Vol.2013-DBS-158 No.30
2013/11/26

車両センサーデータを蓄積・活用するための データベースシステムの提案

疋田敏朗[†1a] 堀口賢司[†1a] 栗原慶典[†1a] 三宮千尋[†1a] 那和一成[†1a]

概要 近年の情報処理技術の発達により、自動車内部で利用されている車両制御用のセンサーデータを蓄積して、情報活用を行うことが可能になりつつある。車両制御用のセンサーデータは時系列の数値データであり、列方向への強いアクセスと複数のサンプル周期をもつデータ系列の結合処理が必要であるという特性を持つ。本稿では試験的に取得した車両センサーデータを利用して、従来のデータベースシステムに時系列で生成されるセンサーのデータを蓄積し、幾つかのバッチ処理とオンライン処理を行うことにより性能を評価した。その結果、車両データを高速で効率良く処理を行うためには、現在のデータベースシステムの機能が不足していることが判明した。その実験結果に基づき、大量の車両制御用センサーデータを蓄積・情報処理するために必要なデータ処理システムに関する提案を行った。

Database and Storage System Proposal for Vehicular Sensor Data

Toshiro HIKITA[†1a] Kenji HORIGUCHI[†1a] Keisuke KURIHARA[†1a]
Chihiro SANNOMIYA[†1a] Kazunari NAWA[†1a]

Abstract Recently new information processing systems have been proposed for vehicular sensor data, which have been used to control inside vehicles. Those vehicular sensor data are mainly numeric time series data, which tend to access column oriented. Also handling vehicular data requires a join function to merge multi tables with different sampling rates. Those characteristics are new challenge to existing database systems.
In this paper, we have described characteristics of vehicular sensor data of Control Area Network (CAN) Bus. Then, have discussed benchmark of existing database systems about vehicular sample applications and vehicular sensor data. Finally, we have proposed an architecture of large scale database and storage system for vehicular sensor data.

1. はじめに

近年の情報処理技術の発達により、自動車によって生成されたデータを利用することが可能になりつつある。テレマティックスと呼ばれる自動車のナビゲーションユニットとセンターとの間での通信を活用した情報提供サービスのデータを集約することで、渋滞情報や交通情報の提供、急ブレーキ地点の解析による危険個所の特定などが実際に行われており、今後もより多くのサービスが提供されることが見込まれている。

一方で、自動車内部においても車両制御のために様々なセンサーのデータを活用している。近頃の自動車には数十個以上の Electronic Control Unit（ECU）と呼ばれる制御用のコンピュータが搭載されていることが知られており、これらの ECU の一部は Control Area Network（CAN）[1]と呼ばれる制御用ネットワークで交互に接続されている。

CAN は ISO11898 で定義される通信ネットワーク規格であり、最も広く使われている CAN Specification version2.0 PART B （CAN2.0B）規格では、最大 1.0Mbps で各デバイスを共有バスで接続することができる。図 1 に CAN バスの接続例を示す。近年のストレージならびに情報処理技術の向上により、研究レベルでは、CAN バスを利用した車両制御用のセンサーデータの集約が可能になりつつあり、従来あまり着目されてこなかった車両制御用のセンサーデータの情報活用を行うことが可能になりつつある。

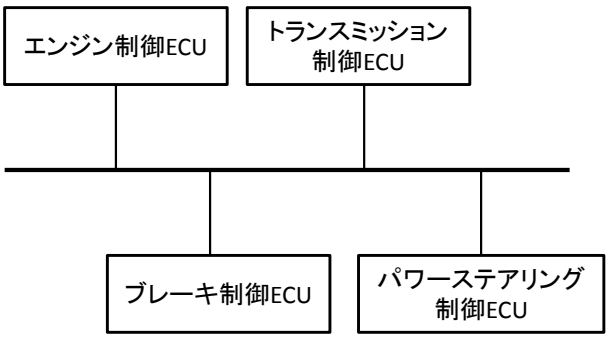

図 1　CAN バスの接続例

本稿ではこれらの ECU から発生する車両センサーデータの情報処理を効率的に行うシステムを構築することを目的とした DB システムについて検討を行った。まず 2 章では、車両制御データの特性について説明を行う。3 章では、

