大規模・複雑化した組込みシステムのための障害診断手法

事後 V&V の体系と要素技術

付録

2017 年 3 月

独立行政法人情報処理推進機構
Information-technology Promotion Agency, Japan

はじめに

モデルベース開発を行うことで、各工程で実環境に依らずに仕様や動作確認ができ仕様ミスや不具合の早期発見・解消が可能になる。大規模・複雑化するシステム、ソフトウェアをモデル化して記述し、モデル上で環境や条件の変更を加えてシミュレーションすることで障害発生時にその原因の迅速な分析に資すると期待できる。

当 WG では、提案する事後 V&V フレームワークに沿った障害原因診断手順をモデル上で実際に確認し、それによって障害原因を推定するためのサンプルシステムとなるシミュレーターを開発している。本付録では、このサンプルシステムについて解説する。

シミュレーターを用いた実行によって、ハザード分析で得た障害原因の仮説シナリオをモデルで検証し、そのシナリオに沿った故障注入やオペレータの操作を再現すること、また、不具合の修正によって事故を回避できることをモデルで事前確認すること、などが実施できる。

2014 年度のタンク水位制御を行う化学プラントシミュレーターに続き、2015 年度はロボット制御のシミュレーターを開発して下記の仕様を実現している。

- 設計（モデル動作）と実機動作との差異の監視
- オペレータが介入するシステム動作の模擬
- 事故原因の改修による効果の確認

今年度は昨年度開発したロボットの制御系に走行制御系を追加し、人間と機械の協調制御が行われるシステムのサンプルとしてハザード分析を行い、その結果を基に障害対策を充実することで安定な制御が実現できることを示した。

目次

図表目次

A-1　二輪倒立ロボットシステム開発の目的

ここでは事後 V&V 要素技術の検証のためのサンプルシステムとして作成した二輪倒立ロボットについて説明する。二輪倒立ロボットとは、ちょうど振り子を逆向きにした形のもので、ジャイロセンサーと二つのモーター、およびモーターに装着されている車輪と、モーター制御を行うコントローラーから構成される。二輪倒立ロボットは何も制御しないと倒れてしまうが、車体の傾きをジャイロセンサーから取得し、傾きを打ち消すために必要なパワーをコントローラーで計算して、その結果をモーター動力として車輪に伝える、という姿勢制御を行うことで自立を実現している。

この二輪倒立ロボットに対して 2015 年度は、ユーザーにより動作条件を設定する機能、前進・後退あるいは左右に一定量の回転を行う機能､黒線を辿って走行するライントレース機能､を与えている。こうして作成したロボットシステムは、動作状態が変化する要素が複数存在するシステムとして、障害を注入し、事後診断の検証に用いることのできる教材となることを目的としている。

今年度はユーザー操作及び設定されたプログラムに従って左右両輪を個別に動作させる走行制御機能を追加した。こうして、人間と機械が協調して制御を行うサンプルシステムとなった二輪倒立ロボットに対して、STAMP/STPA によるハザード分析を行った。様々なハザード要因を摘出して安全対策を講じるという手順を繰り返すことで、安定した倒立走行制御を実現することができた。

2015 年度は、ET ロボコン等でも利用されている LEGO 社の Mindstorms NXT および同 EV3 という 2 種のロボットを作成に用いたが、その後 NXT が販売終了となったため、今年度は EV3 のみを用いて作成している。また、上述の制御には昨年度と同様に MathWorks 社の MATLAB/Simulink を用いている。以下、その詳細を記述する。

A-2　二輪倒立ロボットについて

(1)　ロボットの構成

二輪倒立ロボットは、ジャイロセンサーと二つのモーター、及びモーターに装着される車輪とコントローラーから構成される。本サンプルシステムでは、さらに物体との距離を測る超音波センサー、輝度を認識する光センサーを追加している。これは後述するライントレース及び衝突回避を実現するためのものである。

これらの組み立てには LEGO 社 Mindstorms EV3 を用いている。組み立て設計図は ET ロボコン向け設計図を参考にしているが、後部のロボットを支えるモーター、所謂「尻尾」をはずした構成としている。組立て後のロボットの外観を図 A-2-1 に示す。

(2)　OS

EV3 でコントローラーの制御プログラムを動作させるプラットフォームとなる OS はいくつかあるが、ここでは brickOS と呼ばれるフリーの OS を採用している。brickOS v1.08H、および Simulink を用いた制御プログラムの開発に必要なツールボックスは、LEGO 社のホームページ [1]から入手する。

(3)　PC との通信

EV3 と PC との通信には無線 LAN を用いている。これは後述する開発環境の制約によるものである。モジュールの選定及び設定の概要については後節で述べる。

[1] http://www.lego.com/en-us/mindstorms/?domainredir=mindstorms.lego.com

付録 図 A-2-1　EV3 全体図

A-3　ロボットの要求仕様

（1）　姿勢制御

ロボットの電源がオンで、制御ソフトが動作しており、かつ姿勢制御が有効な場合に、ロボットは自身が倒れないように制御を行う。ジャイロセンサーから得る車体角速度をもとに、ロボットの傾きを打ち消すようモーターに対して回転要求を出す。あわせて、モーターのエンコーダーから得る車輪回転角を元に、ロボットが指示された位置に戻るようモーターに対して回転要求を出す。上記２つの要求により、ロボットは姿勢制御有効時、指示された位置で倒れないように自立する。

（2）　ユーザーによる操作

ロボットの電源がオンで、制御ソフトが動作しており、かつ姿勢制御が有効である際に、ソフトウェアによるユーザー向け UI パネルあるいは PC と接続されたハードウェアであるコントロール PAD から前後それぞれの移動、左右それぞれの旋回、及びこれらの停止の命令を受け付ける。UI パネルより前後それぞれの移動の命令を受けると、ロボットは指定された方向に 800ms 移動する。また、左右それぞれの旋回の命令を受けると、ロボットは指定された方向に 350ms 旋回する。一方、コントロール PAD から前後の移動あるいは左右の回転を要求するボタン操作があった場合には、ボタンが押されている間は動作を継続し、ボタンを離すと停止する。いずれの場合にもこれらの動作は、左右のモーターに対しての目標回転角をそれぞれ変更することで実現する。

（3）　ラインとレース

ロボットの電源がオンで、制御ソフトが動作しており、姿勢制御が有効かつラインとレースが有効である際に、ロボットは、地面の黒線を追いながら走行する。ロボット下部に設置された光センサーにより地面の輝度を計測し、そこから黒線か否かを判断する。黒線を検知した際は右に、検知できなかった場合は左に旋回することで、ロボットは黒線の右端を辿る。またラインとレースが有効である場合、ロボットは常に前進する。ユーザーの操作は受け付ける。

（4）　プログラム走行

今年度追加された動作モードである。ロボットの電源がオンで、制御ソフトが動作しており、姿勢制御が有効かつ UI パネルから前後、楕円走行、方形走行のいずれかの走行動作が指定された場合に、ロボットは指定された形状と大きさの軌跡を描いて継続走行する。

（5）　衝突防止

ロボットの電源がオンで、制御ソフトが動作しており、姿勢制御が有効かつ障害物検知が有効であ

る場合、ロボットは前方に障害物を見つけると、その距離に応じて次の二つの動作のいずれかを行う。ユーザーによる操作、プログラム走行、及びライントレースによる前進命令がある場合にも、23cm 以内に物体を検知するとロボットは前進命令をキャンセルし、その位置に留まるよう動作する。10cm 以内に物体を見つけると、ロボットは衝突防止のため一定速度で後退する。ロボット上部に設置された超音波センサーにより、常時前方との距離を測りながら、上記処理を行う。この 10cm 以内に物体を見つけた際の処理は、ユーザーによる操作、ライントレース、及びプログラム走行による前進命令より優先される。ユーザーによる後退操作は制約されない。

（6） ロボットの情報出力

ロボットの電源がオンで、制御ソフトが動作している際に、ロボットは自身のディスプレイにセンサー値やコントローラー出力値を逐次表示出力する。

（7） PC による監視

ロボットの電源がオンで、制御ソフトが動作している際に、ロボットは PC に対して常時センサー値を初めとする各種データを表示する。PC 側では受信したデータの表示と制御ソフトの一部設定値の書き換えを行うことができる。また、プログラムを停止した際には、受信データをログデータとして保存する。事後診断用のプログラムを起動することにより、ログデータに対して 逐次確率比検定(SPRT)による分析を行う。

A-4 ロボットのハザード分析

ロボットの要求仕様をもとに、STPA によるハザード分析を行った。以下にその結果を示す。

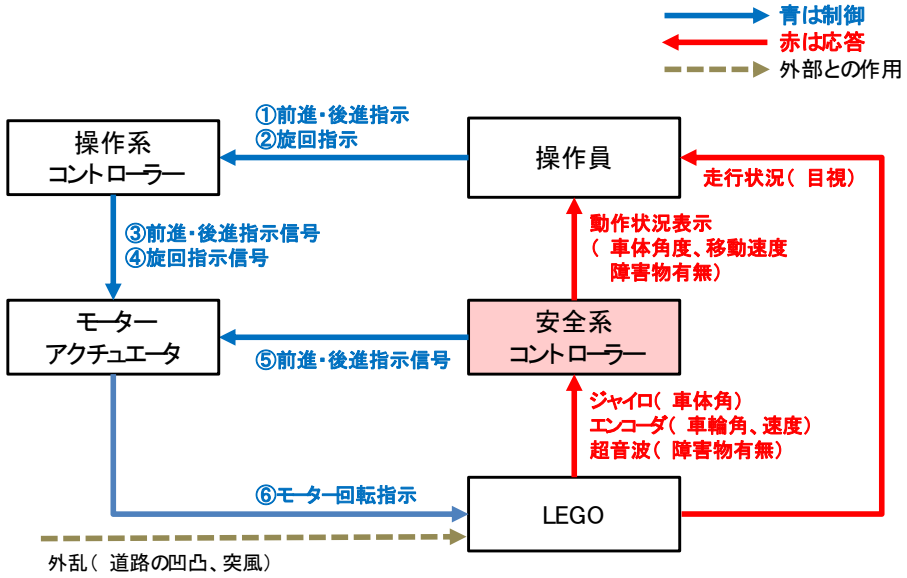
（1） アクシデント、ハザード、安全制約の識別

STPA を行うための事前準備として、アクシデント、ハザード、安全制約を次のように定義した。この際、トレーサビリティを担保するために、アクシデントには[A-1],[A-2]、ハザードには[H1-1],[H1-2]…を、安全制約には[SC1-1][SC1-2]…のように番号付けを行った。ハザード、安全制約における先頭の数値は、それぞれ対応するアクシデントの番号である。

アクシデント
[A-1] ロボットが転倒する
[A-2] ロボットが障害物にぶつかる
ハザード
[H1-1] ロボットの重心が制御可能域を逸脱する
[H2-1] 障害物を検知しても停止が間に合わない
安全制約
[SC1-1] ロボットの重心を常に制御可能域に抑える
[SC2-1] 衝突防止策が間に合う範囲で障害物を検知する

