RT コンポーネント操作マニュアル

メディアアートコミュニティ実現に向けた RT コンポーネントの開発と提案

2014 年 10 月 31 日版

芝浦工業大学大学院

ロボティクスシステムデザイン研究室

土屋彩茜　立川将　遠藤太貴　佐々木毅

更新履歴

2014 年 10 月 31 日版　第 1 版.

2014 年 10 月 31 日版　第 2 版　HOTMOCK_master のポート名修正。図 12 の訂正。

目次

1. 本提案の概要

1.1 提案の背景

メディアアーティストやデザイナが簡単に RT 技術を利用し、メディアアートのようなインタラクティブな作品の制作が出来る環境が実現できれば、分野の発展が期待されます。そのための環境作りとして、RT コンポーネント開発者とメディアアート製作者によるコミュニティを提案し、実現に向けた活動を行いました。このコミュニティ活動より得られたフィードバックよりコンポーネントの開発を行っています。

その中から、今回はデザイナが使いやすいツールとなるコンポーネントの開発として、デザイナのフィジカルプロトタイピング用のツールである HOTMOCK（株式会社ホロンクリエイト）のコンポーネント化を行いました。

コンポーネントの設計指針等につきましては、

土屋 彩茜, 立川 将, 佐々木 毅,
“メディアアートへのRTミドルウェアを用いた開発手法の提案”,
第13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, 2013.

土屋 彩茜, 立川 将, 遠藤 太貴, 佐々木 毅,
“メディアアートコミュニティ実現に向けたRTコンポーネントの開発と提案”,
第14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, 2014.

に詳細がありますので、そちらもご参照いただけましたら幸いです。

1.2 開発環境

本コンポーネント群は Windows にて動作確認を行いました。開発環境は以下の通りです。

- Windows 7（64bit 版）
- RT ミドルウェア：OpenRTM-aist-1.1.0-RELEASE（C++版）
- コンパイラ：Microsoft　Visual　C++ 2010　Express（SP1）
- Eclipse：Eclipse 3.8.1 + OpenRTM Eclipse tools 1.1.0-RC4
- CMake：CMake 2.8.8

1.3 必要システム

今回使用している HOTMOCK（株式会社ホロンクリエイト）の動作確認済み OS は以下の通りです。

- Windows Vista/7/8

また、現在プロトタイピング用ツールコンポーネントとして開発した HOTMOCK コンポーネントでは XML ファイルの解析に Microsoft XML パーサー（MSXML）を使用しているため、実行環境に msxml6.dll があることを確認してください。

1.4 開発したコンポーネント群

プロトタイピング用ツールコンポーネントとして、以下のコンポーネントを開発しました。

・HOTMOCK コンポーネント（HOTMOCK_master）
HOTMOCK デバイスを使用するためのコンポーネント。

また、HOTMOCK_master の使用手順と実用例を示すために、以下のコンポーネント群を開発しました。

・閾値による二値化コンポーネント（Thresholding）
入力された値と閾値から二値化された値を出力するコンポーネント
・HOTMOCK による Kobuki スイッチ操作コンポーネント
（KobukiControllerByHMSwitches）
Kobuki を HOTMOCK のスイッチで操作するためのコンポーネント

それぞれのコンポーネントの詳細については 2 章を、HOTMOCK_master の使用方法については 3 章を参照してください。

1.5 利用したコンポーネント群

今回開発した HOTMOCK_master の使用例と実用例を示すために、以下のコンポーネントを利用いたしました。

・プレゼンテーションコンポーネント（CVPresentation）
http://www.openrtm.org/openrtm/ja/project/contest2013_1B3-3
・Kobuki コンポーネント（KobukiRTC）
https://github.com/rt-net/kobuki_rtc

各コンポーネントの下に表記したページは開発者様のサポートページとなっておりますので、詳細につきましてはそちらのサポートページを参照してください。

また、HOTMOCK_master を開発する際、Dynamic_port を利用しています。こちらの詳細に関しては http://www.openrtm.org/rt/RTMcontest/2008/doc/1L3-2 を参照してください。

2. コンポーネントの説明

2.1 プロトタイピング用ツールコンポーネント

2.1.1　HOTMOCK

HOTMOCK（図 1）は株式会社ホロンクリエイトで開発されたフィジカルプロトタイピングツールです。電子工作やプログラミングの知識が無い人でも簡単にプロトタイピングできるというコンセプトで、UI（User Interface）デザイナによって開発されました。スイッチやセンサなどがモジュール単位でキット化されており、コアデバイスに繋ぐだけでそのモジュールが利用できるようになっています。別アプリケーション上で動作させることも可能で、デバイスとのソケット通信仕様が Web（http://www.hotmock.com）で公開されています。今回はこの通信仕様の下、開発を行いました。