†1 (株)トヨタ IT 開発センター
TOYOTA InfoTechnology Center Co, Ltd.

a) {hikita,k-horiguchi,ke-kurihara,chi-sannomiya,nawa}@jp.toyota-itc.com

車両データの解析システムのアプリケーションを利用することを前提に、既存のデータベースシステム上に車両制御用のデータを蓄積した場合の処理動作を検証する。4 章では、動作検証を踏まえて、車両制御データの処理に必要なストレージシステムの要件について提案を行うこととする。

2. 車両センサーデータの特性

本章では車両制御用のセンサーデータの特性について検討を行う。

表 1 に車両制御用のデータの例を示す。左から時刻（フレーム番号）、緯度、経度、速度、アクセル開度、ハンドル角を示しており、それぞれ、異なった周期で発生するセンサーデータを、約 30fps で刻まれる時刻を基準に結合したものである。これらのデータは、それぞれの時刻に対応したセンサーの出力値を意味しているため、センサーごとの内容を理解するためには図 2 に示すように、テーブルをカラム方向に読んで解析する必要がある。

表 1　車両センサーデータの例

フレーム番号（時刻）	緯度[1/100度]	経度[1/100度]	速度[km/h]	アクセル開度率[%]	ハンドル角[deg]
00:09:01:12	3540.42441	13945.13333	57.5	38	12
00:09:01:13	3540.42441	13945.13333	57.6	37.5	12
00:09:01:14	3540.42441	13945.13333	57.6	37.5	12
00:09:01:15	3540.42441	13945.13333	57.7	37	10.5
00:09:01:16	3540.42441	13945.13333	57.7	37	10.5
00:09:01:17	3540.42441	13945.13333	57.8	36.5	10.5
00:09:01:18	3540.42441	13945.13333	57.8	36.5	10.5
00:09:01:19	3540.42623	13945.13196	57.9	36.5	10.5
00:09:01:20	3540.42623	13945.13196	58	36.5	10.5
00:09:01:21	3540.42623	13945.13196	58.3	36.5	10.5
00:09:01:22	3540.42623	13945.13196	58.3	36.5	10.5
00:09:01:23	3540.42623	13945.13196	58.4	36.5	10.5
00:09:01:24	3540.42623	13945.13196	58.4	36.5	10.5
00:09:01:25	3540.42812	13945.13059	58.4	37	10.5
00:09:01:26	3540.42812	13945.13059	58.7	37.5	10.5
00:09:01:27	3540.42812	13945.13059	58.7	38	10.5
00:09:01:28	3540.42812	13945.13059	58.7	38	10.5
00:09:01:29	3540.42812	13945.13059	58.9	40	10.5
00:09:02:00	3540.42812	13945.13059	59	42	10.5
00:09:02:01	3540.42812	13945.13059	59	42	10.5
00:09:02:02	3540.42812	13945.13059	59.3	44	9
00:09:02:03	3540.42988	13945.12916	59.3	44	9
00:09:02:04	3540.42988	13945.12916	59.2	44.5	7.5
00:09:02:05	3540.42988	13945.12916	59.2	44.5	7.5
00:09:02:06	3540.42988	13945.12916	59.2	44.5	7.5
00:09:02:07	3540.42988	13945.12916	59.6	44.5	7.5

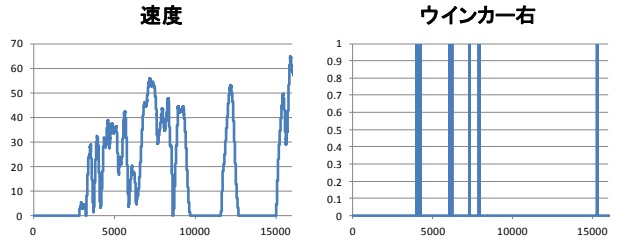

図 2　カラム方向に抽出したデータのプロット例

これらのデータは、取りうる値の数で分類することができる。図 3 に 1 走行分のセンサーデータから、それぞれのカラムが取得した値の数を示す。

このグラフから、「アクセル開度率」、「速度」といった多くの値をとる項目と、「ウィンカー右」、「GPS ステータス」のように非常に限られた種類の値しかとらない項目とに分かれることがわかる。また、カラムの内容と照らし合わせることで、前者はセンサーの出力そのものの数値データ、後者はスイッチで切り替える操作状態を保持するフラグデータであることがわかる。

