計測自動制御学会東北支部 第 356 回研究集会 (2025.12.10)
資料番号 356-14

さくらんぼ収穫ロボット用台車の開発

Development of a Mobile Platform for Cherry Harvesting Robot

○五十嵐 凜，妻木 勇一

○ Rin Igarashi, Yuichi Tsumaki

山形大学

Yamagata University

キーワード： 農業用ロボット (Agricultural robot)，移動台車 (Mobile platform)，CAN (Controller Area Network)，QDD モータ (Quasi-Direct Drive motor)，

連絡先： 〒 992-8510 山形県米沢市城南 4 丁目 3-16 山形大学工学部 妻木研究室，
Tel.: (0238)26-3882，E-mail: twf04672@st.yamagata-u.ac.jp

1. 諸言

山形県の主要農産物であるさくらんぼ生産現場では，農業従事者の高齢化と減少に伴う労働力不足が深刻化している．この課題解決のため，我々はさくらんぼ収穫作業の自動化を目指したロボット開発を行っている [1)]．

さくらんぼ収穫ロボットは，収穫を行うロボットアーム部と園地を移動するための台車部に大きく分けることができる．現在，我々はクローラ型の台車を使用しているが，超信地旋回を行うと柔らかい園地を大きく削ってしまうという問題がある．特にさくらんぼは根が浅いため，根を傷めない台車が必要である．そこで，低コスト化を目指したステアリング機構（前輪操舵・後輪駆動）を持つ移動台車を開発してきた [2)]．Fig. 1 に開発中の台車を示す．ステアリング型の農業用に使える台車は，エムスクエア・ラボの Mobile Mover[3)]，輝翠の Adam[4)]，SUZUKI の MITRA[5)]，DON-KEY の CP200[6)] などがある．我々が開発している台車は，さくらんぼ収穫ロボットに必要十分な性能に絞ることで，これらの台車よりも簡素な構成とし，低コスト化を狙っている．

一方，移動台車は，自律走行（SLAM や経路計画）の実装やロボットアームとの連携が求められる．このため，計算負荷の分散やリアルタイム性の確保，機能の拡張性が不可欠である．そこで本論文では，ロボット用ミドルウェアである ROS 2 (Robot Operating System 2) と，耐ノイズ性や拡張性に優れた通信規格である CAN (Controller Area Network) を採用した制御システムを設計・実装すると伴に，ステアリングに QDD モータを導入したさくらんぼ収穫ロボット用台車について詳述する．

2. 試作車両のハードウェア概要

最初に，移動台車のハードウェアについて述べる．駆動方式には，前輪操舵・後輪駆動（2WD）を採用している．旋回時に園地への負荷を最小限に抑えるためである．

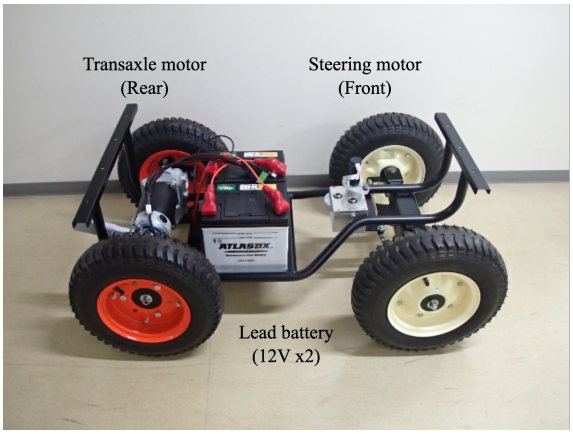

Fig. 1 The prototype mobile platform. (without electronic control system)

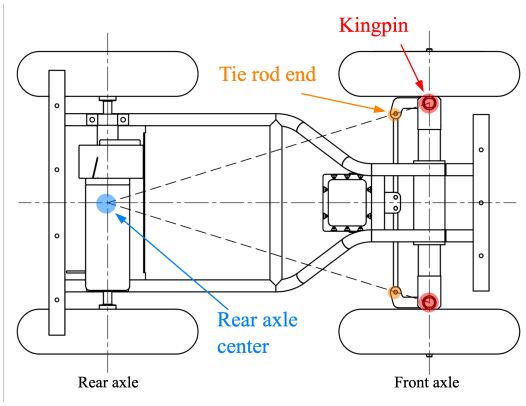

Fig. 2 Steering mechanism based on Ackermann geometry.

