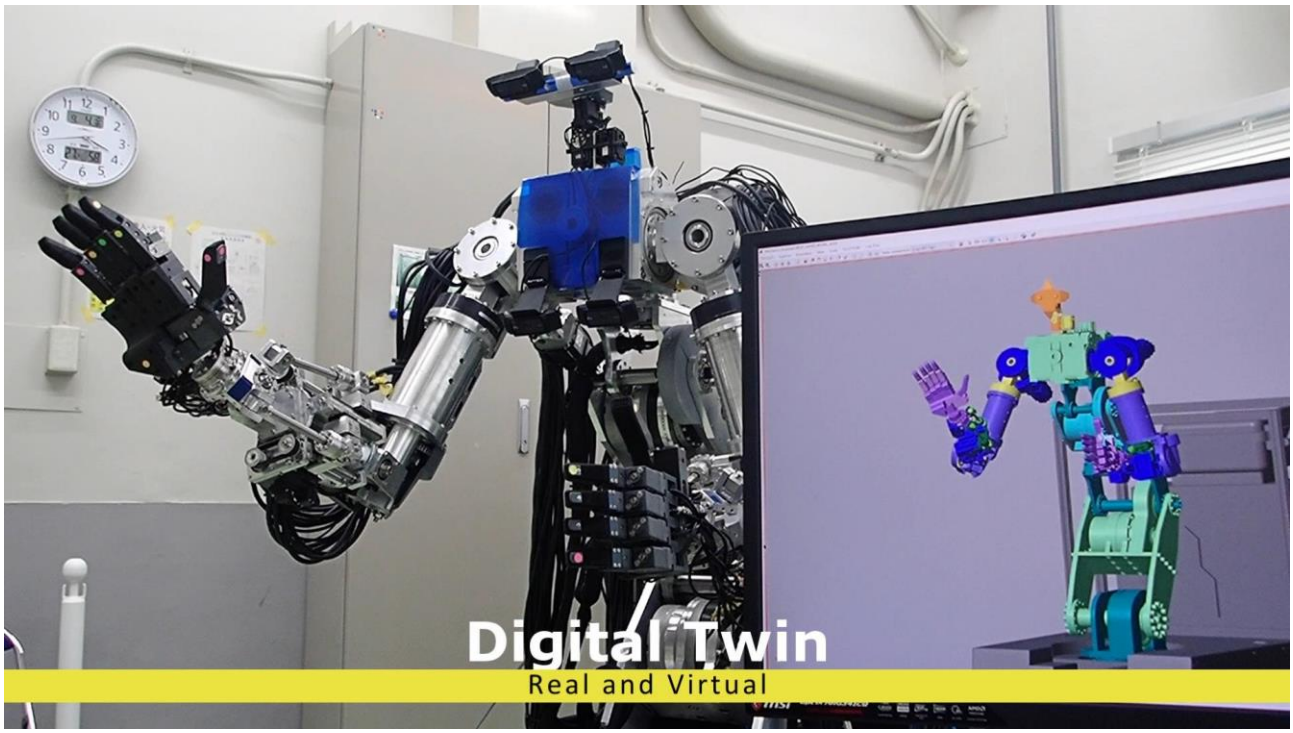
モデルベース開発手法による リアルタイムシミュレーション

神田 吉孝



1. まえがき

モデルベース開発（MBD：Model Based Development）とは，コンピューター上で再現した“モデル”を用いることで，設計時からシミュレーションによる検証を繰り返し実施する開発手法である。その結果，従来の仕様書ベースでの開発に比べ，手戻りを排除し，効率化・短時間化を図ることができる。自動車業界では既に標準的な開発手法のひとつであり，航空宇宙分野でも成果を上げている。

当社の防衛・宇宙事業へのモデルベース開発の適用を図るべく，人型遠隔操作ロボット“DiaroiD”を題材に，2019年度から開発に取り組んだ。

開発を通じ，MILS(Model-in-the-Loop Simulation)，自動コード生成などの成果を得るとともに，モデルベース開発の長所・短所の知見を蓄積することができた。

その成果を活かして，リアルタイムシミュレーションによる“DiaroiD 操縦支援システム”を製作した。

本稿では，これら開発の経緯，成果及び事業展開への展望について述べる。

2. 開発概要

防衛・宇宙事業でもモデルベース開発の適用を図るべく開発をスタートした。企画当初は，技術アピールのためのデモンストレーション素材の作成が目標であったが，その後，自動コード生成環境及びリアルタイムシミュレーション環境の構築を通して，技術要素確立，さらには，事業展開できる技術力の獲得に目標を改めた。

ターゲットは，デモンストレーション素材として人々の興味を引くことができ，機械的動作が分かりやすい人型遠隔操作ロボット“DiaroiD”とした。

DiaroiD は，胴体，腕，手指，首の各部が操縦者からの遠隔操作によって動作するロボットである。

開発ツールは，MATLAB/Simulink に加え，ロボットのプラントモデル製作に Simscape Multibody を採用した。

2.1 DiaroiD

DiaroiD とは,図1に示す人型遠隔操作ロボットである。1 号機を三菱電機先端技術総合研究所，2 号機を三菱電機通信機製作所が製作した。

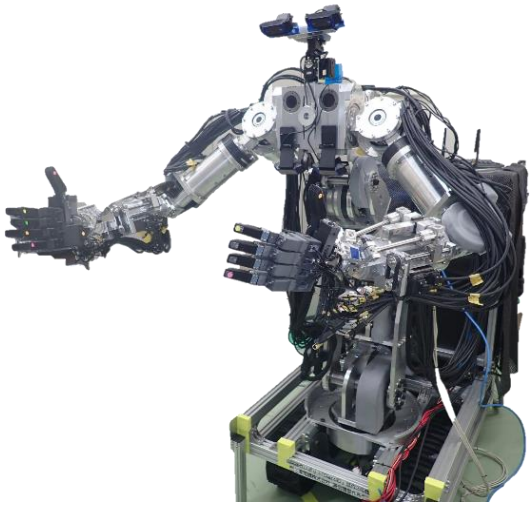


図 1. 人型遠隔操作ロボット“DiaroiD”2 号機

胴体6，腕7×2，手指6×2，首3，合計35自由度を一人で操縦できる。

操作者の腕に沿わせて装着する入力装置により腕部と手指部を,音声入力により胴体部と首部を操縦する。また,あらかじめプログラミングした定型動作を実行することもできる。