（2） コントロールストラクチャーの作成

次に、ロボット、および動作環境を制御構造図によりモデル化する。この際、ロボットの動作環境を次のように仮定した。即ち、『「操作員がパッドコントローラー、もしくは UI パネルを用いてロボットの移動を制御する。操作員はロボット（LEGO）が常時見渡せる環境にいる。あわせて、操作員にはロボットを移動させ、任意の目標地点に移動させ元の位置に戻すというミッションが与えられている。目標地点の周辺には段差、障害物が存在している』。作成した制御構造図を図 A-4-1 に示す。



付録 図 A-4-1　二輪倒立ロボット EV3 の制御構造図

操作員からコントローラーを通じて前進・後退指示、および旋回指示がモーターアクチュエーターを通じてロボットに出される。一方でロボットの姿勢制御、衝突防止制御は安全系コントローラーからモーターアクチュエーターを通じてロボットに出される。モーターアクチュエーターでは２つの指示値の和をロボットに伝えている。ロボットからは安全系コントローラーを通じて操作員に動作状況を表示しフィードバックしている。併せて、ロボット自身の動作状況を捜査員は目視で確認できる。ロボットに対しては道路の凹凸、障害物といった外乱が加わる。このような、動作指示とその動作結果のフィードバックでロボットをとりまくシステムは構成されている。

（3）　非安全な制御動作(UCA)の抽出

制御構造図にて表されたコントロールアクション(制御指示)に対して、4 つの場合を考えてそれが安全制約を侵害しうる非安全な制御動作(UCA)とならないか、を図 A-4-2 に示す表で検証した。ここでも、トレーサビリティを担保するために、コントロールアクションへの番号付け、および仮に UCA が抽出された場合、それがどの場合に起こりうるか(Not Providing、Providing causes hazard ： Incorrectly Providing、Timing ： Too early/Too late、Duration ： Stop too soon/Applying too long)に応じて、それぞれ UCAx-N、UCAx-P、UCAx-T、UCAx-D と採番している。

#	コントロールアクション	FROM	TO	Not Providing	Incorrectly Providing	Too early / Too Late	Stop too soon /Applying too long
1	前進・後進指示	操作員	操作系コントローラ	該当なし	(UCA1-P)LEGOが不安定な状態での前進・後進指示による転倒 (HC1-1違反)	該当なし	該当なし
2	旋回指示	操作員	操作系コントローラ	該当なし	(UCA2-P)LEGOが不安定な状態での旋回指示による転倒 (HC1-1違反)	該当なし	該当なし
3	前進・後進指示	操作系コントローラ	モーターアクチュエーター	該当なし	(UCA1-P)LEGOが不安定な状態での前進・後進指示による転倒 (HC1-1違反)	該当なし	該当なし
4	旋回指示	操作系コントローラ	モーターアクチュエーター	該当なし	(UCA2-P)LEGOが不安定な状態での旋回指示による転倒 (HC1-1違反)	該当なし	該当なし
5	前進・後進指示	安全系コントローラ	モーターアクチュエーター	(UCA3-N)LEGOの重心を制御できず転倒 (HC1-1違反) (UCA4-N)障害物前で停止できず衝突 (HC2-1違反)	(UCA3-P) LEGOの重心を制御できず転倒 (HC1-1違反) (UCA5-P)障害物前で前進してしまい衝突 (HC2-1違反)	(UCA3-T) LEGOの重心を制御できず転倒 (HC1-1違反) (UCA4-T)障害物前で停止できず衝突 (HC2-1違反)	該当なし
6	モーター回転指示	モーターアクチュエーター	LEGO	(UCA3-N)LEGOの重心を制御できず転倒 (HC1-1違反)	(UCA3-P) LEGOの重心を制御できず転倒 (HC1-1違反) (UCA5-P)障害物前で前進してしまい衝突 (HC2-1違反)	(UCA3-T) LEGOの重心を制御できず転倒 (HC1-1違反) (UCA4-T)障害物前で停止できず衝突 (HC2-1違反)	該当なし

付録 図 A-4-2　二輪倒立ロボット EV3 の UCA

一例として、[UCA1-P]を見てみると、操作員から操作系コントローラーへの前進・後進指示が誤ってなされた場合（Incorrectly Providing)、ロボットの状態が不安定な状態でこれらの指示を出すとさらに不安定になり、安全制約「SC1-1: ロボットの重心を常に制御可能域に抑える」に違反する可能性が出てくる。従って UCA に該当すると分析した。同様に、他のケースについても分析を行っている。

(4)　ハザード誘発要因(HCF)の特定

導出された非安全な制御動作がなぜ発生するのか、ハザード誘発要因(HCF)を特定する。先の UCA 抽出では制御構造図の制御動作に着目したが、ここでは関係するコントローラーに対してのフィードバックも考慮する。一例として前記の[UCA1-P]について考える。操作員は安全系コントローラーからの動作状況表示と、ロボット自身の状況を目視で、それぞれ監視することができるにもかかわらず、なぜロボットが不安定な状態にありながら操作員から操作系コントローラーに対して前進・後退指示を出すのか。一因として、操作員がロボット自体の走行状況を注視せず、表示情報のみを見ていたことが考えられる。あるいは情報表示に遅延が生じて、実際にはロボットが不安定な状況になっているにもかかわらず、ロボットは安定していると思い込む可能性が考えられる。もしくは、操作員はロボットが不安定な状況であることはわかっていたが、これを是正しようと操作し、悪影響を及ぼした可能性も考えられる。このようにして、それぞれの UCA に対してハザード誘発要因を導出した。

(5)　安全対策のまとめ

導出した HCF を防ぐために、システムに対して新たな安全制約を設けた。その一部を下記する。

- 操作要求信号に対して、最大速度と最大加速度、加加速度（Jerk）を制限する
- ジョイスティックから手を放すと停止する安全設計
- ルールベース制御（ロボットが不安定な時は操作系コントローラー入力を受け付けない）の採用
- 衝突防止時/ルールベース制御時にロボットから音を出して、操作員に衝突防止が働いていることを知らせる

ロボットの制御仕様は、これら安全制約を反映した状態となっている。

A-5　ロボット制御系の開発環境

二輪ロボットの制御は PID 制御によって行う。PID 制御系の制御定数の初期値は、Mathworks 社が

NXTWay-GS [1] を対象に行った方法を参考に、EV3 の動特性モデルから得た。EV3 は NXT と比べると質量、寸法等、いくつか異なるパラメータを持つが、その動特性は同様であることから、得られた制御定数はほぼ同じ値となった。しかしながら、今年度新たに走行制御系を加え、段差を乗り越える等の試験を行った結果、姿勢制御を更に安定化する必要があったことから、最終的には限界感度法等を用いて係数値を求めている。

本サンプルシステムでは、ロボットの制御系モデルの実装プログラムの作成、及び動作時の監視を同一環境で行っており、これを MATLAB/Simulink 2015b 及び Simulink toolbox for LEGO Mindstorms EV3 により実現している。後者のツールボックスは、Simulink に対して LEGO Mindstorms EV3 向けモデリング、及びエクスターナルモードによる実行を実現するために用意されたものである。

エクスターナルモードによる実行により、モデル実行時に PC による EV3 の各種データの取得、及び PC からのパラメータ変更が可能となる。ツールボックスの制約により、エクスターナルモードの実施には PC と EV3 間での無線 LAN 通信が必要となる。この際、EV3 側の無線 LAN 対応モジュールが限定されている点に注意が必要である [2] 。さらに、EV3 側の OS version は 1.08H である必要がある [3] 。

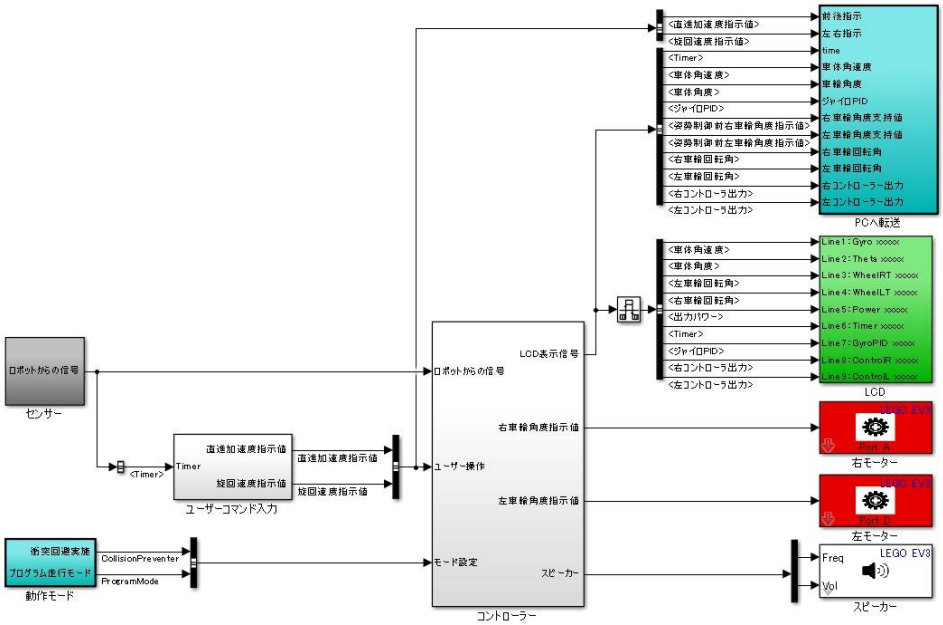
PC 側では無線 Lan モジュールとして Buffalo WLI-UC-GNM2 を利用している。本ツールボックスにて無線 Lan を利用する場合は、アドホック接続が不可能で、DHCP による IP アドレス割り当てが必要であり、無線ルーターが別途必要となる。また、通信時の暗号化は EV3 の制約から WPA2 若しくは非暗号化による通信のみ可能である。今回は、Windows7 の Soft AP を用いて無線ルーター機能を実現した。

A-6　ロボット制御系の構成

ここでは事後 V&V 適応例として作成した二輪倒立ロボットの制御系について説明する。先ず、ロボット中で動作する Simulink モデルを示す。次に、PC 側の Simulink モデルで、動作中に保存されたログデータをもとに事後診断を行うプログラムと、ロボットへの動作指示と動作条件の変更を行う UI パネルの構成を示す。