操舵機構には，アッカーマンジオメトリを採用している (Fig. 2)．これにより，旋回時のタイヤの横滑りを抑制し，円滑な走行を実現すると伴に，園地への負担を抑えることができる．また，路面の凹凸を吸収する機構として，サスペンションの代わりにピボットアーム式を採用することで，構造の簡素化を図っている．

後輪の駆動源には，DC モータ，減速機および差動装置（デファレンシャルギヤ）が一体となったトランスアクスルを採用した．これはシニアカー等に用いられる汎用ユニットであり，低コストかつ平坦な農地での走行に適した特性を持つ．しかし，内蔵センサや通信機能を持たないブラシ付き DC モータであるため，ロボット用アクチュエータとして制御するには，外部に駆動回路やセンシング系を構築する必要がある．

3. システム構成

3.1 概要

構築したシステムの概要を Fig. 3 に示す．制御用機材としてノート PC を用意し，OS は Ubuntu 22.04 LTS を使用した．また，ロボットアプリケーション開発用フレームワークである ROS 2 Humble を採用した．これにより，各機能を独立性の高いノードとして実装し，システム全体の拡張性を確保している．

PC と各アクチュエータは，物理的には CAN バスを介して接続される．CAN は差動信号を用いるため耐ノイズ性が高く，バス型トポロジーにより配線が簡素化される利点がある．PC と CAN バスの接続には USB-CAN アダプタ（CAN-able 2.0 Pro）を用い，
Linux 標準の SocketCAN インターフェースを介して ROS 2 ノード群と通信を行う．

また，電源系には 12V の鉛蓄電池を 2 個直列に接続し，24V にして使用している．鉛蓄電池を用いる理由は容易に手に入り，低コストであることが第一の理由である．デメリットとされる重量は車体の低重心化に貢献している．安全確保のため，緊急停止スイッチとリレー，ヒューズを回路に組み込み，緊急時にはアクチュエータへの電力供給を遮断する構成とした．

3.2 ステアリング駆動系

操舵用アクチュエータとして，Fig. 4 に示す QDD (Quasi-Direct Drive) モータである RobStride 03 を採用した．マイコンやモータドライバ，各種センサ，通信機能を内蔵した QDD モータを採用することで，システム構成を簡素化することができる．

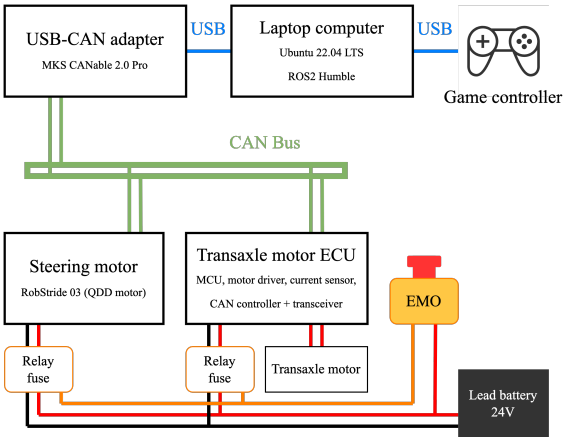

Fig. 3　System overview of the control architecture.

Fig. 4　RobStride 03 QDD motor used for steering drive.

