|  |  |
| --- | --- |
| 専攻分野 | 工学 |
| 専攻区分 | 情報工学 |

テーマ名：二輪ロボット制御システム開発への  
モデルカタログ導入の実践研究

氏名：齋藤理

－目次－

[研究背景と目的 1](#_Toc335389975)

[開発対象 2](#_Toc335389976)

[LEGO Mindstorms NXT 走行体 2](#_Toc335389977)

[走行制御ソフトウェアの概要 3](#_Toc335389978)

[目標制御モデル 5](#_Toc335389979)

[目標制御の概要 5](#_Toc335389980)

[ユースケース 7](#_Toc335389981)

[静的モデル 8](#_Toc335389982)

[動的モデル 10](#_Toc335389983)

[PSMの設計 11](#_Toc335389984)

[旋回量制御 11](#_Toc335389985)

[尻尾制御 14](#_Toc335389986)

[まとめと今後の課題 16](#_Toc335389987)

[参考文献 17](#_Toc335389988)

1. 研究背景と目的

近年，大規模化・複雑化する組込みシステム開発においてソフトウェア設計品質の向上は大きな課題のひとつであり，Unified Modeling Language (UML)を用いたソフトウェアのモデリングが注目されている[1]．モデルとは，さまざまな現象を説明するために対象の特徴を抽出し単純化して表現したものである．ソフトウェアにおいては，ソフトウェアの構造・振る舞いを説明するために単純化されたものといえる．モデルは単純化されているため，対象・現象の把握を容易とする．

モデリングの初心者にとって，システムの本質を捉えたモデルを作成するのは難しい．しかし，モデルはシステムの設計図の一部であり，競争領域にあるために手本となるようなモデルはなかなか出回らない．そのような状況を打開するためにUMLモデリング推進協議会(UMTP)より，モデリングのヒントとして，組込み分野向けの様々なモデルのカタログが提供されている．カタログで提供されるモデルは，プラットフォームに依存しないPlatform-Independent Models (PIM)であり，システムの設計図として容易に導入可能なものとなっている．

本研究では実践研究として，LEGO Mindstorms NXTによる二輪ロボット制御システムを開発する．ロボットには光センサ等の各種センサと，車輪及び走行体背面に装備された尻尾を駆動するサーボモータが装備されており，黒線の引かれたコースをライントレース走行する．走行制御に用いる輝度値制御や曲率半径制御に加え，尻尾の角度制御に対し，カタログに掲載されている目標制御のモデルを利用して開発を進める．

前述の通り，カタログにはPIMのみが掲載されるので，実際のシステム開発に導入する場合は対象とするシステムに合わせたPlatform-Specific Models (PSM)を設計する必要がある．本研究では，組込みシステム開発におけるモデルカタログ導入の有用性を検証し，モデルカタログに基づいたPSM設計の実例の提供を目的とする．

本報告書の構成は次の通りである．まず第2章では，開発対象とする二輪ロボット制御システムについて述べる．第3章では，今回カタログより導入する目標制御モデルについて述べる．第4章では，カタログ掲載PIMに基づき実際に設計したPSMについて述べる．第5章でこれまでの本研究をまとめ，今後の課題について述べる．

1. 開発対象

本章では，開発対象とする二輪ロボットと，それを制御するソフトウェアの概要について述べる．

* 1. LEGO Mindstorms NXT 走行体

本項では，使用する二輪ロボットの仕様と開発環境について述べる．

ロボット（走行体）の外観を図 1に示す．走行体は，LEGO Mindstorms NXT および LEGO ブロックで組み立てられる．メインプロセッサとしてARM7マイクロプロセッサが搭載される．4つの入力ポートと3つの出力ポートを持ち，光センサ，タッチセンサ，ジャイロセンサ，超音波センサが1つずつと，エンコーダ内蔵DCモータが3つ装備される．側面両側に取り付けられた車輪は，モータにより駆動する．また，背面にはモータにより上下移動が可能な尻尾が取り付けられ，尻尾を降ろして自立走行することができる．



図 1 走行体外観

開発には，オープンソースのLEGO Mindstorms NXT用開発，実行環境であるnxtOSEK[3]を用いる．nxtOSEKはC言語/アセンブリで記述されたI/Oドライバ及び，リアルタイムOS (RTOS)から構成される．nxtOSEKは以下の主要機能を提供している．

* gccツールチェインを使用したANSI C/C++言語開発環境
* NXTのモータ，センサ等のデバイスに対するリアルタイム制御用C言語API
* マルチタスクスケジューリング機能
  1. 走行制御ソフトウェアの概要

