した実時間性の要求条件に従うかの検討を行う．車両追従システムと自動駐車システムで扱っているそれぞれのデータ項目において，取

得までの遅延時間を計算することで評価を行う．

データ取得の遅延時間は，**2.1.1** で定義した (2)〜(6) までの値の総和によって求める．

ここで車両追従システムが前方車の情報を取得する場合を例にして，データの取得遅延時間を計算する．(2) から (6) までのそれぞれの値は，表 1，図 11 を使い，次のように計算できる．

(2)=状況 4 の平均入力バイト数の伝送時間
(3)=状況 4 のクエリ実行時間の最大値
(4)=状況 1 の平均入力バイト数の伝送時間
(5)=状況 1 のクエリ実行時間の最大値
(6)=状況 1 の平均出力バイト数の伝送時間

ここで (2)，(4)，(6) はネットワーク上でのデータ伝送時間の計算が必要になる．車載ネットワークとして利用される Controller Area Network (CAN) [20] はイベント駆動型の媒体アクセス制御方式のため，厳密に遅延時間を予測することができない．そのため，ここでは帯域の最大値 (1 Mbit/s) の 10%を割り当てるとして遅延に対する許容時間の見積りを行う．また CAN でデータを伝送する場合，1 フレーム当り 8 バイトまでのデータしか送ることができず，8 バイトを超えるデータに対してはフレームに分割して送ることになるため，フレームごとの送信時間の総和が，送受信を行うデータの通信時間となる．また帯域幅 10%の見積りで 1 フレーム当りの送信時間は 1.35 ms となる [21]．

(2) は，状況 4 の平均入力バイト数が 127.77 バイトであるため CAN では 16 フレーム送る必要があるので，その伝送時間は $16 \times 1.35 = 21.6$ ms となる．(3) は図 11 を参照すると 8.80 ms かかることが分かる．(4) は状況 1 の平均入力バイト数が 92.50 バイトであり，12 フレーム送る必要があるので，その伝送時間は $12 \times 1.35 = 16.2$ ms となる．(5) は，図 11 を参照すると 9.20 ms かかることが分かる．(6) は状況 1 の平均出力バイト数が 94.37 バイトであり，12 フレーム送る必要があるので，その伝送時間は $12 \times 1.35 = 16.2$ ms となる．よって (2) から (6) の総和が 72.00 ms であり，前方車情報の取得遅延時間は 72.00 ms かかることが分かる．

以下同様に，それぞれのデータ項目に対し，データ取得の遅延時間の計算を行いその結果を表 5 に記述した．この結果により，それぞれのデータの取得による遅延時間は 460 ms 以内に抑えられているため，本システムは実時間性の要求条件を満たしている．

表 5　データ取得遅延時間の評価
Table 5　Latency.

アプリ	データ項目	遅延時間
車両追従システム	前方車	72.00 ms
車両追従システム	レーン	52.82 ms
車両追従システム	自車	77.69 ms
自動駐車システム	障害物	83.28 ms
自動駐車システム	自車	90.77 ms

5.7 応 用 性

今回の検証に使った車両追従システムと自動駐車システムについて考察し，3D360DB がどのようなアプリケーションに応用できるかについて検討する．車両追従システムの実行時に取得を行っているデータは，前方車，レーン，自車の 3 種類になる．自車を除くと二つの車両周辺の情報を扱っている．自動駐車システムは自車と障害物の二つのデータを利用しており，自車を除くと一つのみ車両周辺の情報を扱っていることになる．扱う車両周辺のデータが増えた場合，それぞれのデータに対し時間制約が必要であると考えると，データベースへの実時間性の要求もその数に比例して厳しくなると考えられる．例えば，周辺車両，歩行者，障害物に対し自車が衝突するかどうかを判断し，危険であるなら警告を行うシステムであるならば，3 種類のデータそれぞれに時間制約が必要となる．このように考えた場合，今回の検証によって，2 種類の車両周辺のデータを扱う車両制御システムであれば応用ができると考える．

6. 関 連 研 究

6.1 PReVENT

PReVENT [22] は車両の衝突防止や道路上の安全を確保するするため European Commission により設立された研究プロジェクトである．このプロジェクトのサブプロジェクトである ProFusion2 [23] は，複数のセンサの融合により障害物や外部の環境の認識の精度を上げることを目的としたプラットホームの研究を行っている．センサフュージョンを実現するための一手法として，一般的な占有グリッド技術を用いており，センサの追加や削除，センサの種類の変更に柔軟に対応できるものではない．また，本研究で実施しているようなデータに対しての統一的なアクセス方法に関して，具体的な実現方法を示しているものではない．

6.2 ITS-Safety2010

国土交通省自動車交通局が進める安全運転支援シス

テムを実現するためのプロジェクト [24] であり，衝突被害軽減ブレーキ，前方車両追従などのアプリケーション技術の確立を目指すものである．本プロジェクトにおいては，それぞれのシステムは独立して開発され，機能，性能を確認することが目的であり，それぞれのシステムを融合し効率的な開発を目指すものではない．