脚部はクローラ型の移動台車になっており,リモコン操作により移動することが可能である。

ソフトウエアを当社が，メカ設計，製造を三菱電機エンジニアリングが,全体構想設計と電気設計を三菱電機通信機製作所が担当した。

2.2 開発ツール

モデルベース開発環境のデファクトスタンダードとなっている MATLAB シリーズの図2の枠線で囲った製品を採用した。

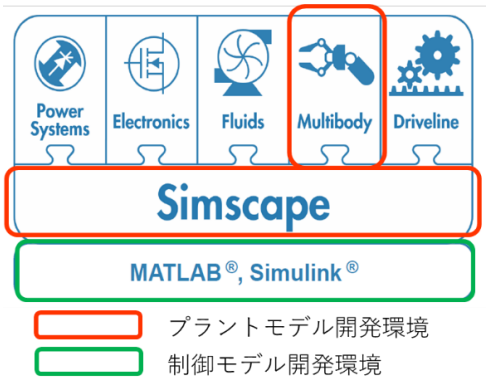


図 2. 開発環境

2.2.1 MATLAB

関数やアルゴリズム開発,行列計算など様々なことを可能にしている数値解析ソフトウエアである。また，その中で使うプログラミング言語の名称でもある。

2.2.2 Simulink

ブロック線図を描いて時間軸でのシミュレーションを行う環境であり,オプションによりブロック線図を C ソースコードへ変換することができる。

2.2.3 Simscape Multibody

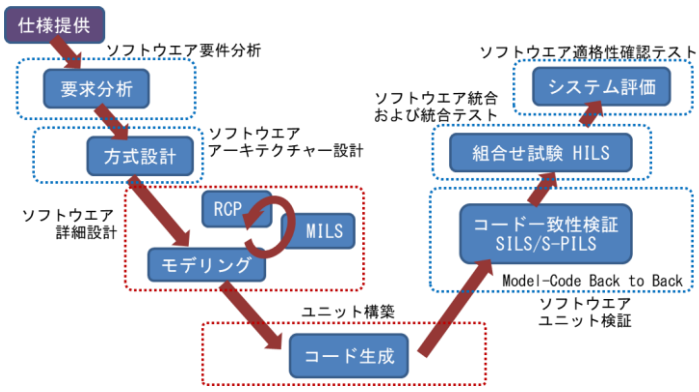
3D 機械システムで使用する剛体シミュレーション環境であり，ボディ，ジョイント，拘束，力学的作用及びセンサーなどを表すブロックを用いて，3D 剛体システムをモデリングし，シミュレーション結果を 3D アニメーションにて可視化することができる。

3. モデルベース開発

図3にモデルベース開発の流れを示す。

特徴は，設計段階から製作対象をモデリングし，そのモデルを用いて MILS により検証を行うこと，及び，検証後のモデルから実装コードを作成することである。

以降，今回の開発内容について述べる。



※RCP：Rapid Control Prototyping
MILS：Model-in-the-Loop Simulation
SILS：Software-in-the-Loop Simulation
S-PILS：Simulated Processor-in-the-Loop Simulation
HILS：Hardware-in-the-Loop Simulation

図 3. モデルベース開発の流れ

3.1 モデリング

DiaroiD のモデルは，以下の 3 つに大別できた。

(1) 外部入力や駆動スケジュールにより，駆動指令入力を行う“コマンド入力モデル”

(2) 入力された指令で機械制御を行う“制御モデル”

(3) ロボット本体をシミュレートする“プラントモデル”

3.1.1 コマンド入力モデル

コマンド入力モデルは，ロボットの駆動指令を生成又は，外部から入力する。

あらかじめ設計した時系列の駆動スケジュールによる“コマンド再生”と，外部から駆動指令を逐次入力する“UDP(注1)通信”の 2 種類のコマンド入力モデルを作成した。

“コマンド再生”は，設計した動作を何回でも再現できるため，モデル更新時の比較検証に向いている。Simulink 標準ライブラリーの“Signal Builder”ブロックを用いて作成した。

“UDP 通信”は，DiaroiD の操作入力装置からの UDP 通信コマンドの入力を可能にしたものである。操作による挙動の確認や，後述するリアルタイムシミュレーションに使用する。UDP 通信機能は Simulink 標準ライブラリーには用意されておらず“C Caller”ブロックを用いて自作した。

3.1.2 制御モデル

制御モデルは，入力したコマンドを座標変換し，位置指令を生成する。

本節では，座標変換として

(1) 胴体部のキネマティクス(注2)

(2) 手首部のキネマティクス

位置指令生成として

(3) サーボ制御

の特徴を述べる。

(1) 胴体部のキネマティクス

DiaroiD 2 号機の胴体部分は，産業ロボットによく見られる 6 軸垂直多関節ロボットの構造を採用している。図 4 に胴体部構造を示す。

6 軸垂直多関節キネマティクスは，式 1～式 4 を用いて数値演算する。

各関節(アクチュエーター)の角度

$[\theta AZ1 \quad \theta EL1 \quad \theta EL2 \quad \theta AZ2 \quad \theta EL3 \quad \theta AZ3]$

から姿勢

$[x \quad y \quad z \quad \theta x \quad \theta y \quad \theta z]$

を求めることをキネマティクス，姿勢から各関節の角度を求めることを逆キネマティクスという。姿勢の要素のうち位置ベクトル$[x \quad y \quad z]$は式 1 の Affine 行列による変換結果となる。ここで，$Trans(Px)$は関節から次の関節への平行移動を指す。

姿勢の要素のうち方向ベクトル$[\theta x \quad \theta y \quad \theta z]$は式 2 と式 3 の行列が同じ方向を示すことを利用して，式 4 で求められる。Q_{lm}は，3×3 行列$[Q]$の l 行 m 列目の要素を指す。