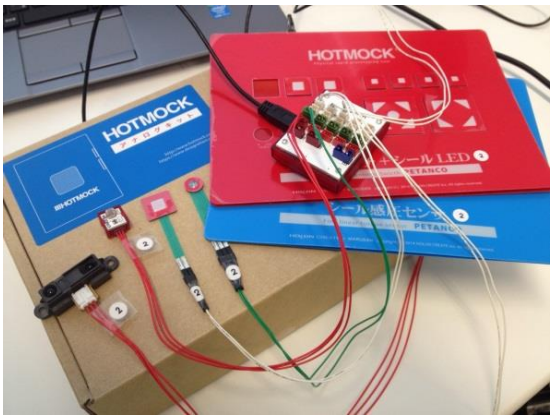

図 1：HOTMOCK（アナログキット）

2.1.2　HOTMOCK コンポーネント（HOTMOCK_master）

HOTMOCK_master は、HOTMOCK Setting とソケット通信を行い、HOTMOCK デバイスを使用するためのコンポーネントです。Configuration の SettingFilename で指定した.hmst ファイルを読み込み、使用しているデバイスに対応した InPort、OutPort を生成します。.hmst ファイルは HOTMOCK Setting によって生成されます。

InPort の DO*は HOTMOCK コアユニットの DO*に接続されているデバイスの値を取得します。AO*、Reset_PI*も同様です。OutPort の DI*は HOTMOCK コアユニットの DI*に接続されているデバイスにデータを出力します。AI*、PI*も同様です。これらのポートはアクティブ化する時に自動で追加・削除されます。

また、OutPort の TS、GS は HOTMOCK コアユニットに内蔵されている温度センサと加速度センサのデータを出力します。これらは設定に関わらず、常にポートが表示されています。

・InPort

名称	型	説明
DO*	DynamicInPort<TimedShort>	HOTMOCK デバイスの DO*に送信する デジタル出力値を取得するポート。
AO*	DynamicInPort<TimedDouble>	HOTMOCK デバイスの AO*に送信する アナログ出力値を取得するポート。
Reset_PI*	DynamicInPort<TimedBoolean>	HOTMOCK デバイスの PI*につながっているパ ルス入力の積算値をリセットするための フラグを取得するポート。

・OutPort

名称	型	説明
DI*	DynamicOutPort <TimedShort>	HOTMOCK デバイスの DI*から受信した デジタル入力値を送るポート。
AI*	DynamicOutPort <TimedDouble>	HOTMOCK デバイスの AI*から受信した アナログ入力値を送るポート。
PI*	DynamicOutPort <TimedDouble>	HOTMOCK デバイスの PI*から受信した パルス入力値を送るポート。
TS	TimedDouble	HOTMOCK デバイス内蔵の温度センサから 受信した値を送るポート。
GS	TimedDoubleSeq	HOTMOCK デバイス内蔵の加速度センサから 受信した値を送るポート。 値は要素数 0 から順に x,y,z の配列として 送られる。 後述する Configuration の GetDataType が 0 の場合は加速度、1 の場合は角度が送られる。

・Configuration

名称	型	デフォルト値	説明
SettingFilename	string	*.hmst	HOTMOCKSetting によって作成された HOTMOCK デバイスの設定ファイル名。
IPAddress	string	127.0.0.1	HOTMOCKSetting の IP アドレス。
PortNumber	int	8888	HOTMOCKSetting の Port 番号。
GetDataType	std::vector <int>	0,0,0,0	HOTMOCK デバイスからの入力を取得す る際、

	0 ならば生値を取得、1 ならば工業変換値を取得する。 配列 0 から順に ordered_list で指定し、AI,PI,TemperatureSensor,GyroSensor の設定をする*。

・*GetDataType の設定表

ポート名	0 の時	1 の時
AI*	現在値。	HOTMOCK Setting で設定した「最小値」「最大値」の範囲に変換した値。
PI*	同じ	
TS	同じ	
GS	加速度値。	角度値。

2.2 閾値による二値化コンポーネント（Thresholding）

Thresholding は、InPort の DoubleInData または ShortInData から入力された値と閾値を比較して、二値化した値を OutPort の OutData から出力するコンポーネントです。閾値は Configuration の Threshold､InPort の DoubleThreshold､ShortThreshold で指定することができ、Configuration の Threshold_Mode でどの閾値を用いるかを選択できます。また、OutPort の OutData から出力する値は、Configuration の OutDataValue で指定することができます。

・InPort

名称	型	説明
DoubleInData	TimedDouble	double 型のデータを入力するポート。 ShortInData とどちらか片方を接続する。
ShortInData	TimedShort	short 型のデータを入力するポート。 DoubleInData とどちらか片方を接続する。
DoubleThreshold	TimedDouble	閾値となる double 型を入力するポート。
ShortThreshold	TimedShort	閾値となる short 型を入力するポート。

・OutPort

名称	型	説明
OutData	TimedShort	二値化された値を出力するポート。

・Configuration

名称	型	デフォルト値	説明
Threshold	double	50.0	閾値。
Threshold_Mode	int	0	どの閾値を使用するか選択する。 0 ならば Configuration：Threshold、 1 ならば InPort：DoubleInData、 2 ならば InPort：ShortInData が閾値として用いられる。
OutDataValue	std::vector <short>	0,0,1	出力する値。 値は要素数 0 から順に閾値より小さい、 閾値と等しい、閾値より大きいとなる。 デフォルト値の 0,0,1 では 閾値以下は 0、閾値より上は 1 を出力する 設定となっている。

