土工現場用 CPS プラットフォーム ROS2-TMS for Construction の開発
-第 3 報 タスク管理機構の実装-

ROS2-TMS for Construction: CPS platform for earthwork sites
-Implementation of Task Management Mechanisms-

○ 笠原侑一郎 (九州大学)　正 井塚智也（九州大学）　柴田航志 (九州大学)
学 前田龍一（九州大学）　学 高野智也 (九州大学)　正 松本耕平 (九州大学)
木村駿介（清水建設）　深瀬勇太郎 (清水建設)　横島喬（清水建設）
山内元貴（土木研究所）　遠藤大輔 (土木研究所)　橋本毅 (土木研究所)
正 倉爪亮（九州大学）

Yuichiro KASAHARA, Kyushu University, kasahara.yuichiro.res@gmail.com
Tomoya ITSUKA, Kyushu University, itsuka@irvs.ait.kyushu-u.ac.jp
Koshi SHIBATA, Kyushu University, shibata@irvs.ait.kyushu-u.ac.jp
Ryuichi MAEDA, Kyushu University, maeda@irvs.ait.kyushu-u.ac.jp
Tomoya KOUNO, Kyushu University, kouno@irvs.ait.kyushu-u.ac.jp
Kohei MATSUMOTO, Kyushu University, matsumoto@irvs.ait.kyushu-u.ac.jp
Shunsuke KIMURA, Shimizu Corporation
Yutaro FUKASE, Shimizu Corporation
Takashi YOKOSHIMA, Shimizu Corporation
Genki YAMAUCHI, Public Works Research Institute (PWRI)
Daisuke ENDO, Public Works Research Institute (PWRI)
Takeshi HASHIMOTO, Public Works Research Institute (PWRI)
Ryo KURAZUME, Kyushu University, kurazume@ait.kyushu-u.ac.jp

In recent years, the shortage of human resources at earthwork sites has been accelerating due to industrial accidents and the declining birthrate and aging population. Against this background, Our lab is developing a CPS platform called "ROS2-TMS for Construction" as a system that simultaneously improves the efficiency and safety of earthwork work. This paper introduces a newly developed task management mechanism as a continuation of the ROS2-TMS for Construction development. The task management mechanism is a mechanism for operating actual construction machinery based on scenarios called task sequences given to ROS2-TMS for Construction.In this time, I adopted the Behavior Tree as the task scheduler and incorporated it into ROS2-TMS for Construction with some extensions according to the ROS2 specification. An overview of the extensions of Behavior Tree we adopted to task management mechanism of ROS2-TMS for Construction will also be explained in this paper.

Key Words: 建設ロボット・建設機械, Robot Operating System(ROS), Cyber Physical System

1　緒言

近年、工事現場においては労働災害や少子高齢化による人材不足が課題となっている。これらの課題に対し、我々は土工作業効率化と安全性向上を同時に実現するシステムとして土工現場用 Cyber Physical System(CPS) プラットフォーム ROS2-TMS for Construction を開発している。CPS とは実空間とサイバー空間が相互に連動し、動作するように構成されたシステムのことである。まずは ROS2-TMS for Construction におけるこれまでの開発経緯について紹介する。

ROS2-TMS for Construction 開発の第 1 報 [1] では工事現場から取得した静的・動的地形情報や建設機械の位置・姿勢情報、地盤情報をもとに VR 空間上に工事現場を復元した。その上で実証実験として VR 空間上に復元した工事現場を VR ゴーグルを介して高い没入感のなかで確認可能であることを検証していた。

第 2 報 [2] では第 1 報の発展として、センサポッド [3] と呼ばれるシステムを ROS2-TMS for Construction の機能の一部に取り込んだ。センサポッドには GNSS や 3D LiDAR, 360° カメラ, バッテリー等が搭載されており、360° カメラから送られてくるリアルタイムな画像データを VR 空間上に表示されたセンサポッドをタッチすることで VR 空間上で表示可能な仕様にしていた。このセンサポッドについては現在、「プチ・センサポッド」[4] という名称で開発が続けられている。