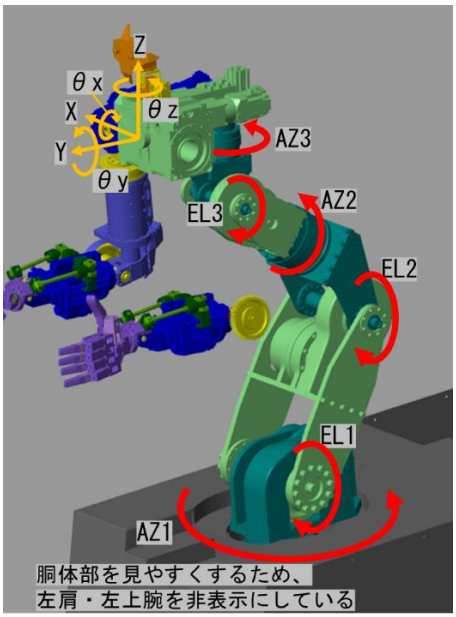


図 4. DiaroiD の胴体部構造

$$[x \quad y \quad z \quad 1] = Rot(\theta AZ1,z)\cdot Trans(P1)\cdot Rot(\theta EL1,x)\cdot Trans(P2)\cdot Rot(\theta EL2,x)\cdot Trans(P3)\cdot Rot(\theta AZ2,z)\cdot Trans(P4)\cdot Rot(\theta EL3,x)\cdot Trans(P5)\cdot Rot(\theta AZ3,z)\cdot Trans(P6)\cdot [0 \quad 0 \quad 0 \quad 1]^t \quad \cdots式1$$

$$[Q] = Rot(\theta AZ1,z)\cdot Rot(\theta EL1,x)\cdot Rot(\theta EL2,x)\cdot Rot(\theta AZ2,z)\cdot Rot(\theta EL3,x)\cdot Rot(\theta AZ3,z) \quad \cdots式2$$

$$[R] = Rot(\theta x,x)\cdot Rot(\theta y,y)\cdot Rot(\theta z,z) \quad \cdots式3$$

$$\theta x = tan^{-1}\frac{-Q_{23}}{Q_{33}},\ \theta y = sin^{-1}Q_{21},\ \theta z = tan^{-1}\frac{-Q_{12}}{Q_{11}} \quad \cdots式4$$

(注1) User Datagram Protocol：インターネットなどのネットワークで，IP(Internet Protocol)の一段階上位層のプロトコル(通信規約)として標準的に使われるもののひとつ。

(注2) Kinematics：運動学。運動の原因，つまり力を考慮せずに，幾何学的な運動のみに注目した力学の一体系。本稿では，各関節の変位量から先端の位置姿勢を求める問題として使用している。

$$[S]=\begin{pmatrix} S_{ax} & S_{bx} & 0 \\ S_{ay} & S_{by} & 0 \\ S_{az} & S_{bz} & 0 \end{pmatrix},\ [T]=\begin{pmatrix} T_{ax} & T_{bx} & 0 \\ T_{ay} & T_{by} & 0 \\ T_{az} & T_{bz} & 0 \end{pmatrix},\ \widehat{T_a}=\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},\ \widehat{T_b}=\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \cdots式5$$

$$[S']=Rot(\theta v,y)\cdot Rot(\theta u,x)\cdot[S] \quad \cdots式6$$

$$\left|\overrightarrow{S'_a}-\left(\overrightarrow{T_a}+l_a\widehat{T_a}\right)\right|=\left|\overrightarrow{S_a}-\overrightarrow{T_a}\right|,\ \left|\overrightarrow{S'_b}-\left(\overrightarrow{T_b}+l_b\widehat{T_b}\right)\right|=\left|\overrightarrow{S_b}-\overrightarrow{T_b}\right| \quad \cdots式7$$

式1，式4は角度

$[\theta AZ1 \quad \theta EL1 \quad \theta EL2 \quad \theta AZ2 \quad \theta EL3 \quad \theta AZ3]$

について直接的に解くことができないため，逆キネマティクスを求める場合は，ニュートン＝ラフソン法による収束演算を行う必要がある。

(2) 手首部のキネマティクス

DiaroiD 2号機の腕部のうち，手首部分の2自由度は，パラレルリンク構造を採用しており，2本の直動リンクにより手首のθuとθvの回転を実現している。図5に手首のパラレルリンク構造の概念図を示す。シリンダー駆動の位置$[l_a \quad l_b]$から姿勢$[\theta u \quad \theta v]$を求めることをキネマティクス，姿勢から各シリンダー駆動位置を求めることを逆キネマティクスという。

パラレルリンクキネマティクスは式5～式7を用いて数値演算する。

式5の$[S]$は$\theta u=\theta v=0$時の回転側の2つのユニバーサルジョイントの座標，$[T]$はシリンダー側のユニバーサルジョイントの座標，$\widehat{T_a},\widehat{T_b}$はシリンダー駆動方向の単位ベクトル，$\left|\overrightarrow{S_a}-\overrightarrow{T_a}\right|,\left|\overrightarrow{S_b}-\overrightarrow{T_b}\right|$は2本のリンクの長さである。

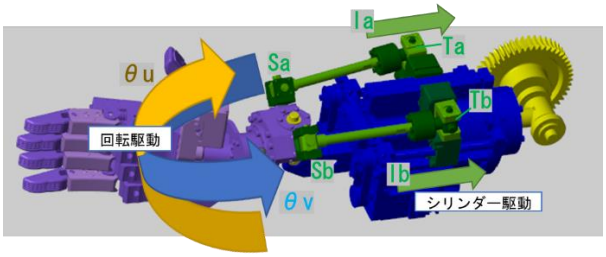


図 5. 手首パラレルリンク構造

リンクの長さは変わらないことから，式6と式7にて逆キネマティクスを演算できる。

パラレルリンクも垂直多関節と同様にキネマティクスは収束演算により回転角を求める。

(3) サーボ制御

DiaroiD のサーボ制御は，ソフトウエア内でフィードフォワード制御を行い，ポジションボード内でフィードバック制御を行っている。これに合わせて，制御モデルもフィードフォワード制御とした。コマンドに対し速度制限と駆動範囲制限を処置し，位置指令を生成するモデルを作成した。サーボ制御モデルを図6に示す。

3.1.3 プラントモデル

プラントモデルは，制御対象の振る舞いを再現する。

最も困難であり，最も力を入れた部分である。本開発では，メカ設計図であるCAD(注3)モデルを基に，モデリングした。モデリング手順は，

(1) CADモデルのリダクション
(2) CADモデルをSimscapeで読込み
(3) 部品の整列と相対位置の算出
(4) 部品の接続
(5) 接続点に回転機構を挿入
(6) 回転機構へ位置指令を接続