A-6.1　ロボット側 Simulink モデル

Simulink を用いて作成した二輪倒立ロボットの制御系モデルの全体構成を図 A-6-1 に示す。



付録 図 A-6-1　二輪倒立ロボット EV3 の Simulink モデル 全体構成

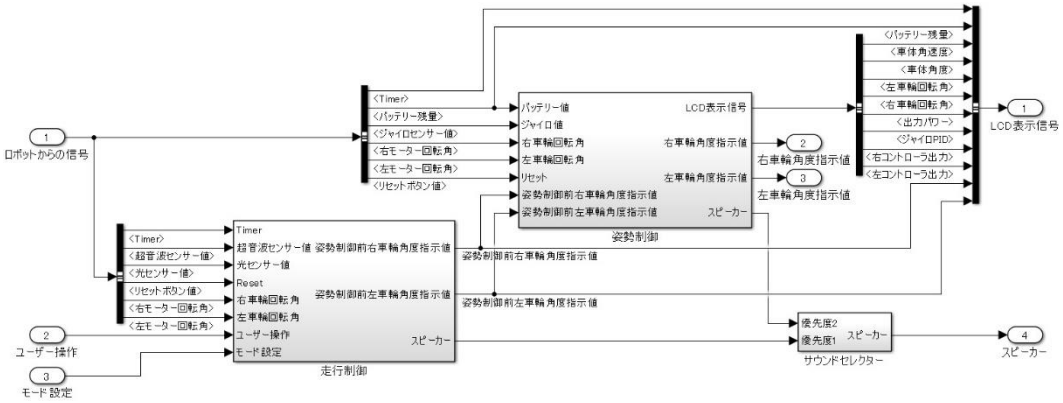
[1] http://www.mathworks.com/matlabcentral/fileexchange/19147-nxtway-gs--self-balancing-two-wheeled-robot--controller-design

[2] EDIMAX EW-7811Un の他、Planex GW-USVALUE2 にて動作を確認している。両者とも搭載チップは Realtek RTL8188CUS ではあるが、同チップを搭載している Planex GW-USNANO2 では本事例環境では動作不可であった。

[3] EV3 の OS version が 1.07H 以下の場合、対応無線 LAN モジュールが販売終息品の 1 種類のみとなる。また、1.09H にて EV3 側で SSH, telnet 通信をブロックするように変更されている（リリースノート非公開のため推測である）。これが原因で、ping は通るがツールボックスと EV3 との通信ができない現象が生じる。

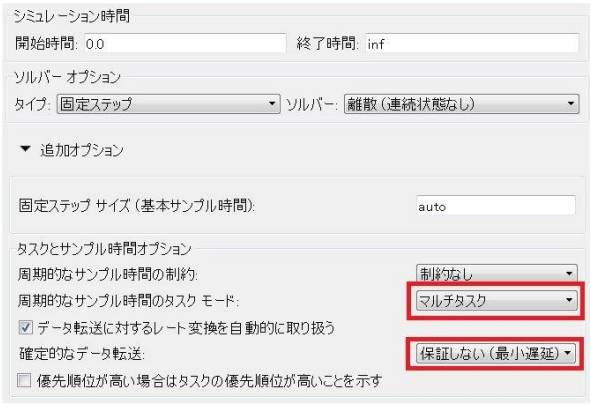
モデルは、ロボットのセンサー値を取得する「センサー」部と、センサー値を用いた姿勢制御とユーザーの指令により走行制御を行う「コントローラー」部、及び姿勢制御のために計算された動力をロボットのモーターへ伝える「モーター」から構成される。さらにはロボットのディスプレイ画面及び PC へ情報を伝達するための「LCD」、「PC へ転送」部のほか、ユーザー操作を受ける「ユーザーコマンド入力」部が存在している。

「コントローラー」部は図 A-6-2 に示す内部構成を持ち、「姿勢制御」部において特殊な制御が行われた場合、あるいは「走行制御」部において衝突防止動作が行われた場合、スピーカーへのサウンド出力を行う機構を有している。これらについては後述する。



付録 図 A-6-2　「コントローラー」部の内部構成

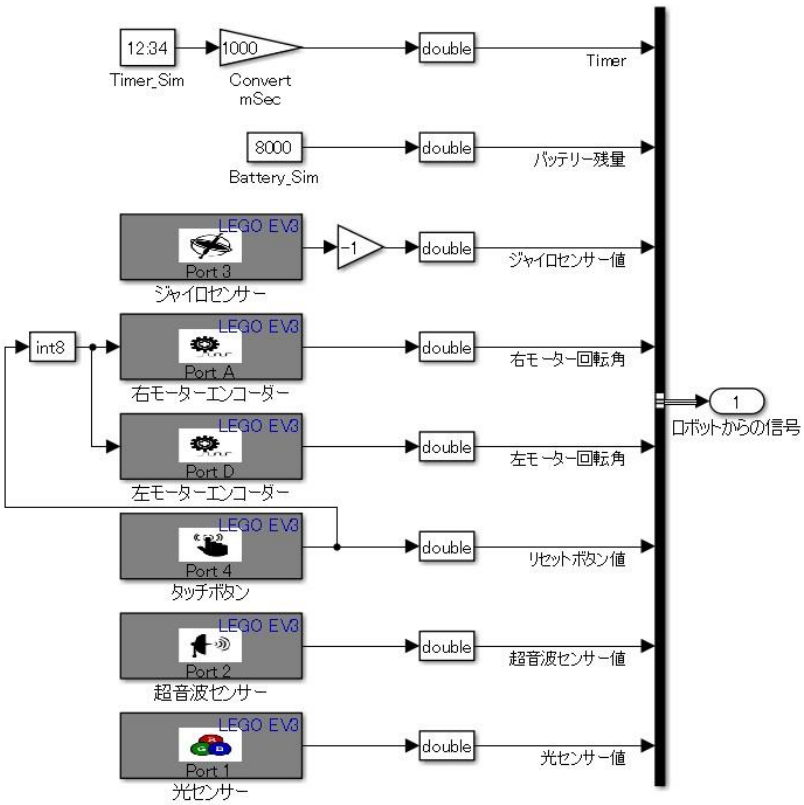
本モデルの実行にあたっては、Simulink のモデルコンフィグレーションパラメータの設定で、図 A-6-3 に示すように、ソルバーを「固定ステップ」「離散(連続状態なし)」、基本サンプル時間を「auto」、周期的なサンプル時間のタスクモードを「マルチタスク」としている。また、EV3 の動作負荷が従来の NXT に比べて重い現象が見られたため、確定的なデータ転送を「保障しない(最小遅延)」としている。



付録 図 A-6-3　モデルコンフィグレーションパラメータ

(1) センサー部

「センサー」部は図 A-6-4 に示すように、ロボットに備えられた各種センサーに対応するブロックから構成される。



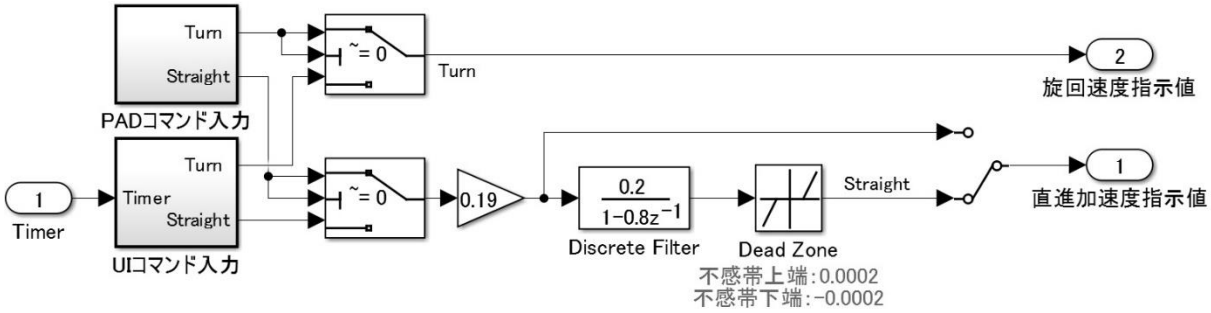
付録 図 A-6-4 「センサー」部の構成

EV3 は、従来の NXT で取得できた時間、電源電圧が取得できないため、ここではそれぞれモデルのシステム時間、固定値を用いている。タッチセンサーは EV3 の動作の開始/終了のスイッチとして用いており、カラーセンサーはライントレースに用いている。ジャイロセンサーの取り付け方向の問題から、前方に傾いた際にプラスの値となるように、出力値に「-1」を掛けて用いている。

姿勢制御に用いるジャイロセンサーとモーターエンコーダー、カラーセンサーは 10ms で動作するよう設定している。ただし、物体検知に用いる超音波センサーは、その特性から 10ms では誤動作する可能性がある。超音波センサーは仕様として 250cm (2.5m) まで物体を検知するが、センサーから出た超音波が帰ってくる時間からその距離を算出するため、2.5m 地点の物体を検知するには、音速 340m/s で約 15ms かかる計算である。センサー値取得はこれよりも遅い間隔で行う必要があるため、動作周期は余裕を持って 100ms としている。これ以外に、時間値はユーザー入力の基準、及び表示にのみ用いるため、100ms とし、電源電圧の値は定数値のため動作周期は「inf」(変化しない) としている。

(2) ユーザーコマンド入力部

図 A-6-1 においてコントローラー部へのユーザー操作入力を与える「ユーザーコマンド入力」部の構成を図 A-6-5 に示す。昨年度と同様に PC 側の Simulink プログラムで動作する UI パネルを介した「UI コマンド入力」と、今年度導入したハードウェアのコントローラーPAD を介した「PAD コマンド入力」の二通りの入力手段が用意されている。