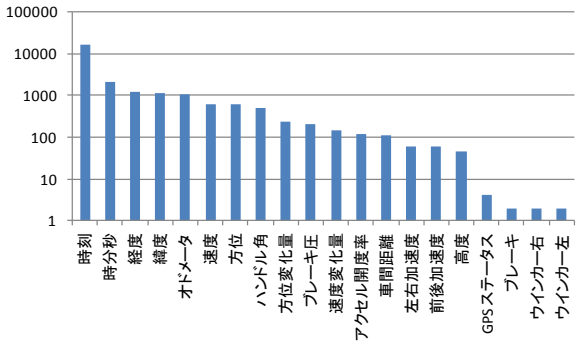

図 3　各データが取り得る値の数

次に、データの種類ごとにその特性を述べる。

まず、数値データの特性であるが、これらは ECU の中で処理をされているセンサーのデータであり、主に時系列データとしてその変動を確認するために用いられる。

この数値データの特徴として、カラムごとに時間方向のデータの推移をみると数値にあまり変化がないことがあげられる。緯度の 1 秒は約 30m 程度であるため、自動車が 100km/h （30m/s）で走行しても 100ms 毎にサンプリングする場合、0.1 秒程度しか変動しない。これらの数値は基本的に時間方向に対する物理値をサンプリングしたものであるため、1 サンプルごとの数値変動量は、値自体に比べて小さく、サンプル間の差分は一定の範囲内に収まりやすい。

図 4 に実際の車両数値データの例として、速度値の分布と差分値の分布を示す。0km/h 付近のデータ集中は、待機時のアイドリング状態を指す。このアイドリング状態を除くと、走行速度は 0-60km/h の範囲に分布しており、差分値は±1.5km/h の範囲に収まっていることがわかる。

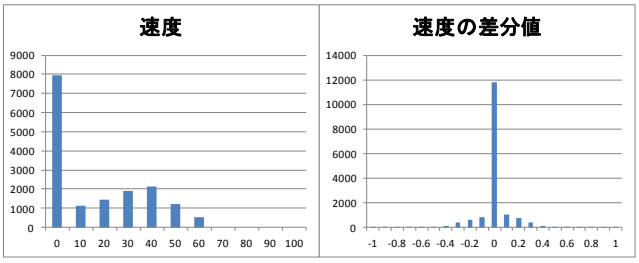

図 4　車速の値と差分値の分布

一方、フラグデータは、前述したように、ECU でのスイッチの ON/OFF や燃料の種類情報などで定義されるものである。これらのフラグ値は、一定のビットで事前に定義した情報であることを示す。

図 5 は、ウィンカー動作時の値の変化を示したもので、

ウィンカーの点滅を指示するフラグが、0 と 1 の間で断続的に変化する様子がわかる。

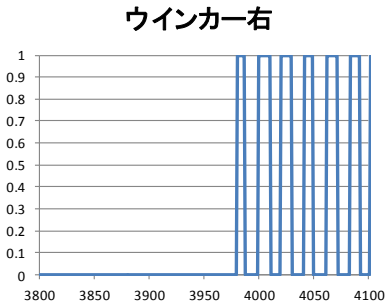

図 5　フラグデータの変化の例

これまでに述べたように、車両センサーデータは、カラム方向で連続的に変化する数値と、断続的に変化するフラグデータの集合体であり、この様なデータを効率的に扱う事のできる DB システムが求められる。

3. 既存 DB システムを用いた車両センサーデータの動作検証

3.1 実験内容

先に紹介した車両センサーデータを処理することを想定し、既存の DB システムの処理性能を検討するために、ベンチマーク評価を行うこととした。今回採用したベンチマーク方法をデータベースとデータ処理の観点で記載する。

3.1.1　データベース

評価対象としたデータベースを次に記載する。従来側 DB として、一般的な RDBMS である MySQL を対象として選択し、並列向きカラム指向ストアとして、並列処理に長所を有する Hadoop + HBase を選定した。さらに、カラム指向に適した市販 DB をベンチマーク対象として選定した。

