RTコンポーネント操作マニュアル

効率的な
RTシステム開発および運用のための
汎用ビューワコンポーネント

2022年9月6日版

芝浦工業大学

デザイン工学部デザイン工学科

佐々木 毅

更新履歴

2008 年 10 月 7 日版	第 1 版。
2008 年 11 月 11 日版	FAQ の記述の見直し。
2008 年 11 月 12 日版	動的入出力ポートを使用したサンプルとして動的入力ポートテストコンポーネント (DynamicInPortTest) を作成。それに伴い DynamicInPortTest に関する記述を追加。
2008 年 11 月 28 日版	3.2.1 節に “gpSeparatePos による区切り位置の指定” を追加。誤字や表現の一部を修正。
2008 年 12 月 7 日版	1.1 節の参考文献にページ数を追加。
2019 年 4 月 11 日版	OpenRTM-aist-1.2.0 対応による記述の修正。レーザレンジファインダコンポーネント、移動体トラッキングコンポーネントの記述を削除し、過去作品として記載。「GnuplotViewer の実用例」を参考とした。「動的入出力ポート」を独立のプロジェクトとしたため、動的入力ポートテストコンポーネント (DynamicInPortTest) の記述を削除。
2022 年 9 月 6 日版	6 章の問合せ先の変更。

目 次

1 本コンポーネント群の概要

1.1 開発の背景

RT コンポーネントによるシステム開発においては、それぞれのコンポーネントの動作確認が必要であり、また運用段階においても、センサデータやその処理結果などの表示、異常発生時の原因の早期発見などが重要となります。したがって、効率的な RT システムの開発および運用には、コンポーネントの出力を視覚化するビューワコンポーネントが有用であると考えられます。ビューワコンポーネントの開発にあたっては、様々なデータを様々な形式で表示できる汎用性や、使用法がわかりやすいことなどが望まれます。そこで、グラフ描画ツールとして広く用いられている gnuplot (gnuplot homepage: http://www.gnuplot.info/) を利用したビューワコンポーネントを開発することといたしました。

コンポーネントの設計指針等につきましては、

> 佐々木毅，橋本秀紀, “効率的な RT システム開発および運用のための汎用ビューワコンポーネント”, 第 9 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, pp.477–478, 2008.

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現バージョンについては Windows にて開発・動作確認を行っていますが、過去のバージョンでは Linux でも動作することを確認しています。開発環境は以下の通りです。

- OS: Windows 10 (64bit 版)
- RT ミドルウエア: OpenRTM-aist C++ v1.2.0
- コンパイラ: Microsoft Visual Studio 2017 Community (VC++ 2017)
- Python: Python 3.7
- CMake: CMake 3.14.1

1.3 開発したコンポーネント群

ビューワコンポーネントとそのツールとして、以下のコンポーネント群を開発しました。

- ビューワコンポーネント (GnuplotViewer)
  gnuplot を用いた汎用ビューワ。
- コンソール文字列入力コンポーネント (ConsoleInString)
  コンソールから入力した文字列を OutPort に出力する。

また、GnuplotViewer 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。

- 正弦 · 余弦関数出力コンポーネント (SinCosFunction)
  正弦波、余弦波を出力する。

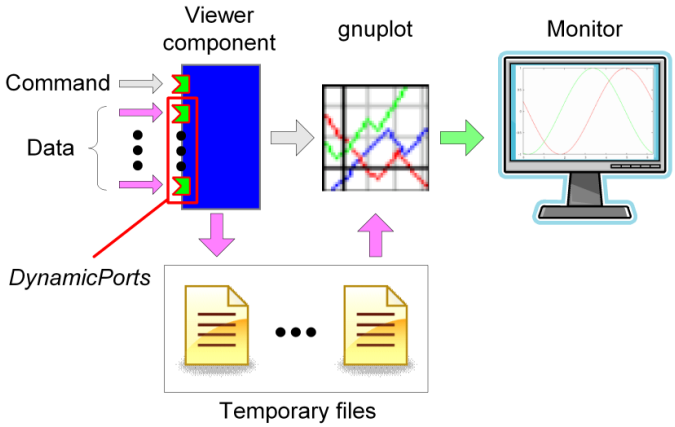
それぞれのコンポーネントの詳細については 2 章を、GnuplotViewer の使用方法については 3 章を参照してください。

2　各コンポーネントの説明

2.1　ビューワおよびそのツールコンポーネント

2.1.1　ビューワコンポーネント (GnuplotViewer)