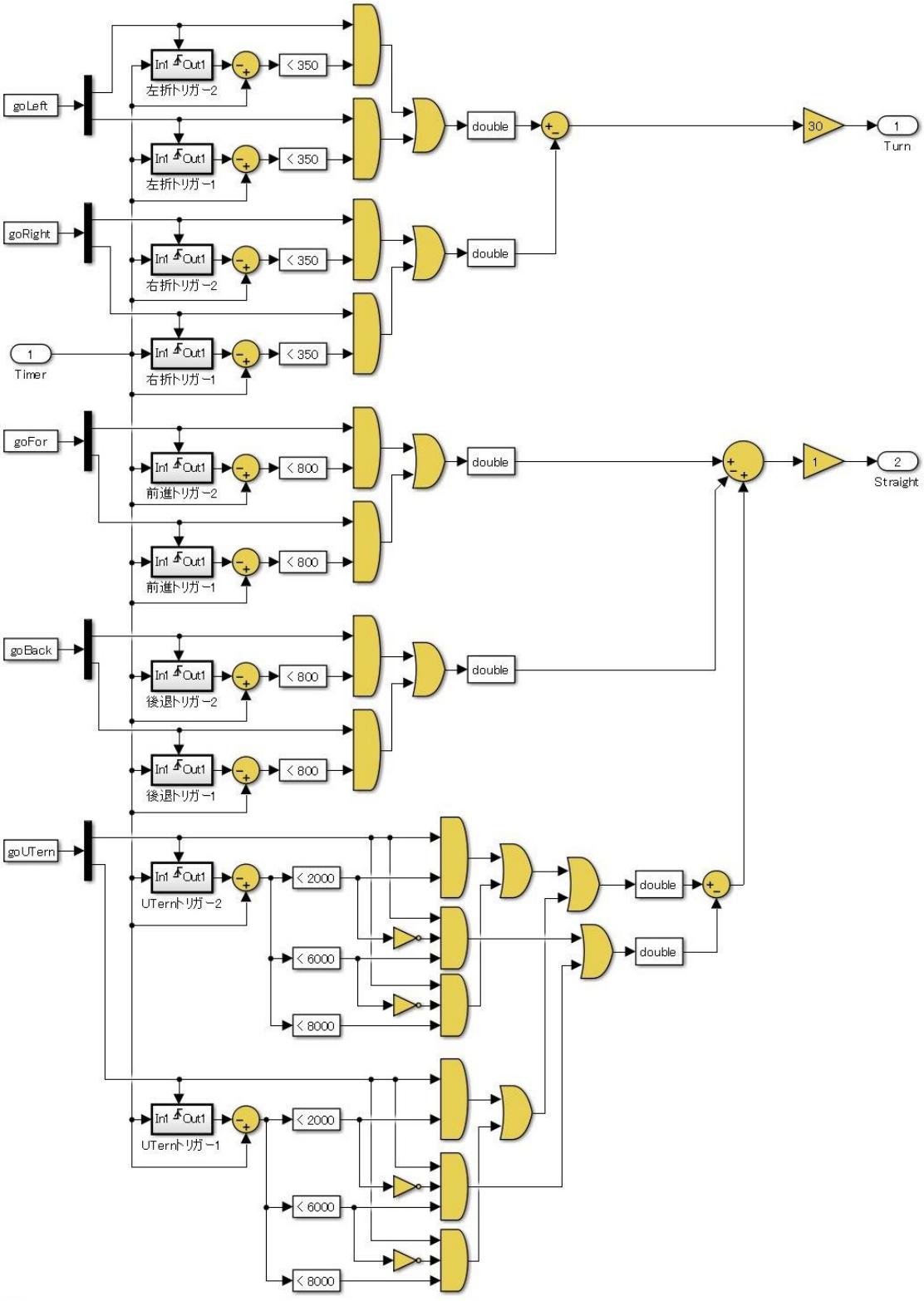


付録 図 A-6-5　ユーザーコマンド入力部の構成

図中に記されているように、何れの入力方法によっても、旋回指示（Turn）については速度、直進指示（Straight）については加速度として、それぞれ指示値が出力されるようになっている。直進指示を加速度で与える理由は、STPA によるハザード分析の結果、ロボットの転倒を防ぐためには前進・後退時の加速度を制限する必要のあることが明らかとなったためである。図中の「Discrete Filter」は、ユーザー操作によって加えられる急激な加速度変化を緩和するために挿入されたものである。

(1) UI コマンド入力

図 A-6-5 の「UI コマンド入力」部の構成を図 A-6-6 に示す。後出の UI パネル上で「左回転」、「右回転」、「前進」、「後退」、及び「前進・後退」ボタンを押すことにより、プログラム「UIPanel.m」において図の左端の定数ブロックに示されている該当変数の値が設定される。例えば初めて「左回転」ボタンを押した場合、図の左上に示された 2 次元変数「goLeft」の値が[0 0]から[1 0]に設定される。これにより、図中の「左折トリガー2」ブロックにボタンが押された時刻が保存され、それから 350ms の間、「30 度/s」の旋回速度指示（Turn）信号が出力される。さらにもう一度「左回転」ボタンを押すと、「goLeft=[0 1]」となり、今度は図の「左折トリガー1」ブロックにトリガーがかかって、同様に信号が出力される。このように 2 つのトリガーブロックを切り替えることで、連続的に操作指示を受け付けられるようになっている。



付録 図 A-6-6 「UI コマンド入力」部の構成

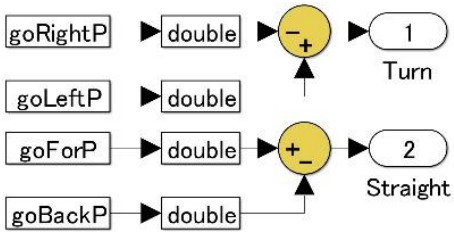
(2) PAD コマンド入力

「PAD コマンド入力」部の構成を図 A-6-7 に示す。

PAD コマンド(DirectInput)を Simulink モデルで用いるワークスペース変数に反映する処理は「getPadInput.m」にて行っている。こちらでは、DirectInput の値を定期的に監視し、値に変更があ

ればワークスペース変数の値を変更する処理を行っている。さらに、ハードウェアの PAD によっては十字キーとスティックキー入力を兼ね備えているものが存在する。両者から入力が同時に与えられた場合は、スティックキー入力を優先する処理としている。Simulink モデル側では現在ワークスペース変数に入っている最新の値を伝える処理となっている。

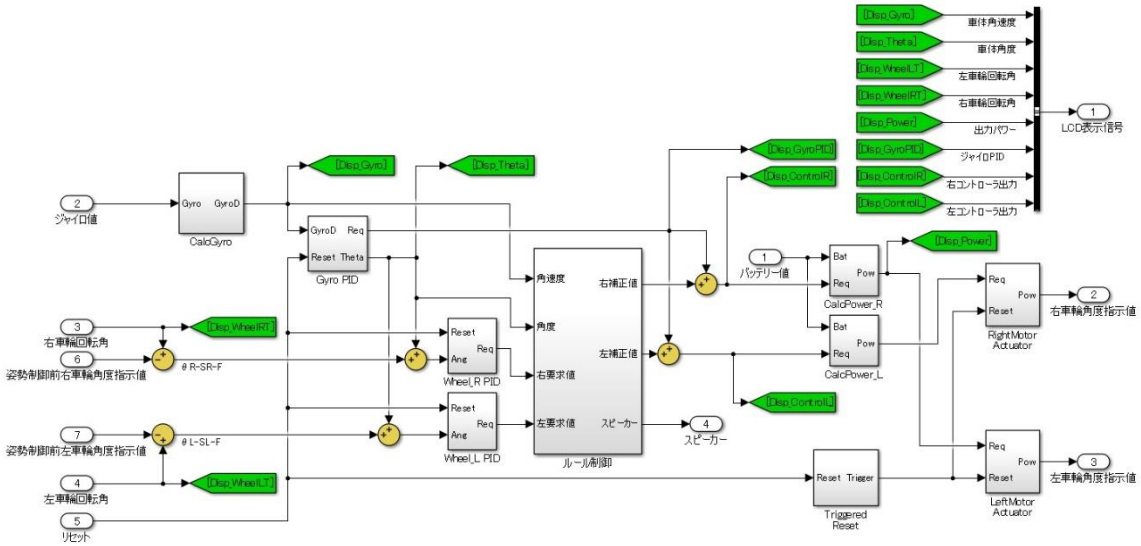
あわせて、図 A-6-7 に示す通り、PAD 入力と UI コマンド入力が同時に発生した場合には、PAD 入力が優先される処理としている。



付録 図 A-6-7　「PAD コマンド入力」部の構成

A-6.2　姿勢制御部

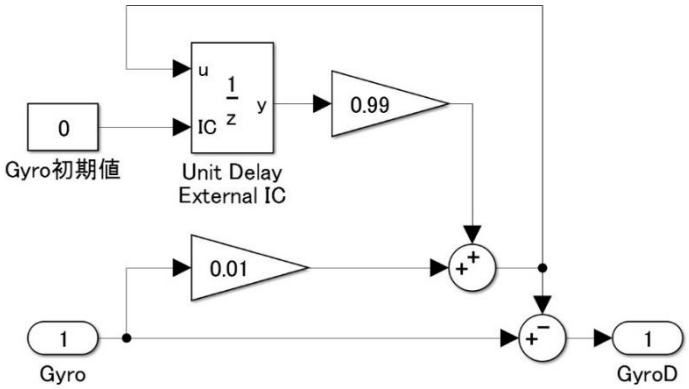
図 A-6-6 に示した「コントローラー」部において、姿勢制御を行う部分の構成を図 A-6-8 に示し、図中の各ブロックについてその詳細を以下に記す。



付録 図 A-6-8　「姿勢制御」ブロックの構成

（1）　ジャイロ補正

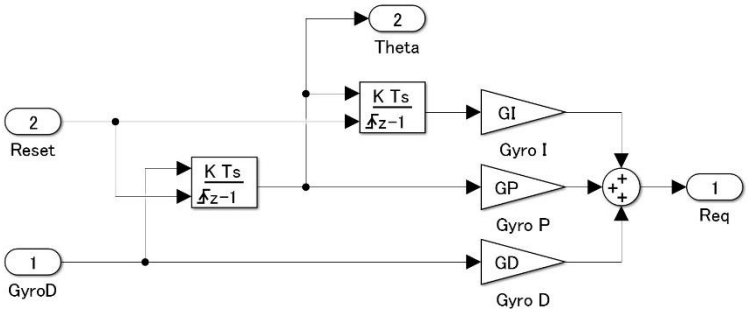
図 A-6-8 に示すモデルではジャイロセンサーから得られた値に対して「CalcGyro」ブロック内で図 A-6-9 に示す処理を行う。EV3 の場合はセンサーから得られる値は-480～480 deg/s の範囲で変化する。角速度が発生していないとき、即ちロボットが静止している際に得られる値は「0」前後となる。「CalcGyro」ブロックでは、オフセット初期値を「0」とし、以後センサーの平均値からオフセット値の更新を行う。そしてオフセット値との差分を角速度として出力する。



付録 図 A-6-9 「CalcGyro」ブロックの構成

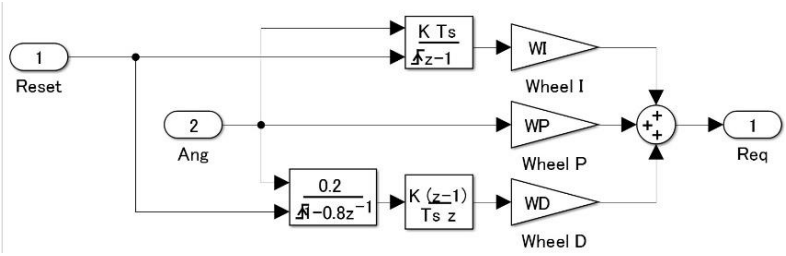
（2） PID 計算

「姿勢制御」部（図 A-6-8）の「Gyro PID」ブロックの構成を図 A-6-10 に示す。前述のジャイロ補正値の目標値「0」からの偏差、即ち補正値そのもの、を一度積分してロボットの傾斜角度に変換した信号に対して PID 制御値を計算するものである。制御定数 P/I/D の値は通常 1.1/0/0.07 としている。



付録 図 A-6-10 「Gyro PID」ブロックの構成

同様に、図 A-6-8 の「Wheel_L PID」及び「Wheel_R PID」ブロックにおいては、それぞれ左右の車輪回転角度に上記傾斜角度を加算した信号と、それぞれの回転角度指示値との偏差に対して、図 A-6-11 に示すように PID 制御値を計算する。制御定数 P/I/D の値は通常 0.0145/0.0078/0.023 としている。