となる。加えて，実空間での現象を再現するため

(7) 衝突の実現

を行った。

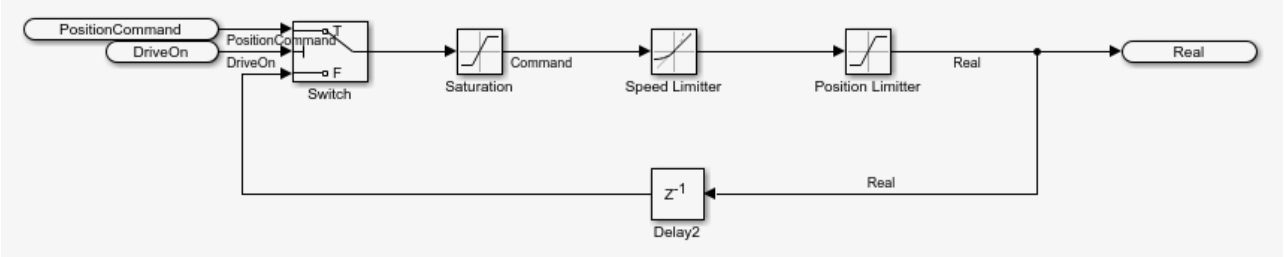


図 6. サーボ制御モデル

(注3) Computer-Aided Design：コンピューターを用いて設計，作図をすること，あるいはその設計支援ツールのこと。

図 7 に作成したプラントモデルを示す。

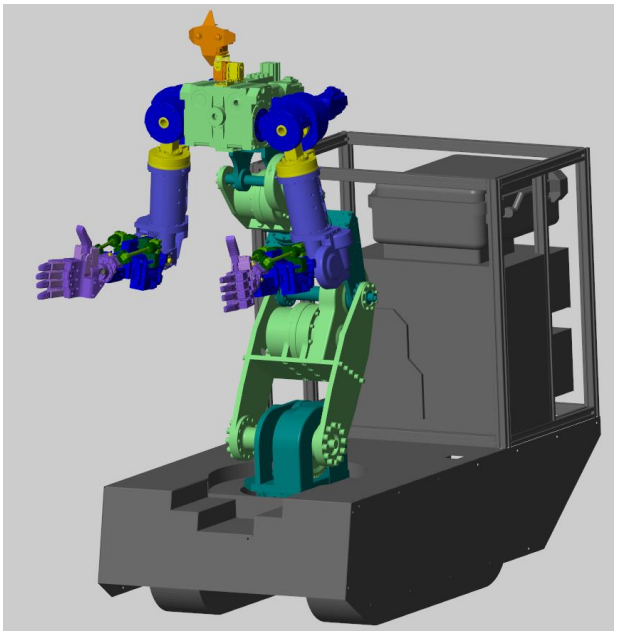


図 7. プラントモデル

(1) CAD モデルのリダクション

メカ設計の CAD モデルはボルト 1 点まで詳細に部品分割されているが, Simscape でこれらを 1 点 1 点変位シミュレーションすると膨大な演算量になり現実的ではない。

そこで, CAD モデルの各部品を一体と見なせる単位まで結合し部品点数を減らす処置が必要であった。これをリダクションという。当社にはこの技術がなかったため, リダクションは三菱電機通信機製作所に協力いただいた。

(2) CAD モデルを Simscape で読込み

CAD ソフト“Creo Parametric”のモデルを Simscape が読み込める STL 形式(注4)のデータに変換し, Simscape へ読み込んだ。形式変換にはプラグイン“Simscape Multibody Link”を使用した。

(3) 部品の整列と相対位置の算出

読み込んだモデルは, 全ての部品がグローバル座標上の絶対位置に浮いている状態であった。

これら部品同士を接続するために, 部品同士の相対的な位置関係を求める必要があった。

そこで, Simulink 上で機構順に部品を並べ, 部品間の相対位置を算出した。

相対位置の算出には Simscape の“Transform Sensor”というブロックを利用した。

(4) 部品の接続

“Rigid Transform”という座標変換ブロックに算出した相対位置を設定し, これを介して部品を接続した。

これにより, 一例として, ロボットの前腕の位置は上腕からの相対位置で決定されるようになった。

(5) 接続点に回転機構を挿入

“Rigid Transform”で接続しただけでは, 部品と部品が固定された状態である。

ここに“Joint”という回転機構ブロックを挿入した。

“Joint”の回転座標系と部品の座標系は異なるため, “Joint”の回転軸と動かしたい回転軸が一致するよう座標変換する必要があった。部品 A－座標回転－Joint－座標逆回転－部品 B といったように, 座標回転と座標逆回転を挟みながら“Joint”を挿入した。