2.3 HOTMOCK による Kobuki スイッチ操作コンポーネント（KobukiControllerByHMSwitches）

KobukiControllerByHMSwitches は HOTMOCK のスイッチデバイスを用いて Kobuki の操作をするためのコンポーネントです。InPort は、それぞれのポートに繋がっているスイッチの値を取得するポートとなっています。HOTMOCK のスイッチデバイスは押すと 1、離すと 2 が、（長押しは 3 が HOTMOCK Setting で設定したときのみ）出力されます。したがって、これらの値が出力されるポート（HOTMOCK_master では DI*）と各 InPort を接続することで使用できます。1 の値を取得したポートに応じて、Kobuki に対して OutPort の Velocity から速度指令値を出力します。速度は Configuration で設定ができます。

・InPort

名称	型	説明
Forward	TimedShort	前進指令を出すスイッチの値を取得するポート。
Back	TimedShort	後退指令を出すスイッチの値を取得するポート。
Right	TimedShort	右回転指令を出すスイッチの値を取得するポート。
Left	TimedShort	左回転指令を出すスイッチの値を取得するポート。

・OutPort

名称	型	説明
Velocity	TimedVelocity2D	kobuki への速度指令値を出力するポート。

・Configuration

名称	型	デフォルト値	説明
Speed	double	0.2	Kobuki が前進・後退する時の速度。 制約条件は x>=0。単位は m/s。
RotSpeed	doube	0.5	Kobuki が右回転・左回転する時の速度。 制約条件は x>=0。単位は rad/s。

3. HOTMOCK コンポーネントの使用方法

3.1 RTSystemeEditor を用いたシステム構築の手順

RTSystemeEditor を用いたシステム構築は、通常以下の手順で行われます。

1. ネームサーバを起動する
2. RTSystemEditor を起動する
3. ネームサーバへ接続する
4. コンポーネントを起動する
5. システムを構築し、実行する

これらの手順はどのようなシステムでも共通です。RTSystemEditor の詳しい操作方法は OpenRTM-aist のホームページ（http://www.openrtm.org/）を参照してください。以下の節では手順 5 に関して説明します。

3.2 HOTMOCK_master の使用方法

3.2.1 HOTMOCK の準備

HOTMOCK_master を実行する前に、HOTMOCK の準備を行う必要があります。その手順をまとめると以下のようになります。

0. HOTMOCK のセットアップ
   *HOTMOCK 動作確認済み OS：Windows Vista/7/8

---

1. HOTMOCK デバイスの準備をする
2. HOTMOCK Setting の起動・環境設定を行う
   (a) コアユニットの型を選択
   (b) 環境設定で接続ポートを選択する
3. HOTMOCK Setting のデバイス設定を行う
4. HOTMOCK Setting をシミュレーションモードにする

HOTMOCKのセットアップに関してはHOTMOCKに付属しているセットアップマニュアルを参照してください。また、HOTMOCK にはソフトウェアが付属しています。その付属CD 内の「HOTMOCK ソフトウェアインストールマニュアル.pdf」に従ってインストールを行ってください。

この章では HOTMOCK アナログキットを例に、手順 1 から 4 について説明します。基本的に他のキットでも使用方法は同じですが、使えるデバイスの種類が異なります。各キットの詳しい使用方法は HOTMOCK 付属のセットアップマニュアル及び HOTMOCK 付属 CD 内の「HOTMOCK Setting ユーザーマニュアル.pdf」を参照してください。

手順 1：HOTMOCK デバイスの準備をする

HOTMOCK のコアユニットを PC に接続します。コアユニットの LED が赤く点灯していることを確認してください。点灯していなかった場合、ドライバのインストールに問題があるので、手順 0 のセットアップをやり直してください。点灯していたら、コアユニットに使用するデバイスを接続してください（図 2)。接続の詳しい方法は HOTMOCK 付属のセットアップマニュアルを参照してください。

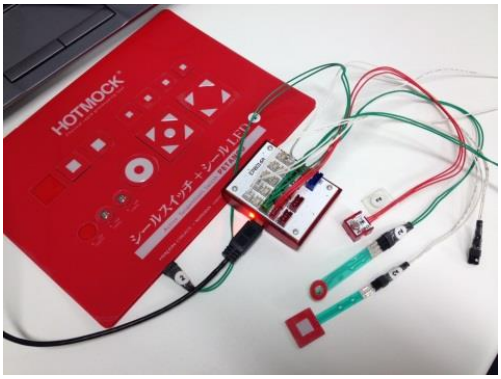

図 2：HOTMOCK デバイスの準備

手順 2：HOTMOCK Setting の起動・環境設定を行う

付属 CD からインストールした HOTMOCK Setting.exe を起動します。はじめにコアユニットの型を選択します。レガシーキット・デジタルキットの場合はデジタルボード、アナログキットの場合はアナログボードを選択してください。その後新規ファイルが開きます。

新規ファイルが開いたら、デバイスの設定を始める前に環境設定を行います。ツールバーの「編集」から「環境設定」を選択してください。その中の「通信設定」で接続ポート名を選択します（図 3)。接続ポートの調べ方として、Windows 7 では「コントロールパネル>ハードウェアとサウンド>デバイスマネージャー」から確認ができます。ポート名が正しく選択されると、デバイス設定画面で動作確認ができるようになります。起動・環境設定の詳しい説明は「HOTMOCK Setting ユーザーマニュアル.pdf」を参照してください。