付録 図 A-6-11 「Wheel PID」ブロックの構成

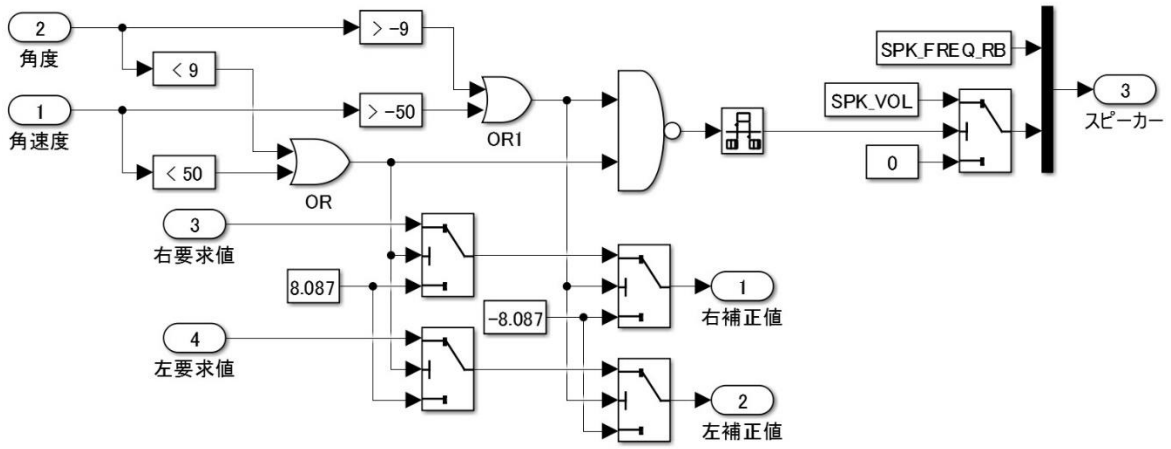
（3） ルール制御部

「姿勢制御」部（図 A-6-8）の「ルール制御」ブロックは、ハザード分析から導かれた対策案として導入されたものであり、その構成を図 A-6-12 に示す。ここでは EV3 の車体角度と角速度を判定条件に用い、以下のようなルールで前記左右「Wheel PID」の出力値を補正する。即ち、

『・もし 車体角度が 「9 度」 以上で、且つ 車体角速度が 「50 度/s」 以上なら、
左右両輪の PID 出力値を 「8.087」とする。

・もし　車体角度が　「-9 度」　以下で、且つ　車体角速度が　「-50 度/s」　以下なら、
　左右両輪の PID 出力値を　「-8.087」とする。
・上記以外の条件では補正を行わない。　』

ここで、「±9 度」と言う値は試行の結果定めたものである。また、「8.087」と言う数値は奇異な感じがするが、ルール制御部の後段のパワー計算部の出力が丁度「100」になるように与えるものである。後出のアクチュエーター部で±100 のリミッターを通すため、「8.087」以上であれば良い。



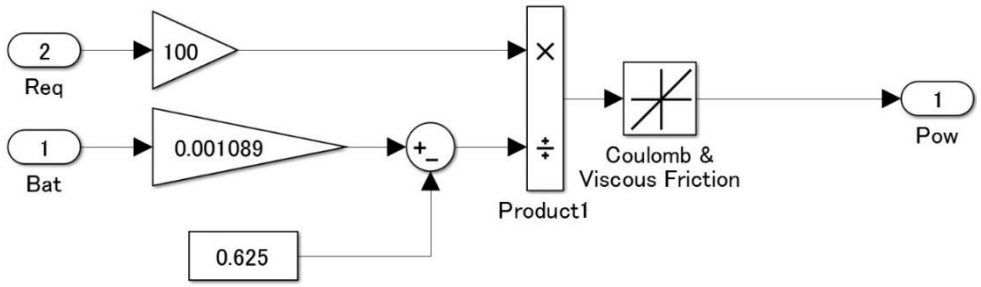
付録 図 A-6-12　「ルール制御」ブロックの構成

ルールの適用によって要求値が補正された場合には、スピーカーへの信号が出力される。

(4)　パワー計算

図 A-6-13 に示す「CalcPower」ブロックでは、ジャイロ側 PID 計算結果と車輪側 PID 計算結果との合計値に対して、バッテリー電圧で補正したモーターパワーを計算し、最後にロボットの設置面との摩擦を考慮して、モーターに与える値を-100～100 の範囲で出力する。

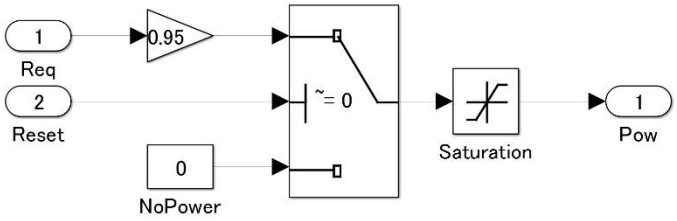
なお、センサー部の説明で述べたように、EV3 ではモーター電圧が測定できないことから、その変化は無視しており、また実際には摩擦も無視している。



付録 図 A-6-13　「CalcPower」ブロックの構成

(5)　アクチュエーター

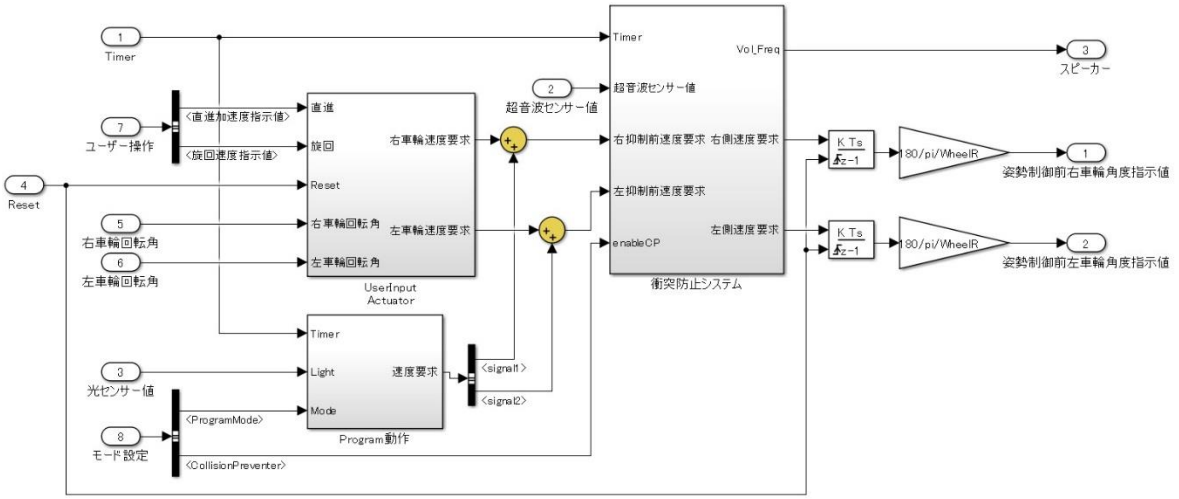
図 A-6-8 に示した姿勢制御部の最終段にある左右両輪の「Motor Actuator」ブロックの構成を図 A-6-14 に示す。センサー部の説明で述べたタッチセンサーによって EV3 が動作終了状態（図の「Reset」が「1」の状態）であれば、モーターへのパワー出力は「0」となって動力は与えられない。それ以外の場合は、左右のモーターの特性差に応じて、伝える動力に乗数をかけた出力値を-100～100 の範囲に制限して出力する。



付録 図 A-6-14 「MotorActuator」ブロックの構成

A-6.3 走行制御部

「コントローラー」部（図 A-6-2）を構成する「走行制御」ブロックは、図 A-6-15 に示すように、ユーザーによる操作入力を制限する「User Input Actuator」部と「Program 動作」部、そしてこれら 2 つのブロックから出力される左右両輪の速度要求値を超音波センサーの値を用いて補正する「衝突防止システム」から成っている。補正後の速度要求値は最終段において積分と単位変換が行われ、左右両輪に対する回転角度指示値（あるいは要求値）として出力される。以下、各ブロックの概要を述べる。



付録 図 A-6-15 「走行制御」ブロックの構成

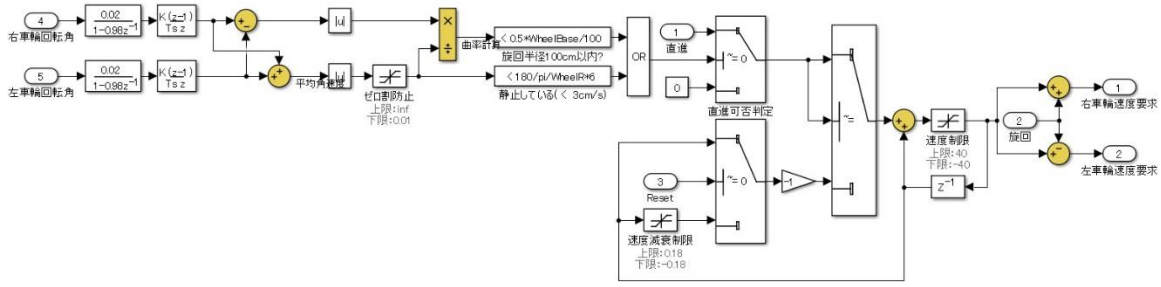
（1） 操作入力制限部（User Input Actuator）

本ブロックの機能もルール制御部と同様にハザード分析より導かれた転倒防止対策として追加されたものであり、図 A-6-16 にその構成を示す。ここでは次のような 2 つの機能を実現している。

即ち、その 1 つは左右両輪の回転角からロボットが旋回中か否かを判定し、旋回中であればユーザーによる前進・後退の指示を無視するというものである。今、右輪と左輪の前進速度をそれぞれv_R、v_Lとすると、ロボットの前進速度はこの 2 つの平均値、旋回半径rは

$$r = \frac{W}{2} \cdot \frac{v_R + v_L}{v_R - v_L}$$

で求められる。ここで Wは車軸長（14cm）である。図 A-6-16 では、両輪の回転角の信号をフィルターにより平滑化した後に微分して回転角速度に変換し、両者の和と差の比と車軸長（図中では「WheelBase」）を用いて計算した旋回半径が 100cm 以上であれば旋回中ではないとして、ブロックに入力されたユーザーからの前進加速度要求信号をそのまま「直進可否判定」スイッチより出力する。また、ロボットの前進速度が 3cm/s 以下の場合にも、ほぼ停止状態にあるとみなしてユーザーの要求信号をそのまま出力する。その後、「直進可否判定」スイッチの出力である加速度要求信号は積分されて速度要求に変換され、且つその上下限を±40cm/s に制限され、最後に旋回速度要求が加・減算された上で、左右両輪の速度要求信号として出力される。

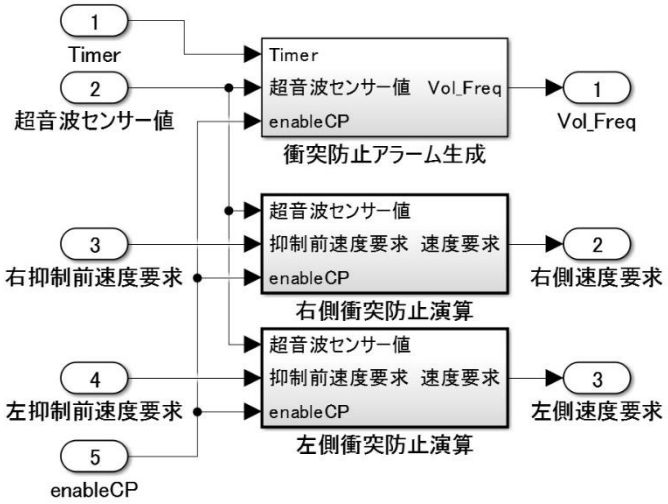


付録 図 A-6-16 「User Input Actuator」ブロックの構成

（2） 衝突防止システム

本ブロックは超音波センサーの信号を基に左右両輪の速度要求を制御することでロボットの障害物への衝突を防止することを目的としており、図 A-6-17 のように構成されている。