本論文はその続報とし、CPS の新たな機能として建設機械の実機を ROS2-TMS for Construction から操作するための機能を紹介する。また、現在の ROS2-TMS for Construction が取りうる出力先の 1 つとして土木研究所が開発を進める OPERA[5] が存在する。本稿では OPERA のシミュレータである OperaSim-PhysX を使用したタスク管理機構の実証試験の結果について紹介する。

2　ROS2-TMS for Construction

図 1 に現在の ROS2-TMS for Construction のアーキテクチャを示す。ROS2-TMS for Construction のタスク管理機構では図 1 に示したモジュール群の内、TMS_DB, TMS_SP, TMS_UR, TMS_TS, TMS_RP, TMS_RC が用いられる。本稿では特に TMS_DB, TMS_SP, TMS_UR, TMS_TS を紹介する。これ以外のモジュール群については第 1 報 [1]、第 2 報 [2]、第 4 報 [7] にて説明されているのでそちらを参照されたい。

- Database (TMS_DB)
  タスク管理機構が参照するタスクデータや動的・静的パラメータのデータベース (DB) への書き込み・書き出し機能を担うモジュール
- Sensing Processor (TMS_SP)
  センシング処理後のデータを受け取り、TMS_DB を介して DB 上のパラメータデータを最新の値に書き換える機能を担うモジュール

- User Request (TMS_UR)
  ユーザーからタスク管理機構への入力を受け取る機能を担うモジュール
- Task Schedular (TMS_TS)
  ROS2-TMS for Construction のタスクスケジューラである Behavior Tree に関する機能の大部分を担うモジュール

なお、タスク管理機構の実装に際しては ROS2-TMS for Construction の前身である ROS2-TMS[8] の実装を参考に、従来 CPS のタスクスケジューラとして用いられていた Finite State Machine(FSM) を Behavior Tree へと移行し、工事現場のような動的な屋外環境にも対応できるように細部の仕様を変更した。

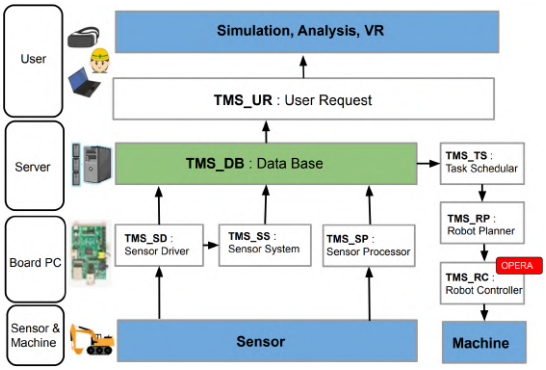

Fig.1 ROS2-TMS for Construction architecture

3　実機制御の枠組みの全体像

ROS2-TMS for Construction では図 2 に示す流れで実機制御が行われる。実際に ROS2-TMS for Construction が処理を担うのは図 2 の赤枠で囲った範囲である。動作を詳細に説明する。

Fig.2 Overall architecture of the actual construction machinery control framework