これにより, ロボットの関節を動かせるようになった。

(6) 回転機構へ位置指令を接続

回転機構の指令入力に制御モデルからの位置指令を接続した。

これにより, 制御モデルからの指令で, プラントモデルを駆動できるようになった。

(7) 衝突の実現

部品は初期状態では, 互いをすり抜けることができてしまう。図 8 の左側は, 下した右腕が左腕をすり抜けて重なってしまった場面である。

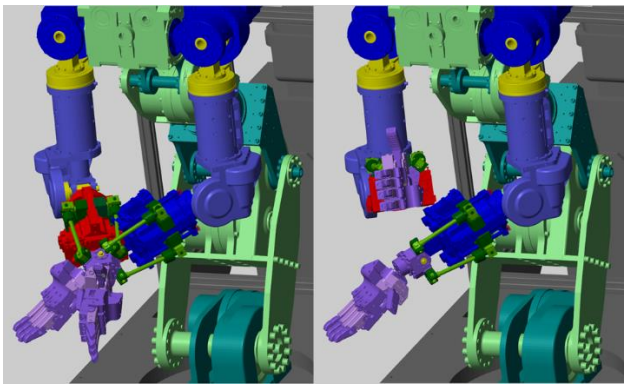


図 8. 衝突の有無

図 8 の右側のように, 下した右腕は左腕に衝突し, 停止又は, 跳ね返るようにする必要がある。

Simscape では, 部品間に垂直抗力という力学的作用を定義することで, 衝突という現象を実現する。

(注4) 三次元形状を表現するデータを保存するファイルフォーマットのひとつ。名称の由来は光造形法を意味する Stereolithography である。

垂直抗力の定義により，部品同士が衝突し，互いにすり抜けられないようにした。

Simscape では，垂直抗力や摩擦力といった力学的作用は，部品そのものに定義するのではなく，部品間の関係性として定義する。

3.2 MILS (Model-in-the-Loop Simulation)

でき上がったモデルを利用して次のような検証を行った。

(1) キネマティクス演算の設計検証
(2) 駆動順序による衝突の回避検証

詳細は割愛するが，この他にも，

(3) アクチュエーターの駆動範囲
(4) アクチュエーターの駆動速度

をシミュレーションで検証した。

3.2.1 キネマティクス演算の設計検証

(1) 不定解の回避

回転軸が一致し不定解となる（解の個数が無限になる）問題があった。

不定解となるケースは以下の 3 とおりであった。

ケース 1： AZ2 と AZ3 が不定(図 9 左)

$\theta EL3 = 0$ のとき

ケース 2： AZ1 と AZ2 が不定(図 9 左)

$\theta EL1 = -sin^{-1}\frac{b}{a}$ かつ $\theta EL2 = -\theta EL1$ のとき

ケース 3： AZ1 と AZ3 が不定(図 9 右)

$$\theta EL2 = -\left\{sin^{-1}\left(\frac{a\,sin\,\theta EL1}{\sqrt{b^2+(c+d)^2}}\right) + tan^{-1}\frac{b}{c+d} + \theta EL1\right\}$$

かつ $\theta EL3 = \theta EL1 + \theta EL2$

かつ $\theta AZ2 = 0$ のとき

不定解発生時に操作者にとって扱いやすい挙動を MILS で検証した。その結果，$\theta AZ3$ と$\theta AZ2$は不定解発生直前の角度で固定し，$\theta AZ1$のみを駆動する方法を採用した。

このように，設計上の問題点とその対策結果を，実機なしで検証できることを確認した。

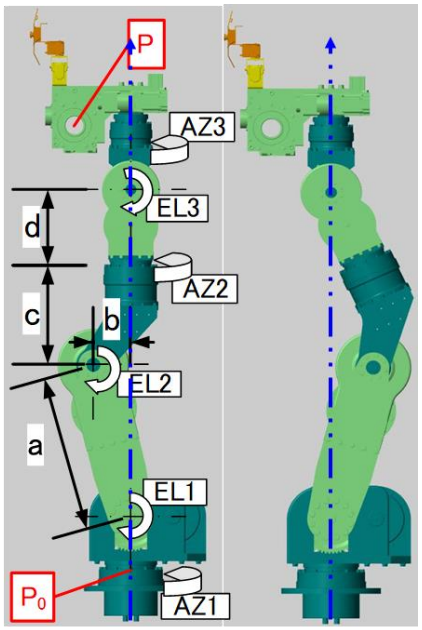


図 9. キネマティクス不定解

(2) 操縦者の感覚に合わせる

理論的なキネマティクスが操縦者の期待とは異なる結果を生じることが MILS で判明した。

正面を向いた状態から“12 度左に向く”と指令した場合，操縦者は図 10 の左側のように胴体全体を向ける動きを期待している。しかし，キネマティクスの演算結果は図 10 の右側のように胸の位置座標を固定し胴体を捻じった動作となる。加えて，この捻り動作では部品の干渉により 15 度程度までしか回転できないことが分かった。

このように，設計が人の感覚に合致しているかどうかなど曖昧な妥当性も，実機なしで検証できることを確認した。

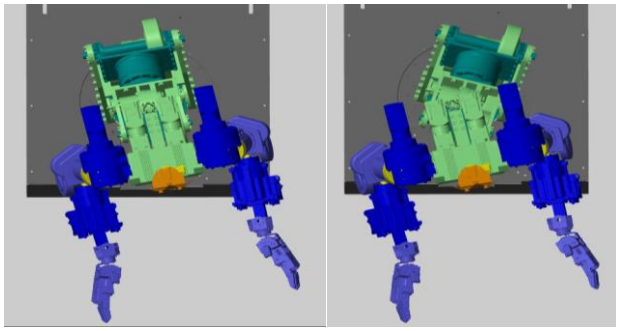


図 10. 胴体部回転動作の様子

3.2.2 駆動順序による衝突回避の検証

一例ではあるが，示指を何度以上曲げていると，拇指と衝突するかなど，機械的な衝突を MILS で検証した。衝突回避のためプログラムで制限を加えた場合も，所望動作の確認だけでなく，望まない副作用がないかどうかも，目視で検証できることを確認した。

3.3 自動コード生成

MATLAB の自動コード生成機能を用いて，検証された制御モデルから実行コードを作成した。実行コード作成には，コード生成を支援する“コード生成アドバイザー”を利用した。

“コード生成アドバイザー”により，コード化のための手順が提示されるため，容易にコードを生成することができた。

本項では，自動コード生成手順として

(1) 離散化
(2) 手書きコードとの結合
(3) 実施動作検証

を，自動生成したコードの評価として

(4) 処理時間評価

を述べる。

3.3.1 離散化

Simulink で書かれたモデルの時間系は連続系と離散系の 2 種類がある。シミュレーションモデルはトリガーを必要としない連続系の方が作りやすいが，コード化するモデルは離散系に変更（離散化）しておく必要がある。

注意点として，ライブラリーブロックによっては，“Continuous”グループや“Simscape”グループなど連続系でしか動作しないものもあり，再設計が必要になる場合がある。

3.3.2 手書きコードとの結合

図 11 は DiaroiD 制御プログラムのブロック構成図である。作成済みの手書きコードを元に，制御部の手書きコードと制御モデルの自動コードを入れ替えてビルドした。その他の部位は手書きコードを使用した。

3.3.3 実機動作検証

ビルドしたプログラムを DiaroiD に搭載し動作を確認した。

ここで特筆するべきは，動作確認開始時から問題なく動作したことである。MILS で検証したモデルは，コードレビューを必要とせず，コーディングミスを混入する余地もなく，実機で動作した。自動コード生成の優位性を証明する事象であった。