先ず、両輪の各々に対して用意されている「衝突防止演算」ブロックの構成を図 A-6-18 に示し、その処理内容を以下に説明する。

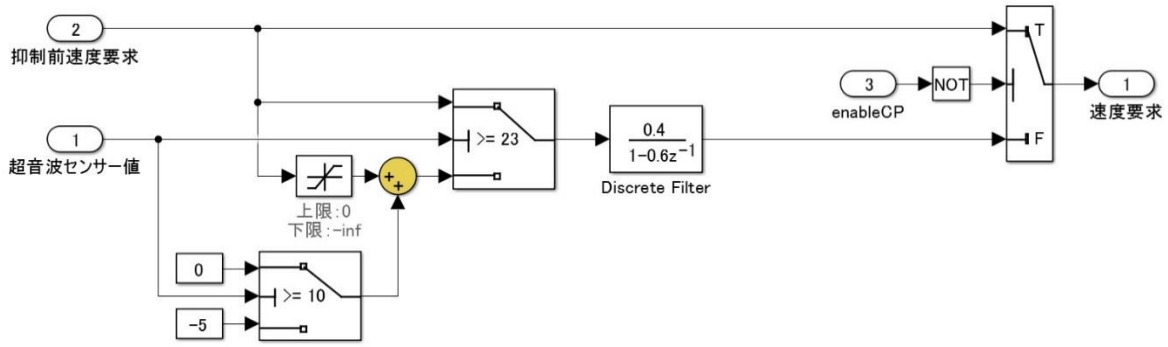


付録 図 A-6-17 「衝突防止システム」ブロックの構成

ユーザーが「衝突防止」機能を有効としていない（図の「enableCP」が「0」の）場合、入力された速度要求はそのまま出力される。衝突防止機能が有効な場合は、超音波センサーの出力する障害物までの距離によって処理内容が変わる。即ち、距離が 23cm 以上の場合、速度要求は一次遅れフィルターを通してそのまま出力される。距離が 23cm 以下の場合には、それが 10cm より遠ければ、速度要求を「0」とし、一次遅れフィルターを通して出力する。一次遅れフィルターを通すことでこのような急激な変化を緩やかなものとすることができる。また、このときユーザーが後進の指示を出した場合には、一次遅れフィルターを通した後にそれがそのまま出力される。そして、距離が 10cm 以下の場合には、入力された速度要求が正、即ち前進方向であれば「-5cm/s」の速度要求を一次遅れフィルター通過後に出力する。一方、入力速度要求が負、即ち後退方向であれば「-5cm/s」を加算した後の速度要求を一次遅れフィルターを通して出力する。

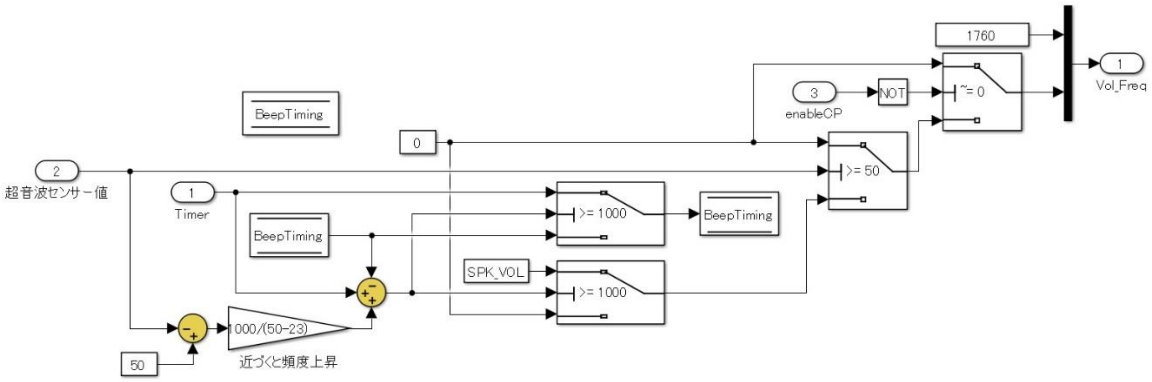
このように構成することにより、衝突防止機能が有効な際に、前方に障害物を検知していない、または十分に遠い場合は通常の走行を行い、障害物が近付いてきた際は前進を止め、さらに近付いてきた際は障害物にぶつからないよう、ユーザー操作が無くても後退する動作を行う。

なお、本ブロックの機能は昨年度も設けられていたが、プログラミングには StateFlow を用いていた。今年度は実装プログラムを簡単化し、少しでも処理時間を短縮することを意図して図 A-5-18 のように Simulink のみを用いて構成した。



付録 図 A-6-18 「衝突防止演算」ブロックの構成

次に、「衝突防止アラーム生成」ブロックの構成を図 A-6-19 に示し、その動作を説明する。



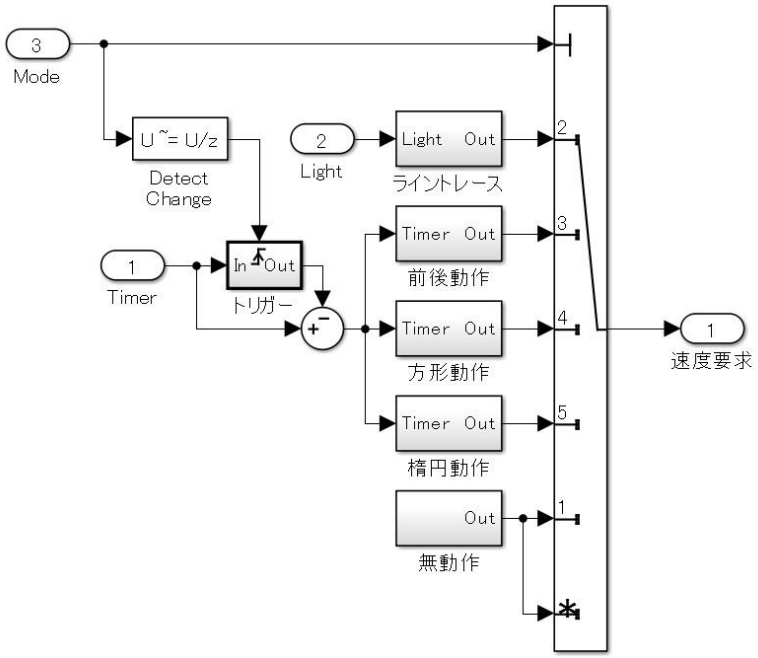
付録 図 A-6-19 「衝突防止アラーム生成」ブロックの構成

「衝突防止アラーム生成」ブロックでは、超音波センサー値を入力とし、音の周波数および音量を出力とする。音の周波数は 1760Hz で固定である。音量を変更することで、スピーカーからの音の発生の有無を制御している。衝突防止が無効（enableCP=0）の場合は音量を 0 としている。衝突防止が有効の場合､超音波センサー値から得られる物体とロボットとの距離に応じて音の出力頻度を変更する。50cm 以上距離がある場合は音量を 0 とする。50cm より近付いた場合、衝突防止を行う予備音として、距離が近付くに応じて 1000ms～0ms（連続）の頻度で音量を SPK_VOL（0～100 のパーセント指定。実験環境では 20）として出力する。衝突防止が働く 23cm になると音は連続で出力される。従って、50cm から 23cm に物体とロボットが近付くに従い、1cm 当たりおおよそ 37ms ずつ鳴動間隔は狭まることとなる。

（3） 「プログラム動作」部

「走行制御」部（図 A-6-15）を構成する「Program 動作」ブロックは、ユーザーコマンド入力部から設定された動作モードに対応して、ユーザーが運転操作をすることなく、ロボットが予め定められた軌道を描くように左右両輪の速度要求信号を生成出力する機能を持つものである。昨年度のサンプルシステムも有していたライントレース機能もその一つに含まれる。今年度新たに追加されたのが、楕円形あるいは長方形の軌道を描く動作モードと、直線上を前進・後退する前後動作モードの３つである。

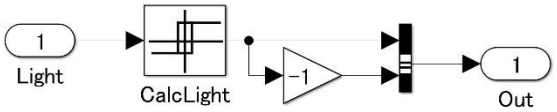
本ブロックの構成を図 A-6-20 に示し、含まれる各ブロックについて順不同で説明する。



付録 図 A-6-20 「Program 動作」ブロックの構成

① ライントレース

図 A-6-21 に示す「CalcLight」ブロックにおいて、カラーセンサーの出力値がしきい値「45」以上の際は「0.4451」を、そうでない場合は「-0.4451」を出力する。この出力値はそのまま右輪の、また符号を反転させた値が左輪の、それぞれ速度要求値として出力される。このしきい値「45」は事前に黒線とそうでない場所のカラーセンサー出力値を取得しておき、十分に振り分けが可能なしきい値とする。これにより、ライントレース時に黒線を検知した場合とそうでない場合で旋回方向を反転し、黒線の縁を走行できるようになる。



付録 図 A-6-21 「ライントレース」ブロックの構成

② 楕円動作

楕円動作ブロックでは、ロボットが周期 T (s)で長径 a、短径 b の楕円軌跡を描くように走行させるために必要な左右両輪の速度要求信号を生成する。今、ロボットを反時計回りに走行させるものとし、車軸長を W とすると、右輪の走行速度v_R、左輪の走行速度v_Lを次式で与えれば良い。

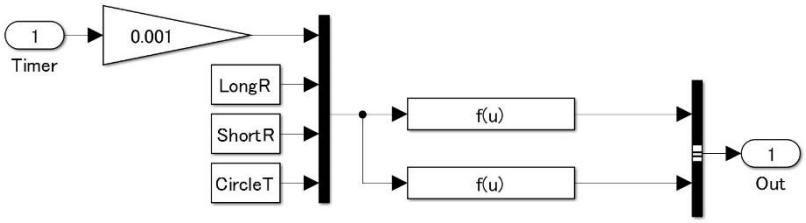
$$v_R = (r + W/2) \cdot \omega, \qquad v_L = (r - W/2) \cdot \omega$$

ここでrは曲率半径、ωは角速度であり、$\alpha = 2\pi/T$とすると、それぞれ次式で表わされる。

$$r = \{(a \cdot \sin \alpha t)^2 + (b \cdot \cos \alpha t)^2\}^{3/2} / (a \cdot b)$$

$$\omega = a \cdot b \cdot \alpha / \{(a \cdot \sin \alpha t)^2 + (b \cdot \cos \alpha t)^2\}$$

ブロックの構成は図 A-6-22 のようになっており、a、b（図中の「LongR」、「ShortR」）、T（図中の「CircleT」）、及び時間 t の値を用いて、関数ブロック内で左右両輪の速度を計算する。