1. まずは目的の施工操作に応じたタスク列を設計する。タスク列とは一度のタスクで実行する施工操作の流れについて記載されたシナリオデータである。タスク列はサブタスクと呼ばれるノードを複数個繋げた構造をしている。サブタスクはタスクを構成する最小単位のノードであり、まず最初にこのサブタスクを Subtask Node というノード上で実装する。
2. Subtask Node が実装できたら、Groot を使用してタスクツリーを構築する。Groot とはタスク列生成のためのエディタ機能と Behavior Tree が実行しているタスクを可視化するためのモニタ機能を有したツールである。また、タスクツリーとはタスク列をツリー状に可視化したものである。このタスクツリーを作成する過程において Leaf Node というノードを使用する。Leaf Node とは Subtask Node とタスクツリーを接続するための仲介役を担うノードであり、タスクツリーにおいては末端の葉の部分に配置するノードである。Leaf Node を Groot で使用する際は、手動でノードを登録する方法とノード名や入出力ポート名が記載されたファイルを Groot に読み込ませる 2 種類の方法が存在する。Leaf Node のノード名のリストについては、基本的なものは事前に ROS2-TMS for Construction 上に用意してある。
3. 次に Subtask Node を Leaf Node を介してタスクツリーと接続する。Subtask Node は Behavior Tree のタスク列の中に直接組み込まれていない特殊なノードであり、Groot 上では Leaf Node に接続する Subtask Node を各々の Leaf Node 上の Black Board のパラメータで指定する。その様子を図 3 に示す。 同時に Leaf Node 上の Black Board の値にはサブタスク処理で使用する DB 上の動的パラメータの値を検索するための 2 種類のキーの値を指定する。現在は model_name と record_name という 2 種類のキーを元に、表 2 に示す仕様の DB 上の動的パラメータの検索を行う。
4. 上記の手順を経てタスクツリーを Groot 上で作成できたら、これに対応する xml 形式のタスク列を出力することができる。これを文字列化したものにメタデータを付与した上でタスクデータとして DB へと追加する。この部分の作業は事前に ROS2-TMS for Construction に用意された ROS2 ノードを使用して半自動的に実行することができる。この ROS2 ノードを使用して DB 上に生成されるタスクデータの仕様は表 3 に示す通りである。
5. 作成した DB 上のタスクデータを指定した上で Behavior Tree を起動すると対応するタスクデータを検索し、DB 上に合致するものがあればそのタスク列を読み込む。 この際にタスク列に含まれる Leaf Node 上で指定された全ての Subtask Node を起動しておく。この状態で、以下図 4 に示す GUI ボタンをクリックすると Behavior Tree はタスクの実行を開始する。また、タスクによっては、現場における動的な状態を示す動的パラメータの値を利用したい場面がある。これについてはシステム外部にてセンシング処理を行ったデータを TMS_SP, TMS_DB を介して ROS2-TMS for Construction の DB へとほぼリアルタイムで取り込む。取り込んだパラメータデータは Subtask Node の実装に従って動的パラメータを利用したいタイミングで TMS_DB を介して DB 上から値を取得する。
6. タスクスケジューラは上記の手順を経て取得した動的パラメータの値と Subtask Node の実装に基づいて建機実機へと操作指令を送る。現行の ROS2-TMS for Construction の出力先には OPERA 対応建機の実機と OPERA のシミュレータ OperaSim-PhysX 上の建機が存在し、ROS2-TMS for Construction のタスク管理機構を使用した制御を行うことが可能となっている。

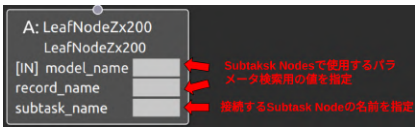

Fig.3 Overview of Black Board parameters for Leaf Node

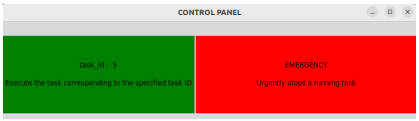

Fig.4 GUI button for starting task execution

4　タスクスケジューラについて

ROS2-TMS for Construction のタスクスケジューラには Behavior Tree が用いられている。Behavior Tree とはツリー状にサブタスクや各種ノードをつなぎ合わせて作成したタスク列をもとに各々のノードの状態を都度判別しながら与えられたタスクを実行していくタスクスケジューラである。Behavior Tree の動作説明図を図 5 に示す。図 5 に例示したタスクツリーは赤矢印のような流れで、基本的には深さ優先探索で実行される。(ノードの起動順を黄色番号で示す) また、通常の Behavior Tree においては図 5 に示す Action Node でサブタスク処理が実装される。

今回は ROS2-TMS for Construction のタスク管理機構の実装に際し、一部 ROS2 の仕様に基づいて Behavior Tree を拡張した。その説明のためにタスクツリーの実装例を図 6 に示す。