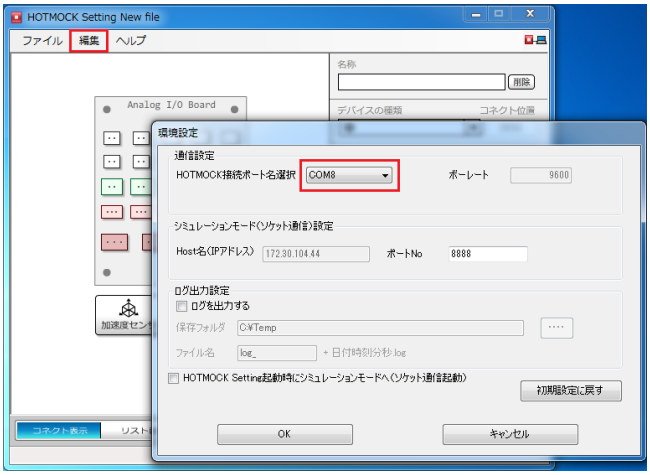

図 3：HOTMOCK Setting の環境設定

手順 3：HOTMOCK Setting のデバイス設定を行う

環境設定が完了したら、デバイス設定を行います。デバイス設定の詳しい方法については「HOTMOCK Setting ユーザーマニュアル.pdf」を参照してください。

デバイス設定が完了したら、ツールバーの「ファイル」から保存を行ってください。この時保存したファイル名が、HOTMOCK コンポーネントで読み込むファイル名になります。

また、以前保存した設定を利用する場合は、ツールバーの「ファイル」から、使用する設定ファイルを開いてください。

手順 4：HOTMOCK Setting をシミュレーションモードにする

手順 1～3 が完了したら、画面右下の「シミュレーションモード」を選択してください。表示画面が「シミュレーション待機中」（図 4）になれば HOTMOCK の準備は完了です。

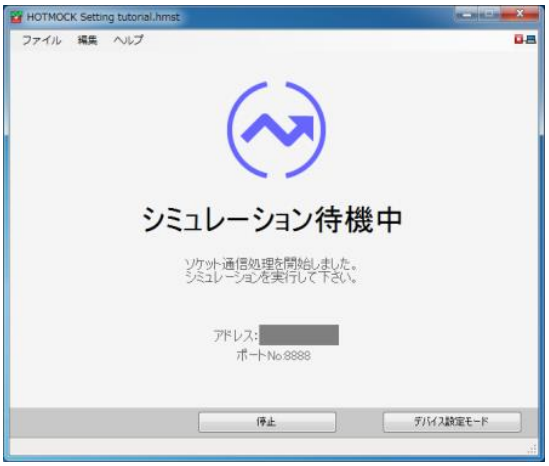

図 4：HOTMOCK Setting の「シミュレーションモード待機中」画面

3.2.2　HOTMOCK_master の使用手順

HOTMOCK_master の使用手順をまとめると以下のようになります。

1. Configuration を設定する
   (a) SettingFilename から HOTMOCK Setting の設定ファイル名を指定する
   (b) IPAddress に HOTMOCK Setting を起動している PC の IP アドレスを設定する
   (c) PortNumber に HOTMOCK Setting で使用しているポート番号を設定する
   (d) GetDataType に取得するデータの種類を設定する。
2. コンポーネントをアクティブ化する
3. 入出力ポートを他のポートと接続する

手順 1(d)については手順 2 のアクティブ化後に変更することが可能です。手順(a)～(c)についてはアクティブ化するときに使用するので、変更が生じた場合、一度非アクティブ化してから再度アクティブ化してください。

一度非アクティブ化すると、HOTMOCK Setting との接続が中断されます。再度 HOTMOCK コンポーネントを使用する場合は、一度 HOTMOCK Setting に戻り、サーバを立ち上げる必要があります。HOTMOCK Setting 画面下の「停止」選択後、「開始」を選択することで「シミュレーション待機中」の表示になりサーバが立ち上がります。この状態にしてから、再度アクティブ化してください。HOTMOCK Setting での表示が「停止中」もしくは「デバイス設定モード」の場合はアクティブ化するとエラーになるので、「開始」もしくは「シミュレーションモード」を選択し、再度「シミュレーション待機中」の表示にしてからアクティブ化を行ってください。

また、HOTMOCK Setting 側で開いている設定ファイルが、SettingFilename と異なる場合、正常に通信が行われないので、HOTMOCK Setting 側で使用する設定ファイルを開いた後、再度「シミュレーションモード」を選択してください。

次章では具体例を挙げながら使い方を説明します。

3.3 HOTMOCK デバイスの使用例

HOTMOCK デバイスの内、デジタル入出力を使った最も基本的なテストを行います。今回は例として DI16 のデバイスにスイッチを、DO7 のデバイスに LED（青）を設定し、スイッチを押したら LED が光るというシステムを構築します。