本モータは CAN バス経由での通信に対応しており，MIT プロトコルを用いた位置・速度・トルクの制御が可能である．また，アブソリュートエンコーダを内蔵しているため，起動時の原点合わせが不要となる利点もある．加えて，QDD 特有の高いバックドライバビリティと高トルク密度により，安価な構成でありながら，優れた応答性と外乱に対する柔軟性を実現している．このため環境になじむような無理のないステアリングなど，より高度なステアリング制御が期待できる．

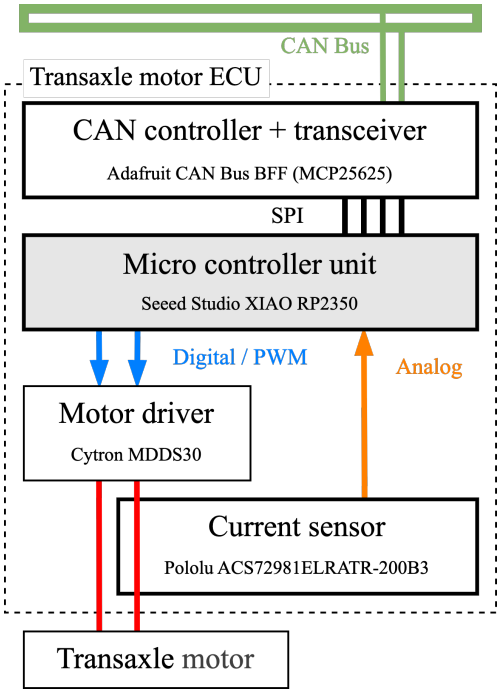

Fig. 5　Drive system with custom ECU for transaxle motor.

3.3　走行駆動系と自作 ECU

後輪駆動には，減速機と差動装置が一体となったトランスアクスルモータ（DC モータ）を使用している．本モータは単体では駆動回路や通信機能を持たないため，ROS 2 からの CAN 通信を介した制御を実現すべく，専用の ECU (Electronic Control Unit) を独自に実装した．

自作 ECU は，マイコンに Raspberry Pi RP2350，CAN コントローラ・トランシーバに MCP25625，電流センサに ACS72981 を採用して構成した．なお，モータドライバは MDDS30 を搭載している．

RP2350 に実装したファームウェアは，上位 PC との CAN 通信および入出力処理を司る．具体的には，受信した CAN メッセージから目標デューティ比を抽出してモータドライバへ PWM 信号を出力すると同時に，電流センサから取得した電流値を CAN メッセージに格納して返信する処理を行う．

4. 制御ソフトウェアの実装

4.1 トランスアクスルモータ制御用 ECU のファームウェア

トランスアクスルモータ用 ECU のマイコン（RP2350）には，C++言語を用いてファームウェアを実装した．本ファームウェアは，CAN 通信の受信割り込みをトリガとするイベント駆動型の処理系となっている．

ROS 2 ノードから送信されるモータ出力/回転方向の制御コマンドを受信すると，モータ出力の目標値を変換し，モータドライバへの入力信号を更新する．同時に，負荷を測定するための電流センサの値を更新し，電流値を返信する処理を行う．これにより，上位 PC からの指令値の反映と負荷状態の監視が可能である．

加えて，安全機能として通信ウォッチドッグを実装した．上位 PC からの CAN 指令が一定時間（500ms）以上途絶えた場合，システム異常と判断して自動的にモータ出力を遮断することで，通信断絶時の暴走を防ぐ仕様としている．

4.2 ROS 2 システムの実装

上位系となる ROS 2 のソフトウェア構成を Fig. 6 に示す．本システムは，ゲームコントローラの操作入力を受け付ける `joy` ノード，指令値を分配・変換する `bridge` ノード，そして各アクチュエータと通信を行うドライバノード群（`rs03_controller`, `transaxle_controller`）から構成される．これら ROS 2 システムと CAN バスの接続には `ros2_socketcan` パッケージを採用し，ROS トピックと CAN フレームの相互変換を行っている．

PC とアクチュエータ間の CAN 通信のデータ構造を Fig. 7 に示す．ステアリング用 QDD モータには MIT プロトコル（位置・速度・トルク指令を含む 8 バイトのフレーム）を用い，トランスアクスル用 ECU には，32bit 浮動小数点