拡張では図 5 に示すタスクツリーのように Behavior Tree の末端である Action Node を Leaf Node と Subtask Node という 2 種類のノードへと分割した。各々のノードは以下のような役割を持つ。

- Leaf Node
  タスクツリーにおける末端のノードである。Subtask Node と ROS2 Action を介して接続し、Subtask Node から送られてくる Subtask

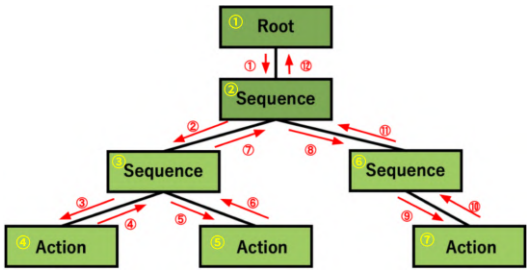

Fig.5 The example of task tree in Behavior Tree

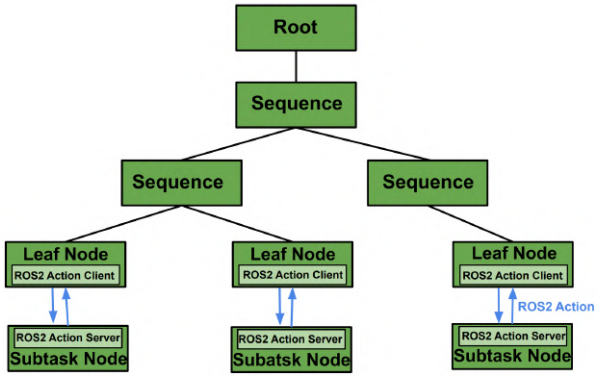

Fig.6 Schematic diagram of the extended Behavior Tree

Node の状態信号を受け取って NodeStatus へと変換し、より上位のノードへ伝播させることで Behavior Tree が Subtask Node の状態を管理することを可能にしている。また、タスク実行のキャンセル時には root ノードからトップダウン形式でキャンセル処理の呼び出しが行われる。このキャンセル処理は Subtask Node に実装されており、Leaf Node は上位ノードから送られてくるキャンセル要請を Subtask Node へと ROS2 Action を介して伝達し、Subtask Node から送られてくるキャンセル操作の実行結果を NodeStatus へと変換した上で root ノードへと伝達する機能も有している。

- Subtask Node

  Leaf Node と ROS2 Action を介して接続されるノードであり、サブタスクと呼ばれるタスクの最小単位の処理が実装される。また、各々のサブタスクの実装に応じたキャンセル処理もここに実装される。ここでいうキャンセル処理とは、サブタスクが緊急で停止しなければならない状態になった際に呼び出され、サブタスクが動作対象としている OPERA 対応建機等の停止処理が完了したことをタスクスケジューラが判別した後に安全に全体の処理を停止させることを目的として実装される。

Leaf Node と Subtask Node 間の接続の動作概要は以下の通りである。

1. はじめに Leaf Node がタスクスケジューラによって起動される。すると読み込んだタスク列の中から Black Board 上のパラメータを読み込む。現在、Leaf Node の Black Board に設定する必須のパラメータには model_name と record_name、subtask_name の 3 種類がある。その概要は図 3 に示したとおりである。
2. Leaf Node は Black Board を通じて model_name と record_name を受け取ると、これらの値を Goal メッセージとし、subtask_name に対応する Subtask Node 上に立てられた ROS2 Action Server に向けて送信する。
3. Subtask Node 上の ROS2 Action Server は Goal メッセージを受け取ると実行可否を判断し、可能であればサブタスク処理を始める。このサブタスク処理の過程において model_name と record_name を使用して DB 上の動的パラメータの検索を行い、TMS_DB を介して取得する。この処理を行う関数は Subtask Node の基底クラスに実装されており、サブタスク処理の任意のタイミングで何度でも DB 上の動的パラメータを取得することが可能である。
4. Subtask Node 上の ROS2 Action Server はサブタスク処理の実行途中に一定間隔で feedback メッセージを Leaf Node 上の ROS2 Action Client へと返す。Leaf Node 上の ROS2 Action Client はメッセージを受け取ると親ノードへ NodeStatus::RUNNING を返す。サブタスク処理の実行中はこの処理を繰り返す。
5. サブタスク処理の実行中に処理がキャンセルされた際にはそれに対応した Result メッセージを Subtask Node 上の ROS2 Action Server から Leaf Node 上の ROS2 Action Client へと返す。これを Leaf Node 上の ROS2 Action Client が受け取ると、親ノードへ NodeStatus::FAILURE を返す。
6. サブタスク処理がキャンセルされることなく終了した際にはそれに対応した Result メッセージを Subtask Node 上の ROS2 Action Server から Leaf Node 上の ROS2 Action Client へと返す。これを Leaf Node 上の ROS2 Action Client が受け取ると、親ノードへ NodeStatus::SUCCESS を返す。