使用するコンポーネントは HOTMOCK_master 1 つです。以下に使用手順を示します。3.2 で述べた手順で説明を行います。

HOTMOCK 準備：手順 1

HOTMOCK デバイスの DI16、DO7 にスイッチと LED を接続する。

HOTMOCK 準備：手順 2,3

HOTMOCK Setting の起動・環境設定、デバイス設定を行います。今回は図 5 のように設定しました。

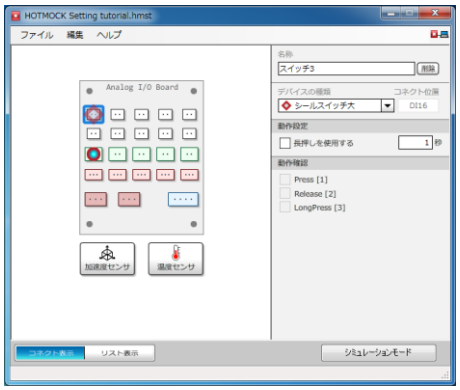

図 5：HOTMOCK Setting のデバイス設定例

HOTMOCK 準備：手順 4

HOTMOCK Setting をシミュレーションモードにし、「シミュレーション待機中」にする。

HOTMOCK_master 使用手順 1：Configuration を設定する

コンポーネントを起動し、RTSystemEditor 上に配置します。配置した段階ではポートの形は図 6 のようになっています。この HOTMOCK_master を選択し、Configuration View から設定を行います。

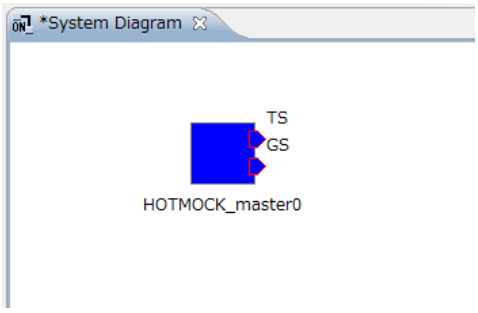

図 6：HOTMOCK_master の初期状態

ここでは、使用しているデバイスに対応したポートを生成するため、設定ファイルの指定が必要となります。また、HOTMOCK Setting とソケット通信を行うための IP アドレスとポート番号を指定する必要があります。

まず、設定ファイルの指定は SettingFilename で行います。デフォルト値ではエラーとなるので、必ず設定を行ってください。指定方法はファイルが保存されているパス+ファイル名となっています。

次に、IP アドレスとポート番号の指定を行います。IP アドレスについては、HOTMOCK Setting と HOTMOCK コンポーネントが同一 PC 上で実行されている場合、デフォルト値を変更する必要がありません。異なる PC 上で実行される場合、HOTMOCK Setting の IP アドレスを設定してください。ポート番号についても基本的にデフォルト値を変更する必要はありませんが、HOTMOCK Setting 上で変更を行った場合、そちらと同じ番号に変更してください。IP アドレスは図 7 の赤部分、ポート番号は図 7 の青部分に表示されます。

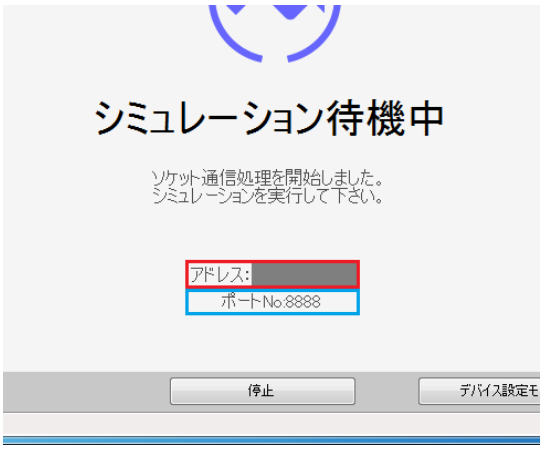

図 7：HOTMOCK Setting の IP アドレスとポート番号

HOTMOCK_master 使用手順 2：コンポーネントをアクティブ化する

HOTMOCK Setting を確認し、画面が「シミュレーション待機中」になっていたら、HOTMOCK_master を選択し、右クリックメニューの Activate をクリックしてアクティブ化します。これにより HOTMOCK_master と HOTMOCK Setting 間でソケット通信を開始します。この時、HOTMOCK_master のコンソールは図 8、HOTMOCK Setting の画面は図 9 のように表示されています。