3.1.2　データ処理

実際の車両から取得した車両センサーデータを元に、車両センサーデータ処理の例題として、実際のアプリケーションで利用する以下の 5 種類の処理を準備し、これらの処理にかかる時間を計測することで、既存の DB システムの性能を評価した。

a)　データロード処理
b)　複雑な JOIN 処理
c)　シーン抽出
d)　サマリー作成
e)　データ抽出

以下に処理の詳細を記載する。

a)　データロード処理

データロード処理では、CSV で保存されているデータをシステムに登録する時間を計測する。本実験では、圧縮を有効とし、各 DB での圧縮方式は、それぞれに、最も有効と思われる方式を採用した。

b)　複雑な JOIN 処理

複雑な JOIN 処理は、図 6 に示すように、サンプリング周波数が異なるマスターテーブル（周波数が高い）とサブテーブル（周波数が低い）の内容を結合することとした。ただし、マスターとサブのそれぞれのテーブルは、異なるサンプリング間隔でデータを取得しており、両方のテーブルで同一のキー（時刻）が存在することはほとんどない。そのため、マスターテーブルの時刻キーの前後一定の時間範囲内（今回は 32msec）以内にサブテーブルのレコードが存在する場合は、そのレコードを JOIN するという方法を採用した。

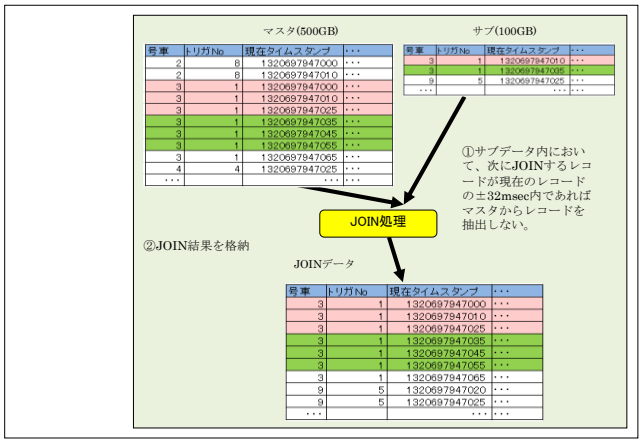

図 6　複雑な JOIN 処理

上記の JOIN 処理は今回の対象 DB では一つの指示として実行することは出来ないため、従来型 DB とカラム指向 DB においては、本実験では、図 7 に示すように、UNION ALL 指示で取得したデータを元に、Java で JOIN 処理を行い、再度 INSERT するという処理を行った。

```
SELECT '2' AS data_type, * FROM tbl_master
UNION ALL   SELECT '1' AS data_type, * FROM tbl_sub ORDER BY gousya, triggerno, ctimestamp, data_type;
⇒取得したデータをJavaで受け取り、JOIN処理を行ってjoinテーブルへINSERTする
```

図 7　複雑な JOIN のサンプルクエリ

c)　シーン抽出

シーン抽出では、図 8 に示すように、特定のカラムについて、あらかじめ定めた閾値（例えば、車速>=40km/h）を超えたデータから、別に定めた閾値（車速<20km/h）を超える範囲までを特定シーンとして抽出する処理を実行した。

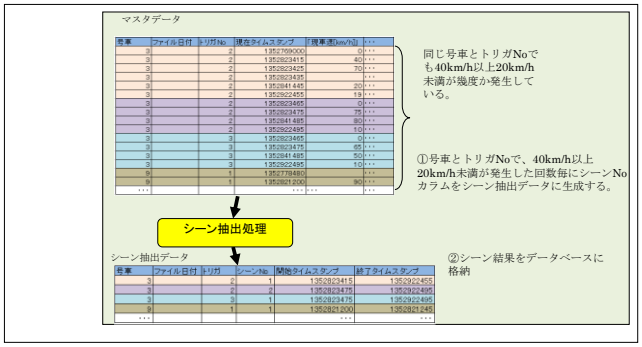

図 8　シーン抽出

また、図 9 に今回使用した代表的なクエリを記載する。

```
SELECT 時刻, 車速 FROM tbl_master WHERE gousya = 'u' AND triggerno = '020' AND
 ctimestamp >= 1315252800000 AND ctimestamp <= 1315407599999 ORDER BY ctimestamp;
⇒取得したデータをJavaで受け取り、該当する閾値の前後のレコードのみをシーンテーブルへINSERTする
```