なお、Leaf Node と Subtask Node には基底クラスが存在する。Leaf Node は基底クラス側に大部分のプログラムが実装されており、ユーザーが Leaf Node を実装する際にはヘッダファイル上でノード名・Black Board のポート名 (必須ポート model_name, subtask_name を含む) のみを指定するだけで実装可能である。一方で Subatsk Node は実装に拡張性を持たせることを目的とし、基底クラスには GetParamFromDB 関数のみ実装している。GetParamFromDB 関数は DB から動的パラメータの値を取得してサブタスク処理に利用するための関数であり、表 2 における model_name, record_name をキーとしてデータの検索を行い、データが見つかった際には “そのほか” と記載がある部分のパラメータデータを返す関数である。

5 データベースについて

ROS2-TMS for Construction の DB にはタスクスケジューラが実行するタスクのタスクデータと、タスク実行中に使用される動的・静的パラメータのデータの値が格納されている。タスクデータの仕様を表 1 に、パラメータデータの仕様を表 2 に示す。タスクスケジューラは必要に応じてここに格納された値を参照する。

Table 1 Specification of task data in DB

要素	概要	種別
_id	MongoDB が自動で付与するデータ識別用 ID	必須
model_name	タスクの動作対象を指定 (例：zx200)	必須
description	必要に応じてタスクデータの内容を記載	選択
task_id	CPS のデータ識別用 ID。必ず固有の番号を指定する	必須
task_sequence	タスクスケジューラが読み込むタスク列	必須

Table 2 Spetification of parameter data in DB

要素	概要	種別
_id	MongoDB が自動で付与するデータ識別用 ID	必須
model_name	タスクの動作対象を指定 (例：zx200)	必須
type	動的か、静的かを指定 (static か dynamic で指定)	必須
record_name	パラメータの意味を識別可能な文字列を入れる (例：boom_joint)	必須
そのほか	サブタスク処理で利用するデータを入れる (例：x,y,z)	必須

6 タスクについて

タスクを実行する際には事前にタスク列を準備しておく必要がある。ROS2-TMS for Construction では Groot を使用して作成した xml 形式のタスク列を ROS2 ノードを使用して DB 上にタスクデータとして保管することを想定している。タスク列の作成時には Groot のエディタ機能を使用し、任意のタスクツリーを構築して xml 形式のタスク列を作成する。ROS2-TMS for Construction においては Groot から出力された xml 形式のタスク列のファイル名をパラメータ指定すると自動でメタデータを付与した上で DB 上の所定の位置にタスクデータを作成する ROS2 ノードが実装されている。生成されるタスクデータの仕様を表 3 に示す。

7 動的パラメータ・静的パラメータ

動的パラメータ・静的パラメータについての説明を表 4 に示す。ROS2-TMS for Construction のタスク管理機構においては TMS_SP を介してセンシング処理後のパラメータデータをシステム外部から受け取り、TMS_DB を介して DB 上の動的パラメータデータの書き換えを行う。保管した動的パラメータの値は Beahavior Tree がサブタスクを実行する際に必要に応じて任意のタイミングで TMS_TS を介して参照する。静