6.3 ストリーム処理

近年，センサデータのような継続的に生成される複数の情報源を対象に，問合せ要求を効率的に処理する連続的問合せ (Continous Query) の研究が行われている [25], [26]．また，筆者らは，ストリームプロセッシングによる車載統合制御システムのための分散型センサデータ処理機構の検討も行っている [27]．連続的問合せでは，従来のデータベースにあるトランザクション処理によってデータを永続的ストレージに保存するということを想定せず，メインメモリ上でストリーム処理をしてデータの配信を行うことで効率化を実現している．本研究では，車両制御システムが扱うデータが，ガードレールや白線のように車両周辺で認識できる半永久的に変化しないようなデータも対象としているため，古いデータを永続的ストレージに保存し再利用することを想定してトランザクション処理を採用した．

7. むすび

現在，複数の種類の車両制御システムが登場してきている．これらのシステムにおいては，センサデータを個別に管理し処理を行っている．そのため，異なるセンサを利用したり，新規のセンサを追加したりする場合は，アプリケーションプログラムを変更する必要が生じ，また，多種多様な複数システムにおいて，それぞれの設計・開発を統合することが困難となる．本研究では，システムからセンサ部を切り離し，確率を利用した占有グリッドにおいてセンサから得られたデータをシーングラフ併用で統合管理し，複数のアプリケーションプログラムから共通に統合管理されたデータを利用する方式の検討を行った．この方式を利用したシステムの設計，実装を行い，シミュレータを利用し，車両追従システム，自動駐車システムの 2 種類の車両制御システムを構築し，評価を行い，許容可能なオーバヘッドで本システムの実現可能性を示すことができた．

今回は，センサデータ統合管理方式の検討が目的であるため，汎用 PC 上を利用し SQL データベースによりセンサデータを集中管理する手法をとった．しかし，実際の車載における実装環境においては，各 ECU が分散してデータを管理する方式が現実的である．今後の課題として，分散データ管理及びデータ処理も含め，実際の組込み環境での設計検討を行っていく予定である．また，占有グリッドにおける複数の確率値の計算方法による精度検討も必要となると考えている．加えて今回の検討範囲では，3 種類以上の車両周辺のデータを扱う車両制御システム，例えば 3 台以上の周辺車両を常に監視し衝突危険の警告を行うことができるシステムについては未検討であり，今後の課題としたい．

謝辞　本研究を進めるにあたり，貴重な御意見を頂いたトヨタ自動車（株）制御ソフトウェア開発部の方々をはじめ，御協力下さった名古屋大学大学院情報科学研究科附属組込みシステム研究センターのメンバ各位に厚く御礼申し上げます．

文　献

[1] 浅沼信吉，加世山秀樹，“安全運転支援のための車両予知・予測技術のとりまく状況，” 国際交通安全学会誌，vol.31, no.1, pp.56–61, 2006.

[2] 西垣戸貴臣，大塚裕史，坂本博史，大辻信也，“予防安全の高度化を実現するセンサーフュージョン技術，” 日立評論，vol.89, no.8, pp.72–75, 2007.

[3] W.D. Jones, “Keeping cars from crashing,” IEEE Spectr., vol.38, no.9, pp.40–45, 2001.

[4] 藤田浩一，宇佐見祐之，山田幸則，所　節夫，“衝突危険性のセンシング技術，” 自動車技術，vol.61, no.2, pp.62–67, 2007.

[5] 国土交通省 ASV 推進検討会，第 4 期 ASV 推進計画パンフレット，“ASV，それは交通事故のない社会への架け橋，” 2006.

[6] 自動車技術専門委員会，“JISD0802 自動車–前方車両衝突警報装置–性能要求事項及び試験手順，” 日本工業標準化委員会，2002.

[7] 大杉啓治，宮内邦宏，古居信之，宮越博規，“ACC システム用スキャン式レーザレーダの開発，” デンソーテクニカルレビュー，vol.6, no.1, pp.43–47, 2001.

[8] R. Ramakrishnan and J. Gehrke, Database Management Systems, McGraw-Hill, Singapore, 2002.

[9] A. Elfes, “Sonar-based real-world mapping and navigation,” Int. J. Robotics and Automat, vol.3, no.3, pp.249–265, 1987.

[10] Avi, Scenegraphs: Past, Present, and Future, RealityPrime, 入手先<http://www.realityprime.com/articles/scenegraphs-past-present-and-future> (参照 2009-09-15).