本項では，開発する走行制御ソフトウェアの概略について述べる．

走行体は，車体底部に搭載された光センサを用いて，黒線で描かれたレーンをリアルタイムで検出しながら自律走行する，ライントレースを行う．その際，走行体は尻尾を接地させた状態で走行する．

モデルカタログに掲載された目標制御システムのモデルを元に，走行体を制御するソフトウェアを開発する．目標制御の概要とカタログのモデルについては3章で述べる．

1. 目標制御モデル

モデルカタログ部品編[2]に掲載された目標制御モデルについて述べる．カタログで検討される目標制御システムは，特定の製品ではなく，どの製品でも共通となる抽象的な目標制御システムとして検討されている．本章のモデルは全て，カタログに掲載されているものである

* 1. 目標制御の概要

目標制御とは，制御対象の計測値がシステム側で設定する目標値に近づくよう制御する仕組みである．前述のPID制御は最も一般的な目標制御手法のひとつである．

目標制御の概要を図 2に示す．システムは計測器，操作器，制御器で構成され，一定時間周期で制御対象を計測しそれに応じた操作を行うフィードバックを繰り返す．制御対象は不特定な要因からも影響を受けるが，その都度入力を修正しフィードバックを繰り返すことで正確な制御を行う．

図 2 目標制御の概要

制御を開始すると，操作器の操作量が制御対象に伝わるまでの時間（無駄時間）を経過した後，計測値は目標値に近づいていく．このときの目標値に近づくまでの時間を立ち上がり時間と呼ぶ．計測値が勢いよく目標値に達した際，オーバーシュートが発生する場合がある．その後，ハンティングをしながら，計測値は徐々に目標値付近で安定する．最終的に安定した際の計測値と目標値の差が定常偏差として残ることになる．

目標制御システムで用いられる用語の定義を表 1に示す．

表 1 用語の定義

|  |  |
| --- | --- |
| 用語 | 用語の意味 |
| 目標制御 システム | 制御対象を制御するための仕組み |
| 制御対象 | 目標制御を行う対象 |
| 目標値 | 目標とする制御対象の値 |
| 計測値 | 計測された制御対象の値 |
| 外乱 | 制御対象に加えられる予測できない影響 |
| 制御器 | 目標制御を司る部品 |
| 入力器 | 利用者からの指定内容や指示を受け付ける部品 |
| 計測器 | 制御対象を計測する部品 |
| 操作器 | 何からのアクチュエータを駆動させることで、 制御対象を操作する部品 |
| 操作量 | 操作器がもつ指定可能な量で、結果的に制御対象を操作する量 |
| 制御方式 | 目標制御を実現する方式、アルゴリズム |

検討される目標制御システムの前提条件を示す．

* 検討する制御方式について  
  閉ループ制御を対象として，デフォルトの制御方式をPID制御とする．
* 一定周期で動作する仕組みについて  
  一定周期で動作させるためにタイマーを使用する．タイマーはシステムが起動したタイミングで開始する．
* 目標制御の終了条件について  
  制御対象を目標値に留めたい場合を考慮し，利用者が目標制御の終了を指示するまで，目標値に達しても制御を続ける．
* 目標値やその他の制御項目の変更タイミングについて  
  任意のタイミングで目標値やその他の制御項目を変更できるが，変更された値は次回の制御時から使用される．
  1. ユースケース

カタログのモデルが実現する範囲のユースケースを図3に示す．



図 3 目標制御システムのユースケース

ここで，アクター「外部システム」とは，最終的な製品に含まれる，目標制御を使用するプログラムを指す．表2にユースケース記述を示す

表 2 ユースケース記述

|  |  |  |
| --- | --- | --- |
| No. | ユースケース名 | 概要／備考 |
| UC01 | 起動する | 目標制御システムを起動する。 |
| UC02 | 制御項目を設定する | 目標値やその他の制御項目を変更する。  操作量の計算途中にも制御項目を変更できます。 |
| UC03 | 目標制御を開始する | 制御対象の計測値が目標値となるように制御する。  一定時間周期で制御対象の計測と操作を繰り返すことで、  制御対象は徐々に目標値に近づいていく。 |
| UC04 | 目標制御を終了する | 制御対象の操作を止める。 |
| UC05 | 停止する | 目標制御システムを停止する。 |