Table 3 Specification of task data to be created

要素	格納されるデータ	パラメータの値指定
_id	MongoDB によってデータ格納時に固有 ID を付与	不必要 (自動)
model_name	タスク列から zx120/zx200/ic120 の単語出現頻度を元に最多のものを文字列として付与	不必要 (自動)
description	ros2 ノード実行時にパラメータとして指定	任意 (指定しないと”No Commented.”が格納される)
task_id	DB 上の既存のタスクデータに付与された task_id を 1 から昇順に検索し一番最初の空き番号を付与	不必要 (自動)
task_sequence	XML 形式のタスク列を文字列に変換して DB に格納	不必要 (自動)

的パラメータは予め事前に DB 上に書き込んでおくパラメータデータであり、動的パラメータのようにタスク実行中に書き換えられることはない。具体的には建機の初期姿勢などを格納する際に用いられる。

Table 4 Parameter types using task scheduler

パラメータ	動作概要
動的パラメータ	タスクスケジューラがサブタスクを実行している際にセンシング PC からのデータによって随時更新される動的なパラメータデータ
静的パラメータ	タスク実行中に書き換えられることのない静的なパラメータデータ

8 OperaSim-PhysX との連携による動作検証

OperaSim-PhysX 上の OPERA 対応建機を使用したタスク管理機構の簡易検証試験について紹介する。動作検証にあたっては図 7 に示すタスクツリーを作成し、タスク実行を行った。その様子の一部を、Groot を使用したタスク実行のモニタリング結果と併せてスナップショット形式で図 8 に示す。モニタリングにおいては実行しているタスクツリーが表示され、実行状況に応じて各々のノードの縁取りが変化する。具体的には未実行であることを示す灰色のノードが実行中は橙色に、実行後は緑色に変化する様子が示されている。

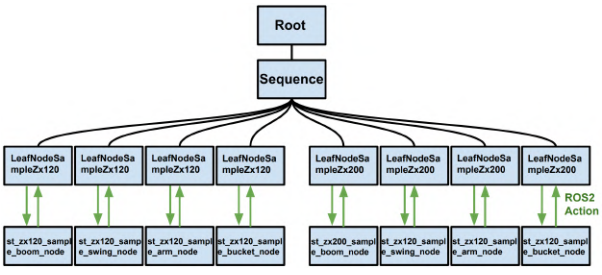

Fig.7 Task tree built for the test

9 結言

本文献においては ROS2-TMS for Construction のタスク管理機構の実装について紹介し、OperaSim-PhysX 上の OPERA 対応建機を使用して動作検証実験を行った。特にタスク管理機構を FSM から Behavior Tree へと移行することによって得られた知見には以下が挙げられる。

- Behavior Tree はツリー状にタスクを構築するため、状態の依存関係が増えた際でも FSM と比較して状態遷移がわかりやすい (可読性が高い)
- Behavior Tree が使用するツリー状のタスクは、ツリー状故に root ノードが存在し、別の木構造へつなげることが容易 (タスクツリーの汎用性が高い)
- Behavior Tree のすべてのノードはモジュール化されており、一つのタスクツリーで同じノードを何度も再利用可能 (ノードの汎用性が高い)

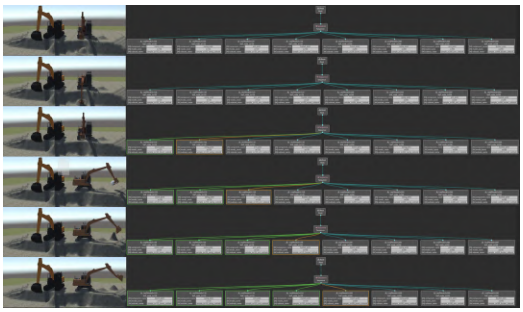

Fig.8 Snapshot when operating a machinery on OperaSim-PhysX

- タスクがツリー状に構築されているため、ステートマシンと比較してタスク構造の変更が容易 (再利用性が高い)
- Groot と呼ばれるタスク列作成のためのエディタ機能、タスク実行中の様子を監視するモニタ機能を有した便利ツールが存在する