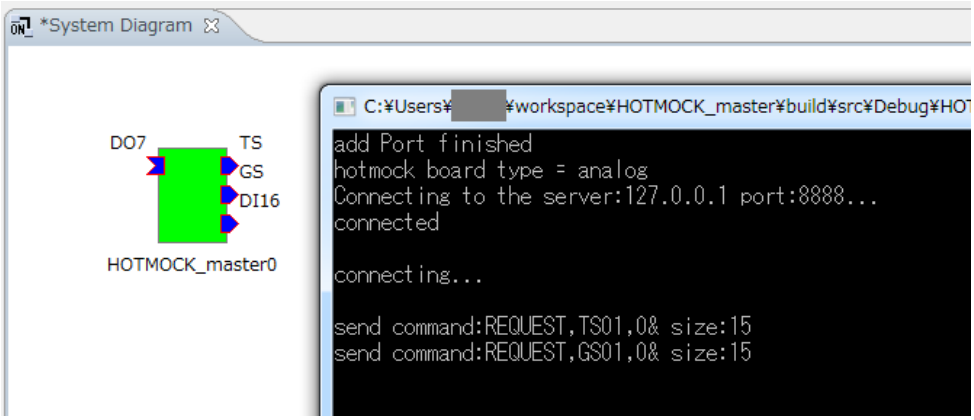

図 8：ソケット通信が成功した時の HOTMOCK_master のコンソール画面

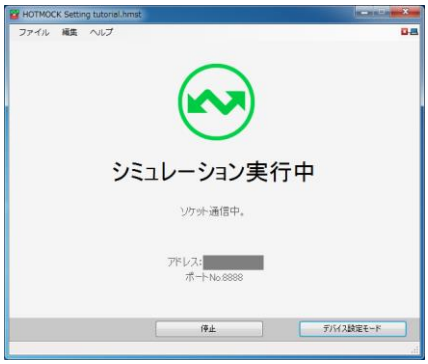

図 9：ソケット通信中の HOTMOCK Setting（「シミュレーション実行中」）

HOTMOCK_master 使用手順 3：入出力ポートを他のポートと接続する

アクティブ化が完了したら、デバイスに対応したポートが生成されています。ポートが生成されたことを確認したら、接続を行います。今回は、DI16 と DO7 のデバイスを使用しているので図 10 のように接続します。

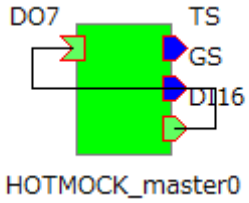

図 10：HOTMOCK_master の接続例 1

この状態で、スイッチを押すと LED（青）が点灯します（図 11）。

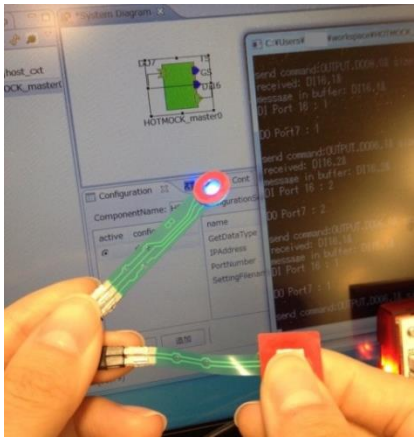

図 11：HOTMOCK デバイスの LED（青）

今回のように InPort と OutPort を直接接続することで、HOTMOCK デバイスとのデータのやりとりが可能となります。

4. 既存 RTC との組み合わせ

この章では、HOTMOCK コンポーネントと既存の RTC を組み合わせることで、応用例の提示を行います。

4.1 プレゼンテーションコンポーネント

フィジカルプロトタイピングを行う際、前節のように HOTMOCK デバイスの組み合わせによって簡単に動作モデルを試すことができます。しかし、デバイスには限りがあるため、より複雑な設定・情報を伝えにくい場合があります。このようなとき、アイディアスケッチを動作させることができれば、より複雑な動作モデルも簡単にイメージを伝えることができます。そこで、HOTMOCK コンポーネントとプレゼンテーションコンポーネントを組み合わせて、動作するアイディアスケッチを実現します。

今回はアナログ入力の説明も兼ねて、HOTMOCK 側のデバイスとしては AI1 のデバイスに照度センサを設定します。これを用いて「部屋が暗くなったら電気が点き、部屋が明るくなったら電気が消える照明」の動作型アイディアスケッチのシステムを構築していきます。使用するコンポーネントは

- HOTMOCK コンポーネント（HOTMOCK_master）
- 閾値による二値化コンポーネント（Thresholding）
- プレゼンテーションコンポーネント（CVPresentation）

の 3 つです。

HOTMOCK デバイス及び HOTMOCK コンポーネントは 3.3.1 と同様の手順で使用します。本章では、「HOTMOCK_master 使用手順 3：入出力ポートを他のポートと接続する」について説明を行います。

HOTMOCK_master 使用手順 3：入出力ポートを他のポートと接続する

HOTMOCK_master のアクティブ化が完了したら、デバイスに対応したポートが生成されています。ポートが生成されたことを確認したら、他 2 つのコンポーネントを図 12 のように配置し、ポートの接続を行います。