* 1. 静的モデル

静的な構造として，目標制御を行う為に必要な構成部品を図4のクラス図に示す．



図 4 目標システムのクラス構造

* 制御器クラス  
  外部システムが実際に使用するクラスとして「制御器クラス」が定義される．制御項目（目標値，制御パラメータ，リスナー，制御範囲）の設定と取得を行う操作と，制御の開始や終了を行う操作を持つ．また，タイマーを起点として，計測，算出，操作の一連の動作も行う．
* 制御対象クラス  
  実際の制御対象の現在の状態の保持やそれに関する情報を保持するクラスとして「制御対象クラス」が定義される．制御対象クラスは属性として目標値と計測値を持つ．尚，「計測値」属性は実際には操作であり，操作を呼ぶと計測器クラスの「計測する」操作を呼ぶ．一般的に，制御対象やセンサの測定限界として上限値・下限値が存在する．
* 範囲クラス  
  そのため，制御対象クラスは「範囲クラス」を保持する形となる．計測値をチェックし，範囲外になった場合リスナーを使用して通知する．これにより，外部システムは制御の中止やユーザへの通知を行うことができる．
* 計測器クラス  
  実際の制御対象に対して計測を行うクラスとして「計測器クラス」が定義される．実際にはドライバ等となる．
* 操作器クラス  
  実際の制御対象に対して操作を行うクラスとして「操作器クラス」が定義される．実際にはドライバ等になる．
* 制御方式・制御パラメータクラス  
  目標値，計測値，および，制御パラメータから操作量を算出するクラスとして「制御方式クラス」が定義される．制御方式クラスはPID制御等の制御方式を抽象化した抽象クラスとなる．制御方式クラスが操作量を算出する際には，元となる制御パラメータが必要になるため，制御方式クラスは「制御パラメータクラス」を保持する．制御方式クラスでは操作量算出時に必要な内部データが必要だが，実際の制御方式に依存するためここでは抽出されていない．
* タイマークラス  
  制御は定期的に行う必要があるため，「タイマークラス」という抽象クラスが定義される．制御器クラスがそのサブクラスとなることで，タイマーが間接的に目標制御システムの持つ操作を使用する形となっている．
  1. 動的モデル

計測してから操作するまでの間に，目標制御の終了が要求された場合やリスナーの操作の中で終了要求された場合の為に，「操作する()」を実行する直前に再度，制御状態を確認する．図5にシーケンス図を示す．



図 5 制御中の相互作用

1. PSMの設計

3章で述べたカタログの目標制御モデルに基づき，開発する走行制御ソフトウェア向けに設計したPSMについて述べる．車輪モータ回転速度算出のための輝度値目標制御，曲率半径目標制御，そして，尻尾角度制御目標制御の3つのモデルを作成し，実装した．

* 1. 車輪モータ制御

API仕様より，車輪モータの回転速度はパルス幅変調 (PWM)により制御される．このときのPWM値を車輪モータ制御量と呼び，符号付き8ビット整数型で表わされる．また，車輪モータ制御量は，それぞれ符号付き8ビット整数型で表わされる前進量と旋回量から次の数式によって求めることができる．

このとき，前進量は-100から100の間で設定した値を用いる．数値の絶対値が大きいほど移動速度は速くなり，符号が正の場合は前進方向へ，負の場合は後退方向へ移動する．

旋回量を求める方法として，輝度値制御と曲率半径制御の2つが挙げられる．

* 輝度値制御  
  ライントレースを実現するために，黒線での路面輝度値を目標値，現在の路面輝度値を計測値としたPID制御方式による目標制御を行い，操作量を算出する．
* 曲率半径制御  
  カーブを追従するために，カーブ毎の曲率半径を目標値，計測時の単位距離当りの車体角度の変化量を計測値としたPID制御方式による目標制御を行い，操作量を算出する．

PID制御について簡潔に述べる．P, I, DはそれぞれProportion（比例），Integral（積分），Differential（微分）の略で，偏差（目標値と現在値の差），偏差の時間積分，偏差の時間微分のそれぞれに定数を乗じたものを足し合わせたものを操作量とする．

このときの定数をそれぞれ比例係数，積分係数，微分係数という．

各制御における用語と数値のデータ型を表3に示す．

表 3 用語の定義

|  |  |  |
| --- | --- | --- |
| 用語 | 輝度値制御 | 曲率半径制御 |
| 制御対象 | 輝度値 | 曲率半径 |
| 目標値 | 黒線の輝度値 | 走行するカーブの曲率半径 |
| 計測値 | 光センサ入力値 | 単位距離当りの車体角度の変化量 |
| 操作器 | サーボモータ | サーボモータ |
| 操作量 | 旋回量 (PWM) | 旋回量 (PWM) |
| 制御方式 | PID制御 | PID制御 |