なお、ROS2-TMS for Construction のタスク管理機構とは DB 上に作成したタスク列と、システム外部から送られてくる現場の動的な状態を表すパラメータ値をもとに OPERA に接続して建機の簡易的な運転を行うものであった。一方でこのタスク管理機構には建機のマニピュレーションやナビゲーションといった機能は搭載されていない。これらの機能は建機の自律運転を行うために必須の機能であり、ROS2-TMS for Construction においては図 1 における TMS_RP や TMS_RC のモジュールが担っている。これらのモジュールの機能については土工現場用 CPS プラットフォーム ROS2-TMS for Construction の開発の第 4 報 [7] として、清水建設・土木研究所と共同で実験した ROS2-TMS for Construction を使用した OPERA 対応建機実機による自律施工操作の第 1 回実証試験の結果と共に紹介する。

謝辞

本研究の一部は，内閣府総合科学技術・イノベーション会議の戦略的イノベーション創造プログラム（SIP）第 3 期「 スマートインフラマネジメントシステムの構築 」JPJ012187（研究推進法人: 土木研究所）によって実施されました。

参考文献

[1] 前田龍一，井塚智也，倉爪亮，“土工現場用 CPS プラットフォーム ROS2-TMS for Construction の開発”，日本機械学会ロボティクスメカトロニクス講演会 *2023*,1P1-B03, 2023.

[2] 前田龍一，高野智也，松本耕平，中嶋一斗，倉爪亮，“土工現場用 CPS プラットフォーム ROS2-TMS for Construction の開発-第 2 報 360 度カメラ映像を用いた CPS 可視化実験- ”，第 *24* 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, 1D1-10, 2023.

[3] 福田健太郎, 中嶋一斗, 倉爪亮，“転圧地盤評価のための分散型センサポッドの開発 -加速度応用スペクトルに基づく転圧評価手法の検討-”，第 *22* 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, 3H4-05, 2022.

[4] 高野智也，松本耕平, 中嶋一斗, 倉爪亮，“土木施工現場の状況把握のためのセンサポッド実証機の開発 ”，第 *24* 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, 1D3-02, 2023.

[5] 鈴木裕敬, 山内元貴, 遠藤大輔，橋本毅 “自立施工技術開発促進に向けた土木研究所の取り組み”，計測と制御, 第 61 巻 第 9 号, p. 651-655, 2022

[6] Ryuichi Maeda, Kohei Matsumoto, Tomoya Kouno, Tomoya Itsuka, Kazuto Nakashima, Yusuke Tamaishi, Ryo Kurazume, “ROS2-TMS for Construction: CPS platform for earthwork sites”, *29th International Symposium on Artificial Life and Robotics(AROB 29th 2024)*, pp., doi:, online, 2024

[7] 柴田航志, 笠原侑一郎, 井塚智也, 前田龍一, 高野智也, 松本耕平, 木村俊介, 深瀬勇太郎, 横島喬, 山内元貴, 遠藤大輔, 倉爪亮，“土工現場用 CPS プラットフォーム ROS2-TMS for Construction の開発-第 4 報 自律施工技術基盤 OPERA との連携-”，日本機械学会ロボティクスメカトロニクス講演会 *2024*, 2024.

[8] Tomoya Itsuka, Minsoo Song, Akihiro Kawamura, Ryo Kurazume, “Development of ROS2-TMS: New Software Platform for Informationally Structured Environment”, *ROBOMECH Journal*, Vol.9, No1, DOI:10.1186/s40648-021-00216-2, 2022

[9] IRVS, ROS2-TMS FOR CONSTRUCTION, https://github.com/irvs/ros2_tms_for_construction.git. （アクセス日 04/03/2024）