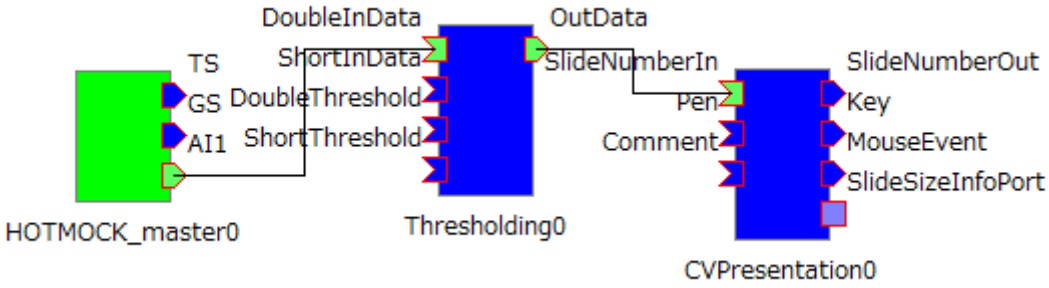

図 12：HOTMOCK_master の接続例 2

次に Configuration の設定を行います。プレゼンテーションコンポーネントの Configuration は、スライドのパスとスライド名を変更し、SlideNumberInRelative を 0 に変更してください。これにより、指定されたスライド番号のスライドが表示されるようになります。閾値による二値化コンポーネントの Configuration は、OutDataValue を 1,1,2 に変更してください。これにより、閾値を Configuration の Threshold で指定し、閾値以下なら 1、閾値より大きければ 2 を出力するという設定が行われたことになります。Threshold は明暗の閾値として用いるので、部屋の明るさ等に応じて調節してください。Threshold はアクティブ化後に変更することが可能となっています。

Configuration の設定が終わったら、それぞれアクティブ化してください。プレゼンテーションコンポーネントをアクティブ化すると、図 13 のようにスライドが表示され、部屋の明暗によって表示が変更されるようになります。

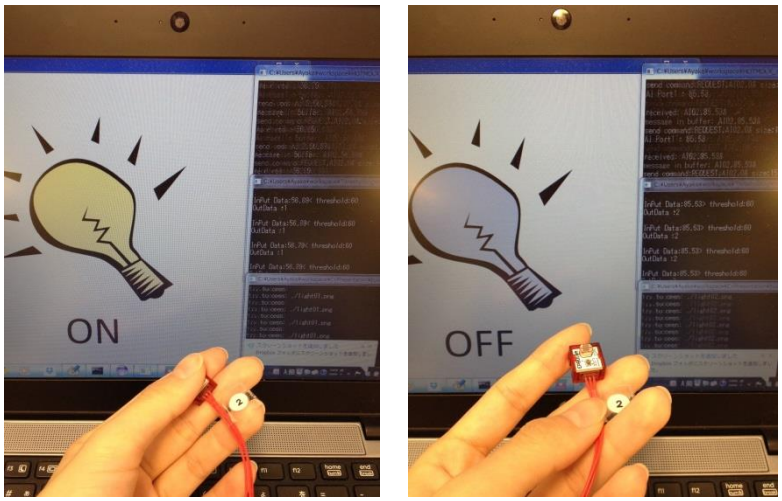

図 13：（左）暗いとき電球が ON　（右）明るいとき電球が OFF

4.2 移動用ロボット：Kobuki

HOTMOCK を RT コンポーネント化することで、既存の RTC と接続することができるようになりました。これにより、デザイナが操作できるデバイス・ハードの幅が広がったと言えます。そこで実際に HOTMOCK コンポーネントを使ってロボットを動かすことができれば、デザイナの人に「RT ミドルウェアを使えばもっといろいろなものが動かせる」ということを伝えることができると考えられます。そこでデモンストレーションとして、HOTMOCK デバイスのスイッチによって移動用ロボットの Kobuki（株式会社アールティ）を操作する簡易コントローラのシステムを構築します。

HOTMOCK デバイスとしては

・スイッチ　4 つ

使用するコンポーネントは

・HOTMOCK コンポーネント（HOTMOCK_master）
・HOTMOCK による Kobuki スイッチ操作コンポーネント
（KobukiControllerByHMSwitches）
・Kobuki コンポーネント（KobukiRTC）

の 3 つを用います。また、HOTMOCK デバイスを立ち上げる PC と Kobuki と接続する PC の 2 台を用意します（図 14）。

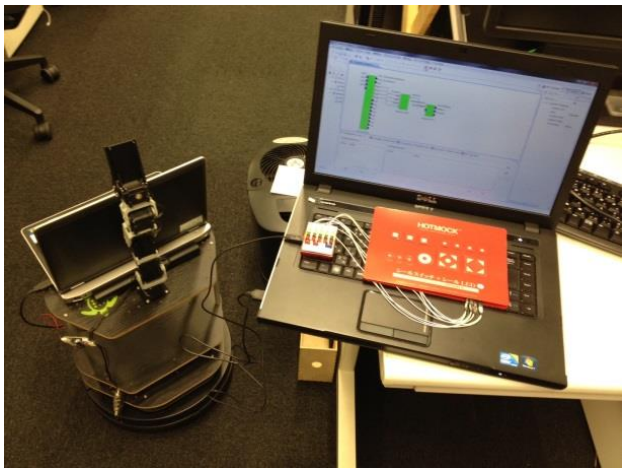

図 14：Kobuki と HOTMOCK

HOTMOCK デバイス及び HOTMOCK コンポーネントは 3.3.1 と同様の手順で使用します。本章では、「手順 3：入出力ポートを他のポートと接続する」について説明を行います。

HOTMOCK_master 使用手順 3：入出力ポートを他のポートと接続する

先に Kobuki 側の PC で Kobuki コンポーネントを立ち上げます。その後、HOTMOCK 側の PC に Kobuki 側の PC のネームサーバを追加することで HOTMOCK 側からこのコンポーネントが利用できるようになります。