3.3.4 処理時間評価

制御部の処理時間には動作の遷移により 4～8ms のばらつきがある。

同一の動作をさせたときの手書きコードの平均処理時間は 5.80ms，自動生成コードの平均処理時間は 5.93ms であった。

図 12 は手書きコードと自動生成コードの処理時間のばらつきを，横軸：処理時間，縦軸：出現割合で示したグラフである。

処理時間のばらつきも含めて自動生成コードの処理効率は，手書きコードと遜色のないレベルにあることが分かる。

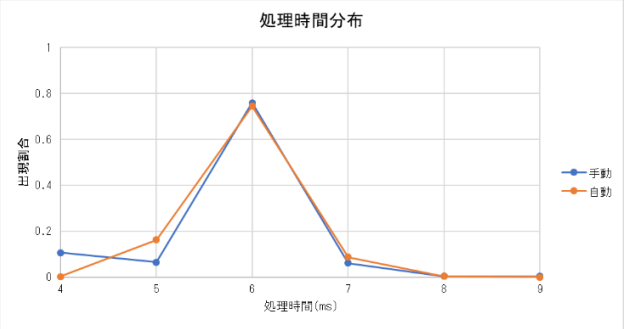


図 12. 処理時間のばらつきの比較

4. リアルタイムシミュレーション

リアルタイムシミュレーション技術を用いて，DiaroiD 操縦者を支援するシステムを構築した。

本章では，“DiaroiD 操縦支援システム”について述べる。

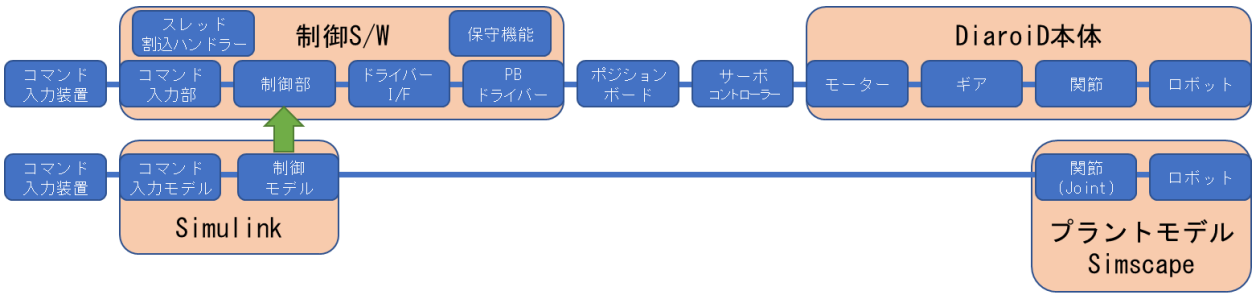


図 11. 実機構成とシミュレーション構成

4.1 DiaroiD 操縦支援システム

4.1.1 実時間への対応

MILS による検証が目的のシミュレーションでは，コンピューターのパフォーマンス不足によって，1 秒間の事象の演算に 2 秒かかっていても，問題にはならない。しかし，操縦支援が目的ならば，実時間内でシミュレーションを実施できなければ意味がない。

シミュレーション時間を計測したところ，実時間を超えていたため，操縦支援にはあまり影響しない手首のパラレルリンク機構を削除し，実時間内に納めるようにした。

具体的には，①制御モデルで逆キネマティクス演算前の角度指令を，プラントモデルの手首の“Joint”ブロックへの角度指令として接続した。さらに，②上腕と手首を接続しているリンクをプラントモデルから削除した。

これによりプラントモデル演算は軽減され，シミュレーション時間を実時間内に抑えることができた。

4.1.2 実周辺環境を仮想空間へ反映

Lidar(注5)を用いて DiaroiD を取り巻く実周辺環境を仮想空間へ投影する手法への取組みについて述べる。

仮想空間に DiaroiD 本体しか存在せず，作業対象物との距離の把握には，役に立っていなかった。そこで，DiaroiD に取り付けた Lidar により作業対象物を認識し，仮想空間に出現させることができないか検討を進めることにした。

検討当初は，Lidar による点群データを小さな球体として仮想空間に浮遊させる方法を考えたが，Lidar の解像度 2 百万点に対し，Simscape に実時間内で処理させられる球体は 100 個程度であり，実現は不可能であった。