[11] H. Choset and K. Nagatani, “Topological simultaneous localization and mapping (slam) toward exact lo-

calization without explicit localization," IEEE Trans. Robot. Autom., vol.17, no.2, pp.125–137, 2001.
[12] 飯島徹也，東又 章，奥田次郎，浅田哲也，橋詰武徳，溝口和貴，"車間自動制御システムの開発，" 自動車技術，vol.53, no.11, pp.98–103, 1999.
[13] 大前 学，橋本尚久，清水 浩，藤岡健彦，"駐車場を有する構内における自動車の自動運転の運動制御に関する研究，" 自動車技術，vol.35, no.3, pp.235–240, 2004.
[14] Virtual Mechanics Corporation，車両運動シミュレーションソフト，CarSim:車両運動モデル，入手先<http://carsim.jp/category/1275944.html> (参照 2009-09-15).
[15] CYBERNET SYSTEMS CO, SimulinkR 7，サイバネットシステム，入手先<http://www.cybernet.co.jp/matlab/products/product_listing/simulink/index.shtml> (参照 2009-09-15).
[16] Virtual Mechanics Corporation，車両運動シミュレーションソフト，トピックス：バーチャルメカトロニクス，入手先<http://carsim.jp/category/1286799.htmll> (参照 2010-02-22).
[17] The Machine Perception and Intelligent Robotics Lab University of Malaga, The Mobile Robot Programming Toolkit (MRPT), Main P age - MRPT, 入手先<http://babel.isa.uma.es/mrpt/index.php/Main_Page> (参照 2009-09-15).
[18] OSG Community, OpenSceneGraph, osg，入手先<http://www.openscenegraph.org/projects/osg> (参照 2009-09-15).
[19] VMWare, <http://www.vmware.com> (参照 2009-09-15).
[20] International Organization for Standardization, "Controller Area Network (CAN)—Part 1: Data link layer and physical signaling, ISO 11898-1:2003," International Organization for Standardization, 2003.
[21] 飯山真一，高田広章，"システム構成を考慮した CAN の最大遅れ時間解析手法，" 情処学論，vol.45, no.SIG1(ACS 4), pp.66–76, 2004.
[22] ERTICO, PReVENT, PReVENT : Home，入手先<http://www.prevent-ip.org/download/Publications/Profusion_E-Journal_volume_2_Final.pdf> (参照 2009-09-15).
[23] PReVENT:: ProFusion2, 入手先<http://www.prevent.-ip.org/en/prevent_subprojects/horizontal_activities/profusion2/> (参照 2009-09-15).
[24] 国土交通省報道発表資料，"ITS による安全運転支援システムに係る公開デモンストレーション等の実施について，" 入手先<http://www.mlit.go.jp/report/press/jidosha07_hh_000019.html> (参照 2009-09-15).
[25] D.J. Abadi, Y. Ahmad, M. Balazinska, U. Cetintemel, M. Cherniack, J.H. Hwang, W. Lindner, A. Maskey, A. Rasin, E. Ryvkina, N. Tatbul, Y. Xing, and S. Zdonik, "The design of the borealis stream processing engine," Proc. CIDR, 2005.
[26] 渡辺陽介，北川博之，"連続的問合せに対する複数問合せ最適化手法，" 信学論（D-I），vol.J87-D-I, no.10, pp.873–886, Oct. 2004.
[27] 山田真大，鎌田浩典，佐藤健哉，手嶋茂晴，高田広章，"ストリームプロセッシングによる車載統合制御システムのための分散型センサデータ処理機構の構築，" 情処学高度交通システム研究会研報，vol.2009, no.24, pp.79–85, 2009.

（平成 21 年 11 月 10 日受付，22 年 3 月 4 日再受付）

山田　真大

名古屋大学大学院情報科学研究科附属組込みシステム研究センター研究員．2006 東京理科大学大学院理工学研究科情報科学専攻修士課程了．2008 より現職．

鎌田　浩典

（株）オクトパス所属．2008 まで名古屋大学大学院情報科学研究科附属組込みシステム研究センター研究員．1997 京都大学大学院工学研究科情報工学専攻修士課程了．

佐藤　健哉　（正員）

同志社大学工学部情報システムデザイン学科准教授．1986 大阪大学大学院工学研究科電子工学専攻修士課程了．同年住友電気工業情報電子研究所入社．1991〜1994 スタンフォード大学計算機科学科客員研究員．2000 奈良先端科学技術大学院大学情報科学研究科博士後期課程了．米国 AMI-C, Inc. Chief Technologist を経て，2004 より現職．2008 から名古屋大学大学院情報科学研究科附属組込みシステム研究センター特任准教授兼務．博士（工学）．

手嶋　茂晴

京都大学大学院工学研究科情報工学専攻修士課程，名古屋大学大学院工学研究科情報工学専攻博士後期課程了．博士（工学），1986（株）豊田中央研究所入社，現在に至る．2006〜2009 名古屋大学大学院情報科学研究科附属組込みシステム研究センター特任教授/ディレクターなど歴任．

高田　広章　（正員）

名古屋大学大学院情報科学研究科情報システム学専攻教授．1988 東京大学大学院理学系研究科情報科学専攻修士課程了．同専攻助手，豊橋技術科学大学情報工学系助教授等を経て，2003 より現職．2006 より大学院情報科学研究科附属組込みシステム研究センター長を兼務．リアルタイム OS，リアルタイムスケジューリング理論，組込みシステム開発技術等の研究に従事．オープンソースの ITRON 仕様 OS 等を開発する TOPPERS プロジェクトを主宰．博士（理学）．IEEE，ACM，情報処理学会，日本ソフトウェア科学会各会員．