IoT デバイスによるセンサデータの取得蓄積に関する研究

奥田　誠(情報・生産技術部 システム技術グループ)

１．　はじめに

近年、IoT や DX (デジタルトランスフォーメーション)を背景に、センサなどのデバイスは増加傾向にある。産業用途としては、装置の故障診断や工具の劣化予測などを目的として、振動や音響、電力などのセンサデータを活用した研究[1)]が行われている。また IoT は、業務改善や課題解決を目的とする手段であるが、IoT 導入が目的となっている場合がある。この場合、必要なセンサの種類や設置位置、サンプリング周期などを深く考慮せずに IoT システムが構築されることが多い。そうすると、取得したデータは、故障診断や劣化予測には不十分となり、データが無駄になってしまう可能性がある。データを有効に活用するためには、あらかじめ、加工現象の解析に適したセンサの種類や設置位置、サンプリング周期を検討する必要がある。さらに、データを漏れなく必要なサンプリング周期で蓄積するためには、データサーバや IoT デバイスのハードウェア性能、データベースの種類やデータ通信方式などを適切に選定しなければならない。

そこで本研究では、簡易的な IoT システムでセンサデータのサンプリング周期の実態を調査することを目的として、データサーバ、IoT デバイスおよびセンサで構成させる IoT システムを構築し、簡易的なデータ取得実験を行った。本稿では、IoT システムの一例およびセンサデータのサンプリング周期について報告する。

２．　実験方法

２．１　実験構成

本実験の構成を図 1 に示す。IoT デバイスには、イーサネットコネクタが付属しておりセンサの取り付けが容易な Raspberry Pi を用いた。センサには、装置の故障診断で多く用いられており、かつ変化を与えやすく簡易的な実験に適している、加速度センサを用いた。Raspberry Pi には、その標準的な OS である Raspberry Pi OS を導入している。Raspberry Pi に Grove AI (Artificial Intelligence) HAT (Hardware Attached on Top)を接続し、Grove AI HAT に実装されている加速度センサ ADXL345 を活用する。ADXL345 の仕様と本実験での設定を表 1 に示す。データサーバには、センサデータ蓄積のために、データベースを構築する。データベースの種類は主に、関係データベース (Relational Database: RDB)と NoSQL (Not Only SQL) データベースが存在する。センサデータを取り扱う場合は、拡張性や処理速度が優れている NoSQL が適している。さらに NoSQL データベースには、表 2 に示すように、複数の種類が存在する。本実験では、センサデータの高速サンプリングを想定

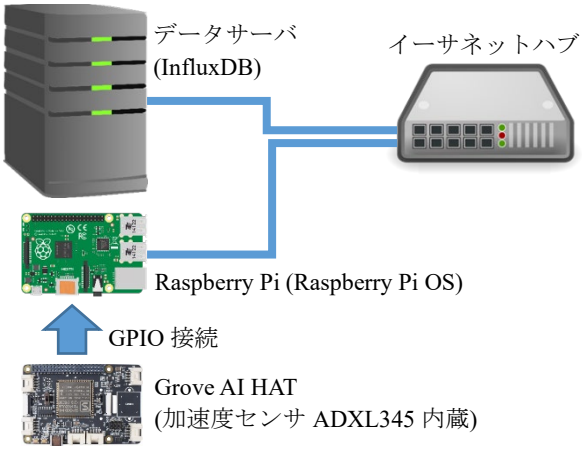

図 1　実験構成

表 1　ADXL345 の仕様と設定

	単位	仕様	設定
測定範囲	m/s^2	±19.6, ±39.2, ±78.4 or ±156.9	±19.6
出力データレート	Hz	0.1 ～ 3200	3200
通信方式	-	SPI or I^2C	I^2C
通信速度	Hz	任意	2M

表 2　NoSQL DB の種類

種類	特徴	主な製品
時系列	時系列データを効率的に処理可能	InfluxDB
キーバリュー型	キーとバリューをペアにして格納	Redis
ドキュメント型	ドキュメントをそのまま格納可能	MongoDB
列指向型	列単位で処理	Cassandra
グラフ型	グラフ構造で格納	Neo4j
オブジェクト型	オブジェクト指向に基づくデータモデル	Caché

しているため、時系列データの処理に特化した時系列 DB である InfluxDB を利用することにした。

２．２　実験手順

Raspberry Pi から Grove AI HAT 上の ADXL345 の加速度データを取得し、データを保存する実験を、以下のように実施した。

(1)　Raspberry Pi 上のストレージにテキスト保存
(2)　データサーバの InfluxDB に 1 サンプリング毎転送
(3)　データサーバの InfluxDB にデータをまとめて転送