図 9　シーン抽出のサンプルクエリ

d)　サマリー作成

サマリー作成では、各カラムの数値について、日単位にセンサーデータ 9 カラムの平均、最大と最小を計算して出力する処理を実行した。図 10 に代表的なクエリを記載する。

```
INSERT INTO tbl_summary
 SELECT gousya, triggerno, year, month, day, ROUND(AVG(data_x), 3.0), MAX(data_x), MIN(data_x) ～ FROM tbl_master
GROUP BY gousya, triggerno, year, month, day ORDER BY gousya, triggerno, year, month, day;
```

図 10　サマリー作成のサンプルクエリ

e)　データ抽出

データ抽出処理では、特定レコード指定と時間範囲指定による抽出を行った。今回は、時間範囲を 1 時間、1 日、1 ヶ月、抽出カラム数を 2、4、8 と変化させて実験を行った。図 11 に代表的なクエリを記載する。

```
SELECT cdatetime, data_x1, data_x2 FROM tbl_master WHERE gousya = 'u' AND triggerno = '020' AND
 ctimestamp >= 1315252800000 AND ctimestamp <= 1315407599999 ORDER BY ctimestamp;
```

図 11　データ抽出処理のサンプルクエリ

3.1.3　実験に用いたデータと評価環境

本実験に使用したデータは、取得データを正規化したマスターデータ（420GB）、サブデータ（80GB）の合計 500GB である。実験データのスキーマ例を表 2 に示す。

表 2　実験データのスキーマ例

カラム名	型	説明
gousya	VARCHAR(255)	号車
fdate	VARCHAR(20)	ファイル日付
triggerno	VARCHAR(20)	トリガNo
startdate	VARCHAR(20)	開始日時
ctimestamp	INTEGER	現在タイムスタンプ
cdatetime	DATETIME	現在日時
year	VARCHAR(4)	年
month	VARCHAR(2)	月
day	VARCHAR(2)	日
time	FLOAT	Time
data_1	FLOAT	緯度
data_2	FLOAT	経度
data_3	VARCHAR(20)	記録日時
data_4	FLOAT	センサーデータ
…	…	…
data_279	FLOAT	センサーデータ

評価実験は、表 3 に示すサーバ構成で行った。サーバは 1、2、4 台のクラスタ構成とし、従来型 DB に関しては、1 台での計測、並列カラム指向ストアとカラム指向データベースは 2、4 台での計測とした。

ディスク領域については、データサイズと HDD のディスクサイズの関係から、カラム指向の DB では RAID5、従来型 DB と並列カラム指向ストアについては RAID0 で評価を実施した。

表 3　サーバ構成

CPU	Xeon E5-2690 2.9GHz　8core × 2
Memory	128GB
Disk	300GB(10000rpm) × 6
Network	10Gbit
OS	RHEL6.2

3.2　結果

実験結果のまとめを表 4 に示す。

従来型 DB では、データロード処理とサマリー処理で 2 時間以上の時間を費やしており、複雑な JOIN 処理やシーン抽出は多大なる時間がかかることが想定されたため実験を見合わせた。また、市販のカラム指向型 DB においても、複雑な JOIN 処理は 10 時間以上の時間を要することが判明したため、実験を途中で終了した。しかしながら、それ以外の処理についてはカラム指向の特徴を生かして高速に処理を行うことができることが判明した。並列カラム指向ストアでは、データロードに 1 時間前後の時間を要したが、それ以外の処理については一定時間内で処理を完了させることが判明し。

表 4　実験結果まとめ

システム構成		1node	2node		4node	
		従来型DB	並列カラム指向ストア	カラム指向DB	並列カラム指向ストア	カラム指向DB
データロード	-	07:17:57.281	01:31:15.881	00:44:26.453	00:56:51.073	00:26:41.769
バッチ処理	複雑なJOIN	未実施(※2)	02:37:07.657	測定不能(※1)	02:22:58.338	測定不能(※1)
	シーン抽出	未実施(※2)	00:21:58.070	00:13:43.831	00:11:43.950	00:12:51.545
	サマリー	02:29:28.062	00:07:43.089	00:00:09.960	00:03:48.254	00:00:06.323
オンライン処理	検索	00:00:14.217	00:00:01.400	00:00:05.374	00:00:01.394	00:00:00.438