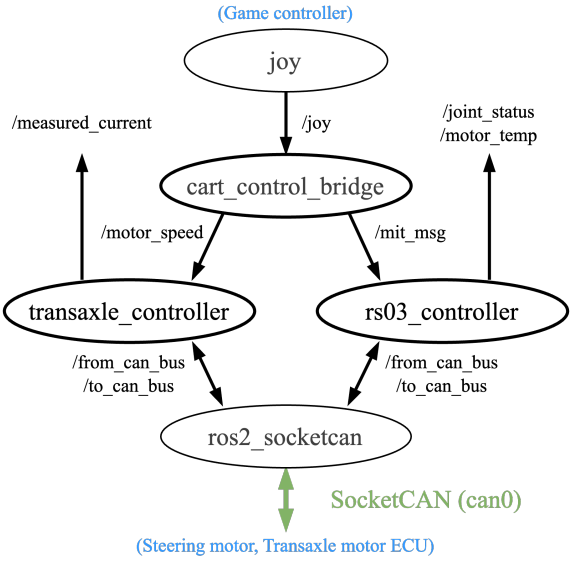

Fig. 6 Software architecture implemented on ROS 2.

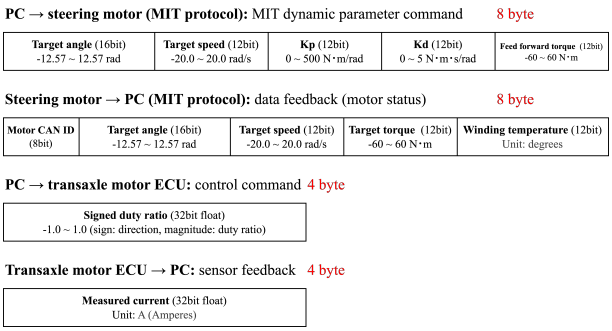

Fig. 7 Data structure for CAN communication.

数（4 バイト）を用いたデータフォーマットを定義した．

4.3 通信安定性の検証

構築したシステムの通信安定性を検証するため，通信速度 1Mbit/s の条件下でバス負荷率等の測定を行った．本検証では，将来的なセンサ増設や高負荷時の挙動を確認するため，各 ECU に対して，Fig. 7 に示したデータフレームを 1kHz の周期で送受信する設定とした．

この条件で `canbusload` コマンドによる計測を行った結果，1Mbit/s の帯域に対して

約 460kbit/s のスループット（バス占有率 46%）が観測された．パケットロス（フレーム欠落）等の通信エラーは確認されず，十分な通信帯域と安定性が確保されていることを確認した．

5. 結言

本稿では，さくらんぼ収穫ロボット用台車に ROS 2 と CAN バスを基盤とした堅牢かつ拡張性の高い電子制御システムを構築した．ステアリング機構への QDD モータの導入と，走行用 DC モータを制御する自作 ECU の開発により，すべての駆動系を CAN ネットワーク上で統合し，上位 PC からの統一的な制御を実現した．

今後，構築したさくらんぼ自動収穫用台車を用いて，自律走行システムを構築する予定である．

参考文献

1) 鈴木佑也，金澤秀太，佐々木成海，小原永義，武士沢慎太郎，瀬野智広，峯田貴，妻木勇一: さくらんぼ自動収穫ロボット，計測自動制御学会東北支部第 304 回研究集会，資料番号 304-8，2016.

2) 鈴木匠泉，妻木勇一: ステアリング型農業用自律移動台車の設計，第 25 回計測自動制御学会システムインテグレーション部門講演会予稿集，2B4-07，2024.

3) 株式会社エムスクエア・ラボ: MOBILE MOVER，https://mobilemover.jp

4) 輝翠 TECH 株式会社: Adam ファームロボット，https://kisui.ai/adam/adamoverview/jp

5) スズキ株式会社: MITRA，https://www.suzuki.co.jp/car/entertainment/mobilityshow/2025/mitra_concept/

6) 株式会社 DONKEY: CP200，https://donkey.co.jp/product/mainunit/cp200/