実際の走行制御ソフトウェアにおいては，輝度値制御を基本とした上で，曲率半径制御を用いてカーブに追従するよう旋回量を補正する．両目標制御より得られる操作量は重み付けされ，その和を旋回量として使用する．そのため，PIMの目標制御クラス構造とは異なり，PSMにおいては目標制御システム外部に操作器（車輪モータ）を配置する構造とした．制御器は「目標制御を実行する」操作を呼ばれると，一周期分の目標制御処理を行い，算出される操作量を返り値とする．外部システムは2つの操作量を受け取り， 前述の方法で旋回量を算出し，前進量と合わせて導出される各車輪モータ制御量を操作器へ送る．

また，いずれの制御においても，PID制御のみを制御方式として扱うので，制御方式クラスを介さずに直接PID制御方式クラスとPID制御パラメータクラスを抽出した．

以上を踏まえ設計したPSMを図9, 図10に示す．なお，制御器の周期的な起動にはnxtOSEKのタスク管理スケジューラを用いるため，タイマーはモデルから削除した．



図 6 輝度値制御のクラス構造



図 7 曲率半径制御のクラス構造

* 1. 尻尾制御

走行体背面には尻尾が取り付けられ，車輪と同様に図8に示すサーボモータで制御される．走行体は，尻尾を用いて自立しながら走行する．尻尾の角度を常に一定に保つ為に，自立時の尻尾角度を目標値，現在の尻尾角度を計測値とした目標制御を行い，操作量を算出する．

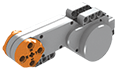


図 8 サーボモータ

操作器はサーボモータ，計測器はサーボモータ内蔵のロータリーエンコーダを用いて実装する．

API仕様より，モータの回転速度は符号付8bit整数のパラメータで制御される．このパラメータをモータ操作値と呼ぶ．このとき，符号は回転方向を表わし，絶対値が大きい程高速で回転する．

また，ロータリーエンコーダにより計測されるモータ回転角度は，符号付32bit整数型で取得される．

尻尾角度制御における目標制御システムの用語の定義を表4に示す．

表 4 用語の定義

|  |  |
| --- | --- |
| 用語 | 尻尾角度制御 |
| 制御対象 | 尻尾角度 |
| 目標値 | 設定角度 |
| 計測値 | 現在の尻尾角度 |
| 操作器 | サーボモータ |
| 操作量 | モータ制御値 (PWM) |
| 制御方式 | PI制御 |

設定角度と現在の尻尾角度との差を偏差として用いて制御を行う．制御対象となる尻尾角度は，走行中の目標値の変更頻度は高くなく，加えて外部の影響も受けにくいので，変化に対する反応性は要求されない．そこで，PID制御からD制御を除いたPI制御を制御方式として用いるものとした． 次に示す数式により操作量を算出する．

以上を踏まえ，目標制御PIMに基づく尻尾角度制御のクラス図を設計した．作成したクラス図を図9に示す．旋回量制御と同様に，制御器の周期的な起動にはnxtOSEKのタスク管理スケジューラを用いるため，タイマーはモデルから削除した．



図 9 尻尾角度制御のクラス構造

実装した尻尾角度制御について，目標角度を90度から一定時間55度に変更し，再び90度に戻した際の尻尾角度の変化の様子を図10のグラフに示す．

図 10 尻尾角度［度］の変化

グラフより，計測値が設定された目標角度に近づく様子が見てとれる．

1. まとめと今後の課題

本研究では，モデル駆動での組込みシステム開発へのモデルカタログ導入の事例研究として，モデルカタログに基づくLEGO Mindstorms NXT二輪ロボットの走行制御ソフトウェアの開発を行ってきた．カタログに掲載されたPIMを元にPSMを設計した過程についてまとめた．

本報告書のまとめとして，開発のモデルカタログ導入からPSM設計までを考察する．

まず，カタログに掲載されたモデルは，プラットフォームに合わせて各属性や操作の型を変更するだけで，容易にPSM設計の基礎として用いる事ができた． また，実装した各モデルについても，要求する動作を満たす動作を行う事が確認できた．これより，カタログに掲載されたモデルは実用に十分耐えうる事がわかる．

今後は，同一PIMから別プラットフォーム向けのPSM設計の検証を行う為，ルネサス社製M16C/29グループマイコンボード搭載の車輪付きロボットキットDonkey向けPSMの設計と検証を行う．

**参考文献**

1. 経済産業省，組込みソフトウェア産業実態調査報告書，  
   http://www.meti.go.jp/policy/mono\_info\_service/joho/ESIR/
2. UMTP組込みモデリング部会，組込み分野のためのUMLモデルカタログ，  
   http://www.umtp-japan.org/themes/original2/pdf/embedded\_UML\_model\_for\_the\_field\_catalog\_v1\_100a.pdf
3. nxtOSEK，http://lejos-osek.sourceforge.net/jp/index.htm