付録 図 A-6-22　「楕円動作」ブロックの構成

なお、上記 a、b、T の値は後出の UI パネル（図 A-6-30）より設定する。

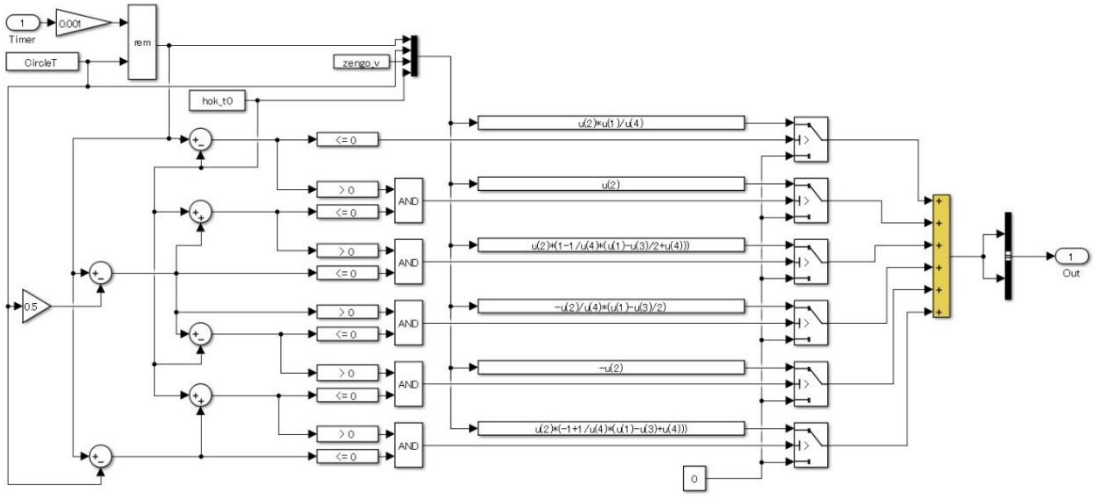
③　方形動作

方形動作ブロックは、ロボットが周期 T（s）で長さ S1、S2 を２辺とする長方形の軌跡を描くように走行させるために必要な左右両輪の速度要求信号を生成する。安定に走行させるため、コーナー手前で減速し、90 度回転する際は片輪を停止し、回転後に加速して直線部を一定速度で走行する。このために必要な減速（及び加速）させる時間 t0、減速率（減速後／減速前の速度比）を UI パネルより設定しておく。

本ブロックの構成は非常に複雑であるためにここでは図を省略し、より簡単な前後動作の場合の図をこの後で示す。なお、ロボットの左右モーター特性の差や、車輪と走行面との摩擦等の関係から、長時間にわたって正確に方形動作をさせることは難しいことを付記しておく。

④　前後動作

方形動作のために与えた長さ S1 の区間を、方向転換することなく、周期 T で単純に往復するための速度要求信号を生成するブロックである。図 A-6-23 にその構成を示す。



付録 図 A-6-23　「前後動作」ブロックの構成

方形動作のために与える減速時間$t0$をここでも用い、走行開始後の時間$t0$で停止状態から速度$v1$まで加速する。そして距離$S1$の点に到達する前$t0$間に速度$v1$から減速して停止し、今度は後ろ向きのまま加速・走行・減速して元の地点に戻るという動作を周期Tで繰り返す。ここで、速度$v1$は$v1 = S1/(T/2 - t0)$ で与えられる。

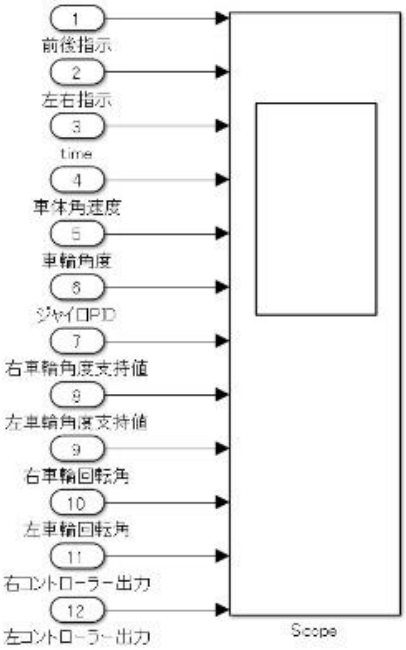
A-6.4　モーター

図 A-6-1 に示したモデルの右下に位置する左右のモーターは、コントローラー部で計算された要求値をロボットのモーターに伝えるものである。値に応じて、回転方向及び回転パワーが決定される。

A-6.5　情報出力

（1）　PC への出力

図 A-6-1 に示したモデルにおいて、図 A-6-24 に示すデータが PC 側の Scope 画面に出力表示される。これらは、①ユーザー操作信号、②コントローラー部のうちの姿勢制御部への入力信号、及び③姿勢制御部で計算された LCD 表示信号の一部となっている。そして、Simulink の実行が停止された場合、後述する事後診断のために一定時間分のデータが時間付き構造体データとして MATLAB のワークスペースに保存される。事後診断に用いるのは②の各信号と、③の一部である。

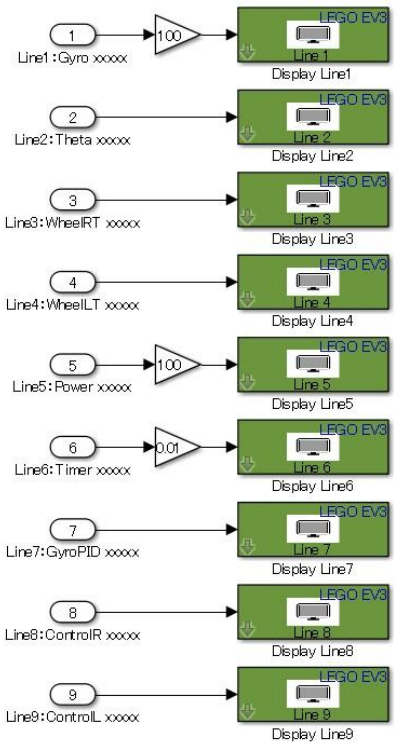


付録 図 A-6-24　PC への出力データ

なお、事後診断プログラムを起動する前にロボット側 Simulink プログラムをアンロードしてはならない。アンロードした場合にはログデータも消去される。

（2）　LCD 表示

コントローラー部の中の姿勢制御部の出力のうち、図 A-6-25 に示す信号がロボットの LCD に表示される。



付録 図 A-6-25　LCD 表示モデル

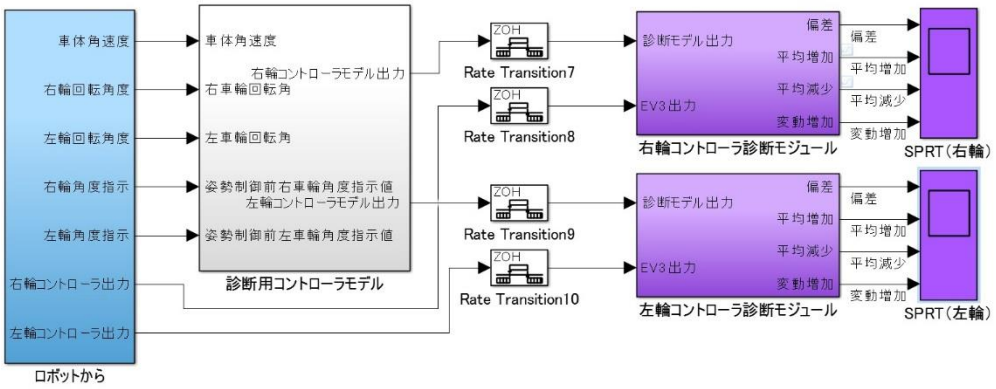
（3）　スピーカー出力

図 A-6-1 の右下に示したスピーカーは、コントローラー部中の姿勢制御部においてルール制御が実行された場合、及び走行制御部の衝突防止システムで衝突回避動作が実行された場合にそれぞれ異なる高さと長さの警報音を発するものである。

A-6.6　診断プログラム

ここで示す診断プログラムは、ロボットが何らかの原因で転倒したような場合に、ロボット側 Simulink プログラムの実行を停止して保存されたログデータを基に転倒の原因を探る一つの手段として用いることを目的としたものである。即ち、ロボット実機のコントローラー入力信号の測定値を正常なコントローラーモデルに入力して得られる出力を、ロボット実機のコントローラー出力信号と比較することで、実機のコントローラーの特性に何らかの変化が生じていたか否かを診断するプログラムとなっている。

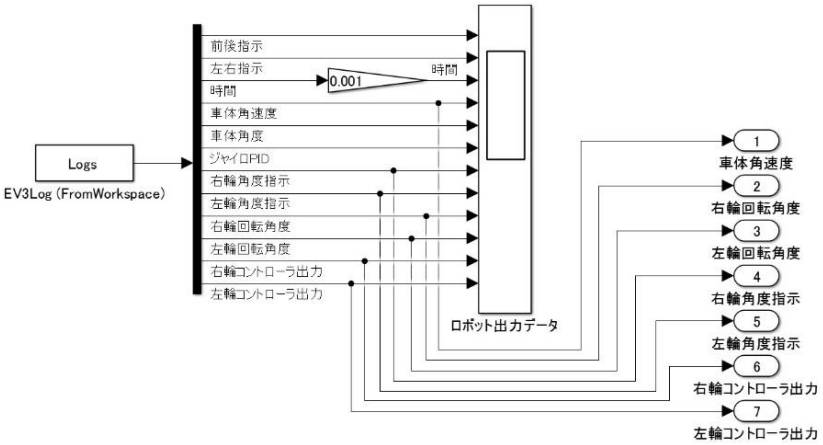
Simulink を用いて作成した、PC 側で事後診断を行うプログラムの全体構成を図 A-6-26 に示す。プログラムは、ロボットから出力保存されたログデータを入力する部分、診断対象であるロボットの姿勢制御系の正常時の特性を持つ「診断用コントローラーモデル」、その出力値と実測値であるログデータとの比較から診断を行う「診断モジュール部」、及びその結果を表示する Scope から構成される。以下、その内容を示す。



付録 図 A-6-26　事後診断プログラムの全体構成

(1)　ログデータ入力部

図 A-6-26 で「ロボットから」と表示されたブロックの構成を図 A-6-27 に示す。

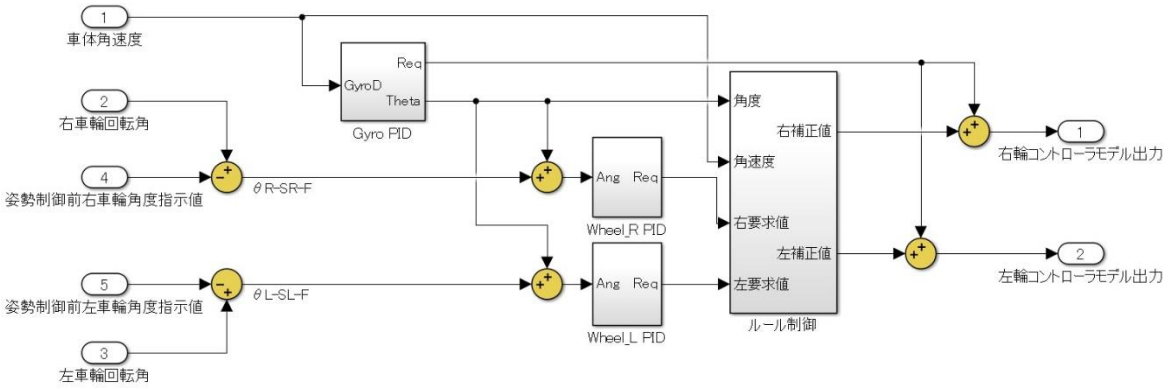


付録 図 A-6-27　ロボットからの出力処理モデル

プログラムの実行を開始すると、先ずワークスペースに保存された時間付き構造体データ「EV3Log」を「Logs」と言う名前で読み込み、「ロボット出力データ」として Scope に表示するとともに、このうちの図 A-6-27 右端に示す 7 信号を本ブロックよりの出力信号とする。構造体データ「EV3Log」がワークスペースに無かった場合は、過去に作成されたログデータ（「LogData.mat」に保存）を読み込んで使用する。