※1「測定不能」項目は10h以上経過しても終了しなかった項目
※2「未実施」は測定不能になると考え実施しなかった項目

3.3 考察

3.3.1 概要

今回の実験について考察する際には、データアクセスが列方向に集中しており、行方向のアクセスを前提とする従来型 DB である MySQL には不利な条件であったことを念頭に置く必要がある。特にデータ列のうち、ごく一部のカラムデータのみを高速に抜き出すという処理を行うことは、全カラムのデータを抜き出すことが前提の行指向 DB には不向きな処理であるため、他のシステムが秒～分単位で処理する、サマリー処理を取り上げてみても 2 時間以上という時間がかかる原因になったと考えられる。

次に車両センサーデータに対する列方向アクセスの効果だが、これについては従来型 DB との比較を考えるとカラム指向 DB の性能は高いと判断することができる。

3.3.2 詳細

複雑な JOIN に対しては、UNION ALL 指示に基づき、外部プログラムで処理した後に、INSERT と呼ばれる非常に速度が低下する処理が入ることが並列カラム指向ストア以外のシステムにとっては不利に働くこととなったと考えられる。それを回避するためには、特定のキー同士ではなく、一定範囲内での最寄りのキーを保持して、結合できるようなインデックス機構を保有すれば、この処理は高速化できると考えられる。並列カラム指向ストアである HBase については、Hadoop を利用して、MapReduce 処理を行うことで、このようなデータの分散結合演算を効率的に行うことができたと考えられる。また INSERT 処理については、一般的な DB システムでは何らかの形でレコードをロックする必要があるが、今回の実験や車両データの活用用途では、データの一貫性保持が必要とはならないことが多いため、ロック機構が存在する分だけ速度が低下することになりかねないと考える。

今回の実験では、市販の列指向 DB の並列性による速度向上の利点を判断することはできなかったが、並列カラム指向ストアについては、DB 分散の効果が確認できることが分かった。ただし、node 数は極小ともいえる台数であり、より大規模なクラスタ構成で実験を行うことによる性能評価が必要であると考える。

また、今回実施したデータ抽出に関しては、Web からの表示リクエストを想定した実験タスクであるが、実行時間は 2 秒以内となり、実用に耐えうる時間であると判断する。しかしながら、データ量が増えた場合の速度低下についての推察が必要である。

4. 車両センサーデータを活用する DB システムの提案

4.1 概要

ここまでの実験を通して、車両センサーデータの処理に関しては、カラム指向でのデータ処理が必要であるほか、一定範囲内で最も近いキーを保持する機構などが必要であることが分かった。これらの結果を元に、本章では、車両センサーデータを処理するためのストレージシステムの要件を検討していくこととする。

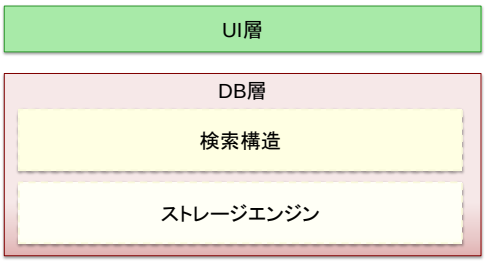

図 12　車両データ蓄積システムの構成

車両データ蓄積システムの全体構成を図 12 に示す。主としてユーザ側の要件である UI 層とデータ蓄積・検索にかかわる DB 層の二つの要件に分けることができると考える。

UI 層と DB 層の要件を検討するにあたり、まず車両センサーデータを扱う上での課題となるセンサー値についての検討を行う。図 13 に示すように、車両のセンサーや ECU から取得したデータは、バイナリ形式の測定数値が保持されており、蓄積システムへの入力もバイナリデータの数値となる。しかしながら、可読性の問題が存在するため、一般的には正規化を行う際にバイナリ形式の測定数値を浮動小数点型の数値に変換する物理値変換と呼ばれる処理を導入している。