データ取得中は、断続的に振動を与え、合わせて Raspberry

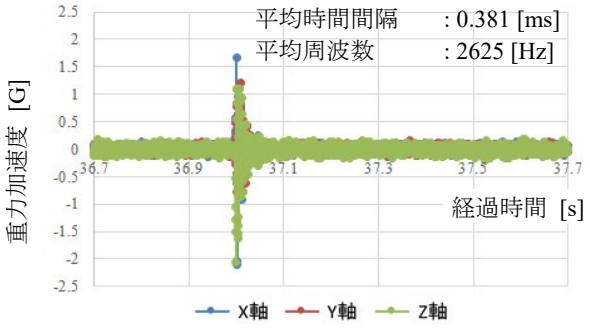

図 2　実験(1)の加速度センサのデータ

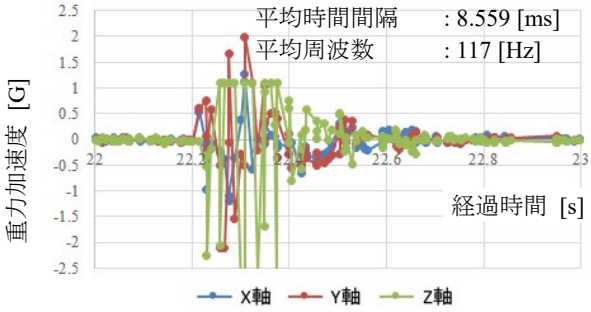

図 3　実験(2)の加速度センサのデータ

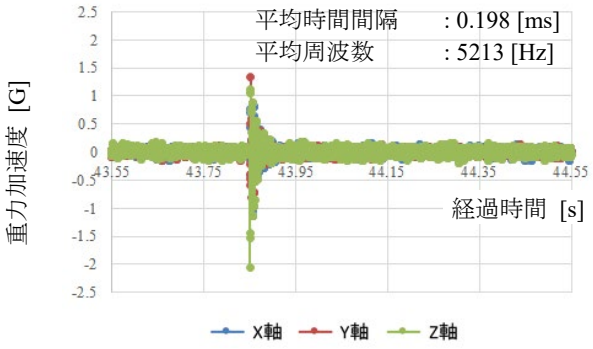

図 4　実験(3)の加速度センサのデータ

Pi の CPU 使用率も取得した。

3.　解析結果および考察

実験(1)、(2)、(3)で取得した加速度データをそれぞれ図 2、3、4 に示す。横軸を経過時間、縦軸を重力加速度とした。ADXL345 の出力データレート、いわゆる単位時間あたりにセンサが出力する個数は 3200 Hz である。実験(1)では図 2 に示すとおり平均 2625 Hz であるため、センサ性能の 80%以上のサンプリング周期でデータを取得できている。一方、実験(2)では、図 3 に示すように平均 117 Hz であるため、センサ性能の 4%弱しか活かせていない。これは、InfluxDB へのデータ転送のオーバーヘッドが大きいためであると考えられる。そこで、実験(3)では、Raspberry Pi 上で一時的にデータを蓄積し、まとまったデータ（本実験では 10,000 サンプリング）毎に InfluxDB へ転送するよう

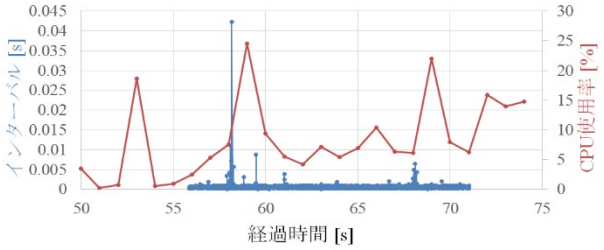

図 5　データ取得間隔(インターバル)と CPU 使用率

改良した。その結果、図 4 に示すように、平均 5213 Hz でのデータ取得が可能となった。センサ性能である 3200Hz を越えているのは、通信の速度が出力データレートを上回り、連続して同じ値を取得している場合があるためである。

しかし、2 点のデータ間の時間間隔（インターバル）を詳しく調査すると、図 5 に示すように、インターバルが最大で約 0.043 s (23.3 Hz)となることがあった。CPU 使用率が高くなる直前にインターバルが大きくなっていることから、Raspberry Pi OS の割り込み処理が原因となって、データ取得の処理が中断されていると推測できる。したがって、割り込み処理の影響を分離できない OS では、高速サンプリングでのデータ取得を安定的に行うのは難しく、最大でも 20 Hz 程度のサンプリング周波数で、加工現象が捉えられる装置に対して有効であることが分かった。

より高速なサンプリングでのデータ取得を行う場合は、データ取得の処理に対して割り込み処理の影響を受けないシステム構築が必要である。本システムに対しては、Raspberry Pi にリアルタイム OS[2)]を導入することや、センサデータの取得処理を行うデバイスを Raspberry Pi の代わりに Arduino[3)]や Raspberry Pi Pico[4)]のようなマイコンを用いることが考えられる。

4.　まとめ

Raspberry Pi を用いて､加速度センサである ADXL345 に対して I^2C 通信によってセンサデータを取得し、そのデータをサーバに導入した InfluxDB へ蓄積するシステムを構築した。高速サンプリングでデータ取得する場合、1 サンプリング毎ではなく、まとめてデータ転送する必要があることを示した。また、割り込み処理が原因でデータ取得インターバルが大きくなることを示し、より高速なサンプリングでデータを取得する方法について提案した。

装置の故障解析や工具の劣化予測を目的とする場合、その加工現象を捉えるために必要なサンプリング周期を把握し、適切なシステム構築を行うことが重要である。

【参考文献】

1) 芝田和雄, 音響・振動に基づく最近の故障診断技術, 騒音制御, Vol.31, No.3, 219-224 (2007)
2) https://aws.amazon.com/jp/freertos/
3) https://www.arduino.cc/
4) https://www.raspberrypi.org/products/raspberry-pi-pico/