次善策として，点群データから STL データを作り，剛体として仮想環境に投影する方法を試行した。

当社は Lidar を保有していないため，Lidar による撮影は，三菱電機情報技術総合研究所に実施いただいた。

図 13 の実周辺環境を Lidar で撮影すると，図 14 のとおり視線距離によって色分けされた点群データとなった。

この点群データを STL データに変換し，仮想空間に読み込んだ結果が図 15 である。1 部品となるため色分けはできず，単色（緑色）で表示した。

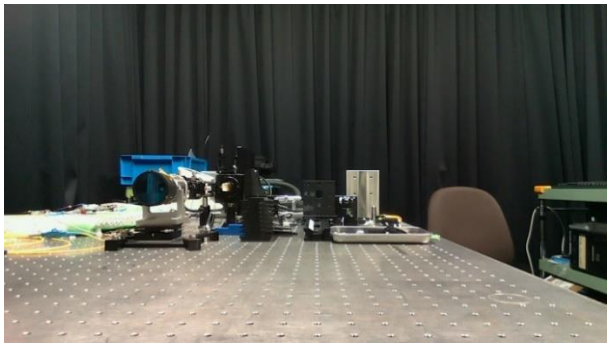


図 13. 実周辺環境

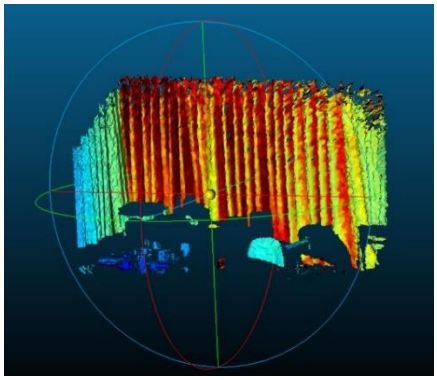


図 14. Lidar による点群データ

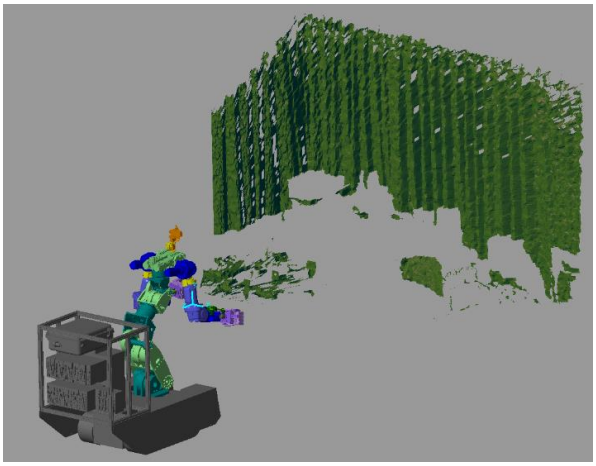


図 15. 仮想空間に取り込んだ実周辺環境

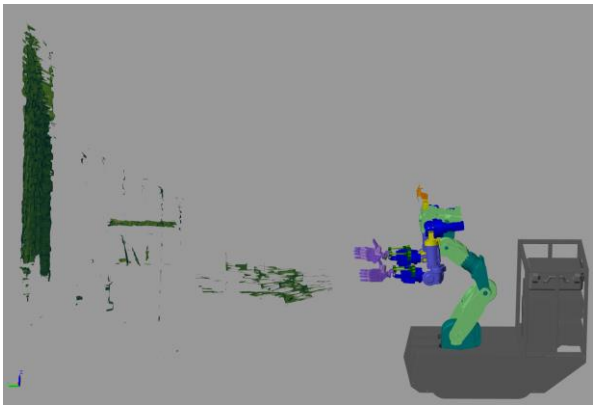


図 16. 仮想空間に取り込んだ実周辺環境（横から）

(注5) Light Detection and Ranging（光検出と測距）とは，光を用いたリモートセンシング技術のひとつ。パルス状に発光するレーザー照射に対する散乱光を測定し，遠距離にある対象までの距離や方向を計測するものである。

この方法において大きな問題点が 2 点あった。

1 点目は，対象の更新には STL データを入れ替えて再ビルドする手続きが必要であり，最速でも 1 分かかることである。リアルタイムシミュレーションにおいて実周辺環境の動的変化に追従することができない。

2 点目は，単一方向からの Lidar 計測では，1 面しか捉えることができず，視点を移動すると図 16 のようになり，物体に見えないことである。

今後，“①仮想空間を UNITY などの 3D プラットフォームに移行し，点群データ再現を高速化する。”，“②Lidar を DiaroiD の腕に装着し，多視点からの点群データを取得可能にする。”などの機能向上を検討する。

4.1.3 仮想視点

リアルタイムシミュレーションにおける仮想空間の長所のひとつは視点を自由に追加できる点である。DiaroiD の操縦システムでは，ロボット取り付けられたステレオカメラ 2 視点（頭部と胸部のステレオカメラ）のみであるが，仮想空間では俯瞰視点はもちろん，可動部に仮想的なカメラを取り付けたかのような視点を作ることができる。そこで，操縦者に任意の視覚情報を与えるため，手のひらに仮想視点を設けた。

図 17 は，左手に固定した視点でのモニター例である。

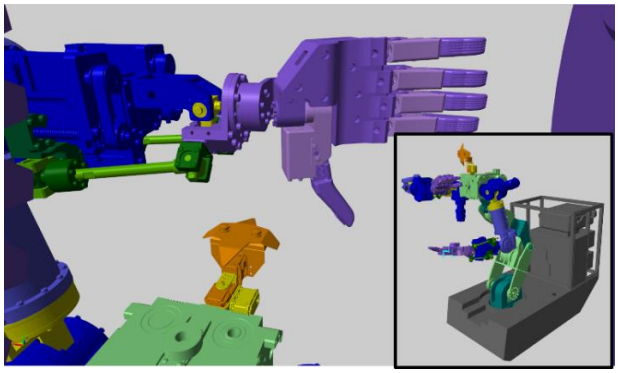


図 17. シミュレーションによる仮想視点

4.1.4 双腕衝突警告

操縦支援システム導入前は，右手の操作に気を取られている操縦者が，左腕への注意を配ることができず，双腕を衝突させてしまう事故があった。そこで，双腕の衝突を警告する機能をプラントモデルに組み込んだ。

定義された 2 つの部品間の距離が近づくと間欠音を発し，さらに，部品間の距離が近くなるほど間欠音の周期を短くして，操縦者に接近の度合いを伝えるようにした。

衝突警告を，衝突する可能性のある全ての部品間に適用すると，リアルタイム性が確保できなかったため，右手部と左手部の衝突に限定した。さらに，衝突判定件数を下げるため指の駆動範囲の形をした透明な仮想部品を作成し，双腕の接近を警告するようにした。

5. 事業展開への展望

本開発の最大の目標である事業，製品開発への展開について，これまでの成果を踏まえて考察する。

5.1 長所と短所

モデルベース開発の長所については，様々な文献がその有用性を示しているが，今一度整理する。

5.1.1 長所

(1) 明確な仕様伝達が可能である。

モデルが動く仕様書であり，従来のソフトウエア設計にあったような日本語の曖昧な表現が入り込む余地がなく，シミュレート可能であるため誤解を生じる可能性も少ない。また，シミュレートによるレビューでは，レビューアの技量に設計品質が左右される傾向も減じることができる。

(2) MILS による早期の検証ができる。

コーディングすることなく，シミュレーションによる設計検証が可能であり，ハードウエアの完成を待たずに妥当性検証が行える。

(3) 自動コード生成ができる。

コード作成工数を大幅に削減することができ，コーディングに起因するヒューマンエラーも無くすことができる。

(4) モデルが資産になる。

従来，ソフトウエア資産といえば，ソースコードとその仕様書であったが，モデルは仕様書と動作検証が一体化しており，資産の利用者にとって再利用にかかる手間を大きく削減できる。