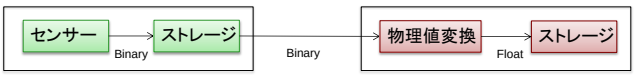

図 13　センサーデータの経路

一般に、計算機にとっては、浮動小数点型よりもバイナリ形式の方が処理速度も速く、必要となる記憶容量も少ないことから、本来はバイナリ形式の測定数値で記録する方が望ましい。ただし、車両センサーデータの場合は、バイナリ形式の数値のままでは、人間にとって可読性のある数値とはならず、一定の演算式を用いて単位系を持つ浮動小数点型の数値に変換する必要がある。図 14 に自動車の標準

化団体である SAE によって規定された診断プロトコルの OBD2 におけるセンサーデータの変換式を記載する[2]。

例えば、車のエンジンの負荷率は

$$X = A * 100/255$$

という変換式で表すことができる。従って、ストレージ側ではバイナリ形式の測定データを記録する際には、同時に変換式を DB 内に記録しておき、他のカラムとの計算を行う場合には、必要に応じて事前に記録した変換式を元に測定データを可読性のある数値に変換する必要がある。

Mode (hex)	PID (hex)	Data bytes returned	Description	Min value	Max value	Units	Formula [1]
01	00	4	PIDs supported [01 - 20]				Bit encoded [A7..D0] == [PID $01..PID $20] See below.
01	01	4	Monitor status since DTCs cleared. (Includes malfunction indicator lamp (MIL) status and number of DTCs.)				Bit encoded. See below.
01	02	2	Freeze DTC				
01	03	2	Fuel system status				Bit encoded. See below.
01	04	1	Calculated engine load value	0	100	%	A*100/255
01	05	1	Engine coolant temperature	-40	215	°C	A-40
01	06	1	Short term fuel % trim—Bank 1	-100 Subtracting Fuel (Rich Condition)	99.22 Adding Fuel (Lean Condition)	%	(A-128) * 100/128
01	07	1	Long term fuel % trim—Bank 1	-100 Subtracting Fuel (Rich Condition)	99.22 Adding Fuel (Lean Condition)	%	(A-128) * 100/128
01	08	1	Short term fuel % trim—Bank 2	-100 Subtracting Fuel (Rich Condition)	99.22 Adding Fuel (Lean Condition)	%	(A-128) * 100/128

図 14　OBD2 取得数値の変換式

以上のことを考慮し、図 12 に示した UI 層と DB 層に関する要件を以下に記載する。

4.2　UI 層に関する要件

最初に、可読性の要件を定義することができる。具体的な内容を以下に記載する。

- 可読性と理解のために、データが一つのテーブルのように見えること
- すなわち一つのレコードに対して近傍時間を集約・結合して一元化できる処理ができること
- ユーザには物理値しか見えないこと

次に以下に記載するような抽出要件をあげることができる。

- 高速に検索が可能なこと

4.3　DB 層に関する要件

データ保存性の観点から以下の具体的な要件を定義することができる。

- センサーの測定データと変換式をそのままストアしておけるようにすること
- データはできればバイナリ形式のままで保存すること。さらに、物理値変換に必要な変換式を保持し、数字データの演算が必要な場合には測定データから数値データへの自動的に展開が可能なこと

検索・演算に関する要件としては、以下を定義することができる。

- 時刻に関して一定範囲内のみのデータを高速に検索/結合処理ができるような仕組み（Index）を持つこと
- 記録容量の節約ができること。差分記録などを活用して記録内容を節約できること。

ここに記載した DB 層の要件を実現させるためには、データをバイナリ形式の測定データのまま保持しつつ、変換式による数値データへの展開が可能で、さらに四則演算を高速に実施し、記録容量を節約しつつ高速に検索する仕組みを構築することが必要となる。特に差分方式での保存は車両センサーデータに対して 1/10-1/20 程度のデータ容量の圧縮が可能になる一方で、列方向に対する高速な検索方式を保持することとは矛盾する。従って、データは差分形式で保存しつつ、インデックスを別に保有するなどの対策が必要になると考えられる。

5.　関連研究

笛田[3][4]らは、電気自動車に後付したセンサーのデータを検索するシステムを開発し、データを各自動車・ドライバごとに一つのタプルとなるように単一テーブルにデータを集約するデータベースを構成している。この研究ではデータ加工のしやすさには正規化利サンプリングとデータの単一集約が効果的であることを示している。