(2)　診断用コントローラーモデル

診断用コントローラーモデルはロボット実機の姿勢制御部（図 A-6-27）に含まれる「Gyro PID」と左右の「Wheel PID」、及び「ルール制御」部から成り、図 A-6-28 のように構成される。各ブロックの構成は実機のコントローラーと全く同一であり、従ってその動作も同様である。



付録 図 A-6-28　診断用コントローラーモデル

(3)　コントローラー診断モジュール

図 A-6-26 に示された左右両輪のそれぞれに対する診断モジュールの内容は全く同一であり、図 A-6-29 のように構成される。本モジュールでは、EV3 のコントローラー出力と前記診断用コントローラーモデルの出力との偏差に対して、逐次確立比検定（SPRT）による異常判定を行う。SPRT については報告書本文 6.3 節に記載があるが、二輪倒立ロボットの場合には姿勢制御が不安定になると前記偏差の平均値は変わらずに変動のみが大きくなる可能性もあることから、以下、若干補足しておく。

先ず、正常状態の仮説 H_0（例えば、平均 a_0，分散 σ^2 の正規分布に従う）と、本文の例のように平均値のみが変化するとした異常状態の仮説 H_1（例えば、平均 a_1，分散 σ^2 の正規分布に従う）とのどちらが確からしいかを判定するための対数尤度比 $\lambda(t)$ は、時刻 t での残差 $\varepsilon(t)$ を得るごとに、次式によって逐次的に計算される。

$$\lambda(t) = \log\frac{P(\varepsilon(t)|H_1)}{P(\varepsilon(t)|H_0)} = \lambda(t-1) + \frac{(a_1 - a_0)}{\sigma_0^2}\left\{\varepsilon(t) - \frac{1}{2}(a_1 + a_0)\right\}$$

これに対して、正常仮説 H_0 と、平均値が変わらずに分散のみが変化するとした仮説 H_2（平均 a_0，分散 σ_1^2 の正規分布に従う）との対数尤度比は次式のように計算される。

$$\lambda(t) = \lambda(t-1) + \log\left(\frac{\sigma_0^2}{\sigma_1^2}\right) + \frac{\{\varepsilon(t) - a_0\}^2}{2}\left(\frac{1}{\sigma_0^2} - \frac{1}{\sigma_1^2}\right)$$

異常判定は、誤検出率(False alarm rate, α)と非検出率(Miss alarm rate, β)を用いて、下記のように行う。

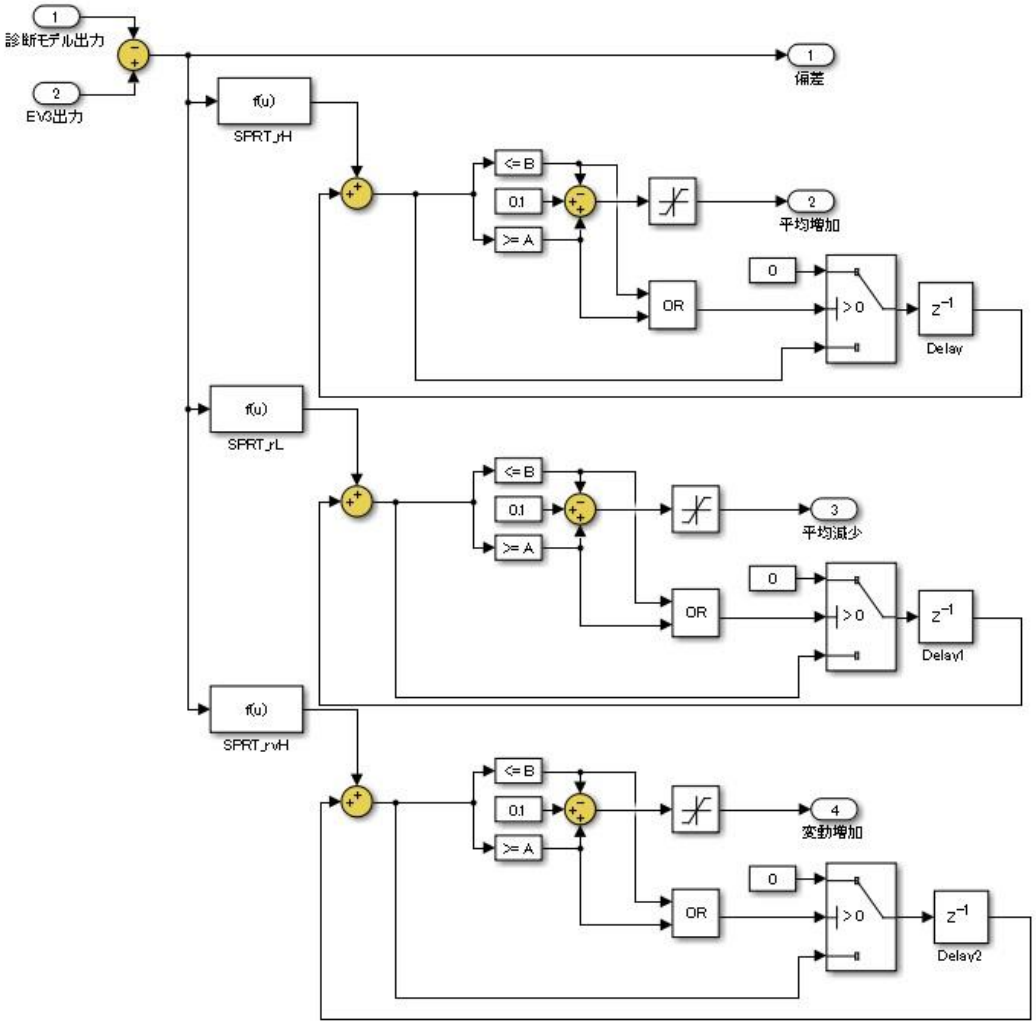
$\lambda(t) \le B$　　：　H_0 を採択（正常）
$\lambda(t) \ge A$　　：　H_1 を採択（異常）
$B < \lambda(t) < A$　：　監視継続（判定保留）
ただし、$A = \log((1-\beta)/\alpha)$，$B = \log(\beta/(1-\alpha))$ である。

本モジュールでは姿勢制御用コントローラーの最終的な出力を診断に用いており、実測値と図 A-6-28 のモデルによる計算値との偏差に対して、平均値の増加、平均値の減少、及び分散の増加の３通りの異常変化が生じているかどうかを判定する。これらの三つの判定で異なるのは対数尤度比の計算式のみであり、図 A-6-29 に示された３つの関数ブロック、「SPRT_rH」、「SPRT_rL」、「SPRT_rvH」でこの計算を行い、異常判定部は全て同じ構成となっている。

誤検出率と非検出率はそれぞれ 1%と 0.1%を与えており、これから $A = 4.60$、$B = -6.90$ となっている。また正常時の平均値は $a_0 = 0$，分散は $\sigma_0^2 = 0.1$ とし、平均増加、平均減少の仮説として $\pm\sigma_0$ の変化を、分散増加の仮説として分散が $2\sigma_0^2$ となることを想定した。

図の構成では診断結果として、正常な場合「0」、判断保留の場合「0.1」、異常な場合「1」として出

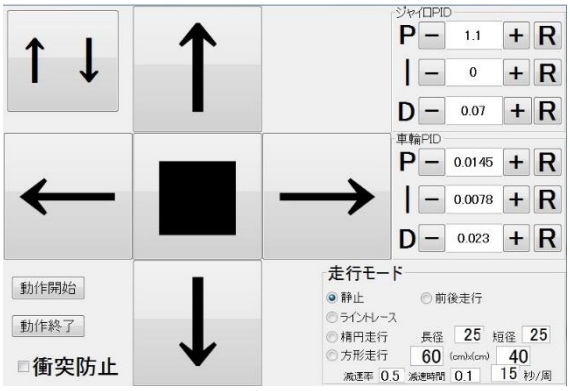
力するようになっており、図 A-6-26 に示したようにこの結果は左右両輪別に Scope にトレンド表示される。



付録 図 A-6-29　診断モジュールの構成

A-6.7　UI パネル

UI パネルを表示するパログラムは、ロボット側 Simulink プログラムのロード時に同時にロードされる。図 A-6-30 に示す UI パネルではロボットへの操作指示、PID 制御定数のリアルタイム変更、及び種々の動作モードの変更が行える。



付録 図 A-6-30　PC 側操作 UI パネル

（1） ロボットへの動作指示

UI パネルに十字状に配置された上下左右の矢印と中央の■ボタンを押すことで、ロボットにそれぞれ前後の移動、左右に旋回、その場に停止(旋回も止める)の指示を出すことができる。左上で上下の矢印が共に描かれているボタンを押した場合には、2 秒間前進し、4 秒間後退し、最後に 2 秒間前進して停止する。

（2） PID 制御定数のリアルタイム変更

UI パネル右上にある車体 PID、車輪 PID のパラメータを変更することで、リアルタイムにロボットが用いる PID パラメータを変更可能である。初期値は EV3 が比較的安定して動作する値を設定している。左右のボタンで初期値の 10%の増減が行えるほか、数値入力によっても直接値を変更可能である。R ボタンは値を初期値に戻すものである。

（3） 動作モード変更

UI パネル右下にあるラジオボタンを押して切り替えることにより、プログラム走行の種類を選択することができる。楕円走行を選択する場合は、事前に当該ラジオボタン右側にあるテキストボックスで長径と短径、及び一番右下の周期（秒/周）を与えておく。同様に、方形走行を選ぶ場合には、右側で２辺の長さを与えておく。さらに、コーナーを回転する際の減速率と減速時間も与える必要がある。これらのパラメータについては A-6.3 に記した。周期は各走行動作で共用である。前後走行の場合には、方形走行で与える第１辺の長さの直線上を往復する。ラジオボタンの初期設定は「停止」である。

衝突防止機能を有効にする場合は、UI パネルの左下にある値はチェックボックスにチェックを入れる。この初期設定は衝突防止無効となっている。このチェックボックスの上方に置かれた「動作開始」「動作終了」の２つのボタンは、ロボット側 Simulink プログラムの実行を開始、停止するものである。