(5) 技術継承に有利である。

取り上げている文献は少ないが，技術継承のツールとしてもモデルは優れている。シミュレーションによる直感的理解から理論的理解へ進むことができる。

5.1.2 短所

(1) 技術の習得に手間と時間がかかる。

MATLAB，Simulink，Simscape を習得しなければ，効率的な開発が望めない。特に Simscape の習得には時間がかかる。

(2) プラントモデル作成に手間と時間がかかる。

プラントモデルには製作対象である制御モデルより多くの工数を必要とする。

従来の開発では,制御対象の模擬には通信用シミュレーターを作成していたが,通信用シミュレーター作成工数とプラントモデル作成工数の差は大きい。

ただし，可視化したプラントモデルは，モデルベース開発の最大のメリットである MILS による早期検証に大きな役割を持つため，安易な工数削減は推奨できない。

(3) 従来のメトリクス指標が使えない。

従来開発と比べて V 字モデルにおける設計フェーズと試験フェーズの配分が変わり,設計フェーズの比重が大きくなる。そのため，従来のメトリクス指標を適用することができず,モデルベース開発用のメトリクス指標が必要になる。

5.2 ソフトウエア部位視点

MATLAB/Simulink は，行列演算ツールを元に発展したツールであるため，得意分野と不得意分野がある。

5.2.1 得意とする分野

(1) 演算処理

高度な演算ライブラリーが充実しており,非常に有用である。

(2) 手続き処理

分岐，繰り返し，状態遷移はブロック化できるので，複雑な状態分岐の検証には有用である。

5.2.2 不得意とする分野

(1) ユーザーインターフェース

MATLAB/Simulink はユーザーインターフェースを構築する能力は脆弱である。ユーザーインターフェースが主機能となる製品には向かない。

(2) 外部インターフェース

Socket 通信のライブラリーがなく，自作するより他なかった。コンピューターリソースを使用するインターフェース部分はモデルを自作する必要がある。データを型変換して引き渡すようなインターフェースが主機能となる製品には向かない。

(3) スレッディング

スレッド生成や割込みハンドラー登録などを書き表すことができない。

異なるタイミングで動作する離散化ブロックを定義し，離散化ブロックの自動生成コードを手書きのスレッドプログラミングとマージする処置が必要である。

5.3 開発工数視点

図 18 は，制御部について，従来開発手法の開発工数を“1”としてモデルベース開発手法の開発工数を比較したものである。

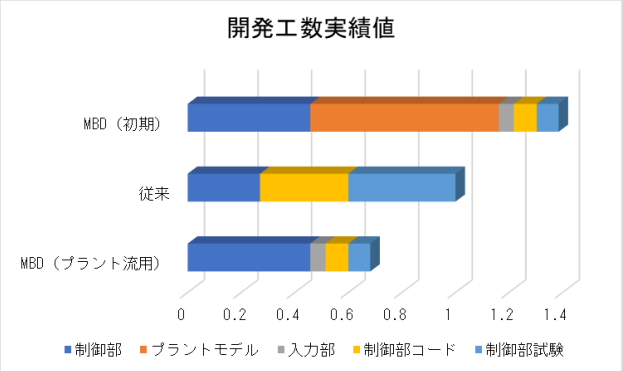


図 18. 開発工数

制御対象となるプラントモデルの構築に多くの工数が必要になるため，モデルベース開発は従来手法の約４割（38.5%）の工数増となる結果が得られた。

なお,プラントモデルを流用する場合は従来手法の約４割（37.5%）減の工数で開発できることが分かった。

5.4 考察

これらの成果を踏まえ，プラントモデルを含むモデルベース開発の事業化について考察する。

(1) 適用部位を広げすぎないこと

ユーザーインターフェース,外部インターフェースとスレッディングなど不得意な分野は従来手法で，演算機能,手続き処理など得意とする分野はモデルベース開発手法で製作し，マージする処置が必要である。

(2) 制御対象を理解していること

プラントモデルを作成するには,制御対象の構造を理解し振る舞いを定義する見識が必要である。

(3) プラントモデルを流用できること

プラントモデルを一度しか使わなければ，損失になる。パラメーターを変えながら何度も検証を行うような開発や,よく似たハードウエアを繰り返し使用するプロダクトラインに適用するのが良い。

自動化試験の導入検討とよく似ているが,自動化試験ではコードの流用率が高いこと。モデルベース開発ではプラントモデルの流用率が高いことが選定の基準になる。

(4) 初期費用が確保できること

導入時は技術習得コスト及びプラントモデル作成コストのため費用が増大する。初期費用が確保できない状況で実施すると,有意義なプラントモデルを製作できなくなり,後の改善効果も見込めない事態が予想される。

予算の範囲で,少しずつ適用部位を広げていく方法が適切と考える。

上述の条件がクリアできる事業であれば,モデルベース開発の適用を検討する価値があると考える。

謝　辞

モデルベース開発にあたり,DiaroiD の使用を許可いただいた三菱電機通信機製作所インフラ情報システム部をはじめ,同製作所技術部,三菱電機先端技術総合研究所,三菱電機情報技術総合研究所,三菱電機エンジニアリングに,深く感謝の意を表する。

商　標

(1) DiaroiD は,三菱電機(株)の登録商標である。
(2) MATLAB, Simulink, Simscape は,米国 The MathWorks, Inc.の登録商標である。
(3) Creo Parametric は,米国 PTC Inc.の登録商標である。
(4) UNITY は,米国 Unity Technologies 又はその関連会社の登録商標である。

参考文献

(1) 春名正樹,川口昇,猿田祐輔,星野隼人,岡本俊樹,道田塁,上田賢,神田吉孝,佐藤亨,小池俊昭,荻野正樹:人×機械の遠隔融合システムの開発 - 視覚的力触覚を利用した操作提案と基礎検証 -,情報処理学会インタラクション 2020 論文集,448～453 (2020)
(2) Shota Narasaki, Noboru Kawaguchi, Masaki Hirano, Toshitaka Nakaoji, Yutaka Ezaki, Yusuke Saruta, Hayato Hoshino, Masaki Haruna, Rui Michida, Yosuke Tomida, Yoshitaka Koda, Toru Sato : Development of labor-saving system for the maintenance of the telescopes under the extreme environment using a remote-controlled robot , SPIE Astronomical Telescopes + Instrumentation Conference (2020)

執 筆 者 紹 介

神田　吉孝(こうだ よしたか)
1991 年入社。
衛星通信・宇宙開発に関するソフトウエア開発に従事。
現在,第１事業部に所属。