HOTMOCK コンポーネントのアクティブ化が完了したら、他 2 つのコンポーネントを図 15 のように配置し、ポートの接続を行います。

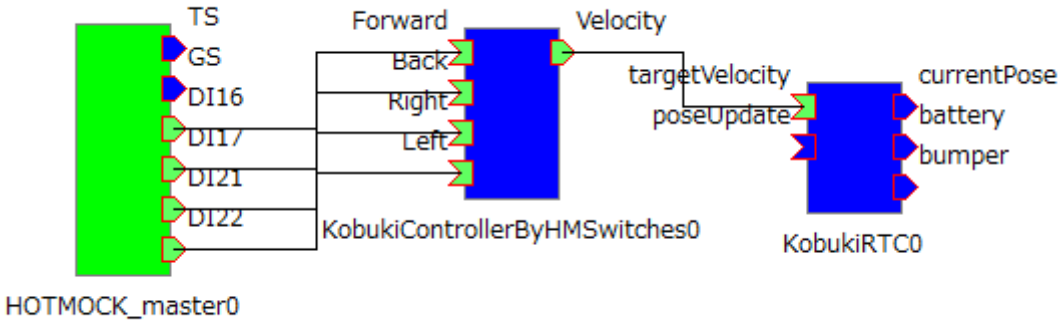

図 15：HOTMOCK_master の接続例 3

次に Configuration の設定を行います。今回は特に変更すべき Configuration はありませんが、状況に合わせて HOTMOCK による Kobuki スイッチ操作コンポーネントの Speed と RotSpeed を変更してください。これらの Configuration はアクティブ化後に変更することが可能となっています。

Configuration の設定が終わったら、それぞれアクティブ化してください。この時 Kobuki の電源が ON になっているか確認してください。全てアクティブ化すると、Kobuki を HOTMOCK スイッチにより操作することができます。

前進・後退、右回転・左回転のスイッチを同時に押した場合は、お互いの指令で打ち消しあい、Kobuki は停止します。前進と右（左）回転、後退と左（右）回転のスイッチを同時に押した場合は、旋回しながら走行します。

5. FAQ

Q1： HOTMOCK デバイスの反応がない。

Ans.： まず HOTMOCK コアユニットの LED が赤く点灯しているか確認してください。点灯されていない場合、ドライバが正しくインストールされていない恐れがあります。点灯している場合は、HOTMOCK Setting の環境設定で接続ポートが正しく設定されているかを確認してください。

Q2： HOTMOCK コアユニットの内蔵センサが使えない。

Ans.： HOTMOCK コアユニットの内蔵センサは加速度センサと温度センサがあります。内蔵センサなので、実際に何かデバイスを接続する必要はありませんが、これらを使用するためには HOTMOCK Setting のデバイス設定画面中央下の加速度センサおよび温度センサをクリックし、画面中央右の「動作設定」のところで「加速度センサ（温度センサ）を使用する」にチェックをいれてください。

Q3： HOTMOCK Setting で設定しているはずのデバイスの値がこない。

Ans.： これには主に 2 つの原因が考えられます。

1. 読み込んでいる HOTMOCK Setting のファイルが違う
2. HOTMOCK Setting で設定ファイルを読み込んでいない

まずは HOTMOCK_master の Configuration : SettingFilename を確認してください。また、指定した場所に設定した.hmst ファイルがあることを確認してください。次に、HOTMOCK Setting のデバイス設定モードで指定したファイルが開いていることを確認してください。HOTMOCK は HOTMOCK Setting のデバイス設定モードで使用しているデバイスに対してのみ、データを取得します。

Q4： HOTMOCK デバイスを Windows 以外で構築している RT ミドルウェアのシステムと接続したい。

Ans.： HOTMOCK の動作確認済みの OS が Windows Vista/7/8 となっていますので、別 PC でこれらに対応する OS を用意してください。また、現在 HOTMOCK_master も Windows のみの対応となっていますので、HOTMOCK を接続している PC で HOTMOCK_master を立ち上げて、その PC のネームサーバを追加する形で利用してください。

6. コミュニティ

現在 OpenRTM-aist の公式 Web ページでは、プロジェクトとしてユーザが作成した様々なコンポーネントやツールを登録・利用することができます。このようにメディアアートに関するコンポーネントを、紹介・共有できる環境作りを目指しています。そのため、現在は blog や Twitter にて意見交換や情報収集・発信をしています。このような場から、どのようなコンポーネントが必要とされているか、どうすればより伝わりやすい・使いやすいのかといったフィードバックを得て、開発・デモンストレーション等行っております。

是非一度ご覧になっていただけたら幸いです。

Blog：MediA-RT

URL：http://rtmediaart.blog.fc2.com/

Twitter:：@MediART2013

URL：https://twitter.com/MediART2013

7. お問い合わせ

本コンポーネント群につきましては、フィードバックに対し随時修正・開発・公開をしているため、まだまだ展開・改善の余地があるものと考えております。提案・要望・バグ報告・マニュアル記述の不備等に関しましては下記までご連絡ください。

【問い合わせ先】

〒108-8548

東京都港区芝浦 3-9-14,611

芝浦工業大学大学院電気電子情報工学専攻

ロボティクスシステムデザイン研究室

土屋彩茜

Email:ma14076@shibaura-it.ac.jp