佐藤[5]は自動車における周辺環境の認識を車両センサーのデータから行うべく、そのデータ処理の方法について述べている。

猿渡ら[6]はセンサーデータベースにおけるデータ量の圧縮と問い合わせ処理の高速化を H-Store[7]などでの例がある問い合わせ事前登録方式に改良を加えた方式で実装行った例を報告している。

これらの既存のセンサーからの時系列データのデータベース処理についてはセンサーデータ自体が数値データとして提供されることを想定しており、ストレージに格納されるセンサーのデータとユーザーが操作するデータの形式が異なることを前提としたデータベースに関する提案は見受けられない。

6.　まとめ

本稿の目的は、車両制御用のセンサーデータを用いたベンチマーク評価を行い、車両制御用のセンサーデータを蓄積・活用するためのストレージならびにデータベースの必要要件を定義することである。そのために、最初に、車両制御用のセンサーデータについて、その技術的背景と特性の分析を行い、車両センサーデータは各データがカラム方向に連続して変化する数値データおよびカラム方向に変化するフラグデータから構成されることを説明した。

次に、実際に活用するアプリケーションと同様のサンプル処理を行うベンチマークプログラムを用いて、既存のデータベースシステム上で 500GB 程度の車両センサーデー

タを投入したときのシステムの特性把握を行った。その結果、従来型データベースシステムは、車両センサーデータ処理に向かないこと。また、カラム指向を前提に組まれているデータベースエンジンは車両センサーデータとの相性が良いことを確認した。

これらの結果を元に、車両センサーデータに適したストレージアーキテクチャの検討を行い、ECU 内で使われているバイナリデータをそのまま保持する形で、データを蓄積することが望ましいことを提案するとともに、ユーザ側については、単一テーブルとして見せる機能を持つほかに、バイナリデータを暗示的に数値データへと物理値変換する機構の必要性について説明を行った。

最後に今後の展望について述べる。第一には、既存データベースへの実装の際に、バイナリの格納と物理値変換を行う User Defined Function（UDF）を実装することで、短期的には実使用が可能なデータベースを構築することができると考える。また、中期的には、バイナリデータを直接保存できるストレージエンジンについても現在検討を進めている。現在は検討の段階であり、プロトタイプ実装にも進んでいない状況だが、今後大規模な車両センサーデータの受け入れを想定した拡張性を考慮した設計を行っていきたい。

参考文献

1) ISO 11898-1:2003 Road vehicles -- Controller area network (CAN) -- Part 1: Data link layer and physical signaling http://www.iso.org/iso/catalogue_detail.htm?csnumber=33422
2) OBD-II_PIDs http://en.wikipedia.org/wiki/OBD-II_PIDs
3) 笛田 尚希、萩本真太朗、林 拓也、讃井 崚、富井 尚志: 車載センサを用いた EV エネルギー消費ログ DB のモデル構築と V2X 効果推定, DEIM Forum 2013, (2013)
4) 笛田 尚希、萩本真太朗、出口 達、富井 尚志、電気自動車の走行ログを蓄積する DB の構築と EV 消費電力推定手法, DEIM Forum 2012、 (2012)
5) 佐藤 健哉,自動車走行環境認識のためのセンサデータ処理機構(センサデータ処理,ストリームデータベース,及び一般),電子情報通信学会技術研究報告. DE, データ工学,09135685,一般社団法人電子情報通信学会, 2010-06 Vol110,Num107,P51-56
6) 猿渡 俊介,高木 潤一郎,川島 英之,倉田 成人,森川 博之,センサデータベースマネージャにおける問合せ処理とデータ圧縮の同時最適化,情報処理学会論文誌,Vol 53,Num 1,P320-335
7) Kallman, R., Kimura, H., Natkins, J., Pavlo, A., Rasin,A., Zdonik, S., Jones, E.P.C., Madden, S., Stonebraker,M., Zhang, Y., Hugg, J. and Abadi, D.J.: H-store: AHigh-Performance, Distributed Main Memory Transaction Processing System, Proc. VLDB Endowment, Vol.1,No.2, pp.1496-1499 (2008).