GnuplotViewer は、gnuplot を用いた汎用ビューワです。そのため、このコンポーネントを使用するには gnuplot がインストールされている必要があります。コンポーネントの動作概要を下図に示します。GnuplotViewer の機能として、入出力ポートの動的追加 · 削除機能を実現しています。ビューワコンポーネントはまず、Configuration の p{Short, Long, Float, Double}InPortNum の値を設定してアクティブ化すると、対応する入力ポートがその数だけ作成されるとともに gnuplot を起動します。このとき、Configuration の gpOpenOption に gnuplot の起動オプションが与えられていれば、そのオプションにしたがって gnuplot をオープンします。アクティブ化後は他のコンポーネントから入力データを受け取ると、それを gnuplot が扱える形式で一時ファイルに出力します。さらに、コンポーネントに新しいデータが入力される度に Configuration の gpPeriodicExecCommand に記入されたコマンドを実行します。InPort から入力されたデータは、例えば 3 つ目の short 型ポートへのデータであれば “short[2]”、最初の long 型ポートへのデータであれば “long[0]” で参照できます。また InPort の Command から gnuplot へコマンドを送ることも可能です。詳しい使い方に関しては 3 章を参照してください。



- InPort

名称	型	説明
Command	TimedString	gnuplot に送るコマンド。
ShortSeqData	DynamicInPort<TimedShortSeq>	プロットする short 型整数値列データ。
LongSeqData	DynamicInPort<TimedLongSeq>	プロットする long 型整数値列データ。
FloatSeqData	DynamicInPort<TimedFloatSeq>	プロットする float 型実数値列データ。
DoubleSeqData	DynamicInPort<TimedDoubleSeq>	プロットする double 型実数値列データ。

- OutPort

  なし

- Configuration 変数

名称	型	デフォルト値	説明
gpOpenOption	string	*	gnuplot を開く際のオプション。アクティブ状態での変更は無効（反映されない）。
gpSeparatePos	string	1	データの区切り位置のリスト。Short, Long, Float, Double のポートの順に各ポートに対しコロン (:) を区切り文字として指定する。詳しくは 3.2.1 節 “gpSeparatePos による区切り位置の指定” を参照。指定されなかったポートの値は 1 となる。
gpPeriodicExecCommand	string	*	gnuplot が周期実行するコマンド。コンポーネントに新しいデータが入力される度に実行される。
pShortInPortNum	int	0	TimedShortSeq 型 InPort のポート数。アクティブ状態での変更は無効（反映されない）。
pLongInPortNum	int	0	TimedLongSeq 型 InPort のポート数。アクティブ状態での変更は無効（反映されない）。
pFloatInPortNum	int	0	TimedFloatSeq 型 InPort のポート数。アクティブ状態での変更は無効（反映されない）。
pDoubleInPortNum	int	0	TimedDoubleSeq 型 InPort のポート数。アクティブ状態での変更は無効（反映されない）。

- サービスポート

  なし

2.1.2 コンソール文字列入力コンポーネント (ConsoleInString)

ConsoleInString は、OpenRTM-aist のホームページ (http://www.is.aist.go.jp/rt/OpenRTM-aist/) にサンプルとして取り上げられている ConsoleIn コンポーネントの出力の型を TimedString としたものです。アクティブ化すると入力を促すメッセージがコンソールに表示されます。コンソールから入力を行うと、その文字列が OutPort に出力されます。

- InPort

  なし

- OutPort

名称	型	説明
OutString	TimedString	コンソールから入力された文字列。

- Configuration 変数

  なし

- サービスポート

  なし

2.2 GnuplotViewer の使用手順の説明で用いるコンポーネント

2.2.1 正弦 · 余弦関数出力コンポーネント (SinCosFunction)

SinCosFunction は、正弦波、余弦波を出力するコンポーネントです。アクティブ化を行うと、$t = 0$ に初期化され、$x, \sin(k_s(x-c_st)), \cos(k_c(x-c_ct))$ $(0 \leq x < 2\pi)$ のデータが 1 次元配列として OutPort に出力されます。出力形式は、xData[0], sinData[0], cosData[0], $\cdots$, xData[SampleNum$-$1], sinData[SampleNum$-$1], cosData[SampleNum$-$1] の順です。つまり、$0, \sin(k_s(0-c_st)), \cos(k_c(0-c_ct)), \cdots, 2\pi\dfrac{\mathrm{SampleNum}-1}{\mathrm{SampleNum}}, \sin\left(k_s\left(2\pi\dfrac{\mathrm{SampleNum}-1}{\mathrm{SampleNum}} - c_st\right)\right)$, $\cos\left(k_c\left(2\pi\dfrac{\mathrm{SampleNum}-1}{\mathrm{SampleNum}} - c_ct\right)\right)$ です。

- InPort

  なし

- OutPort

名称	型	説明
SinCosData	TimedDoubleSeq	$x, \sin(k_s(x-c_st)), \cos(k_c(x-c_ct))$ の値。配列長は 3×SampleNum。

- Configuration 変数

名称	型	デフォルト値	説明
sWaveNum	double	1.0	正弦波の波数 $k_s(=2\pi/\lambda_s)$。単位は rad/m。
sPhaseVel	double	1.0	正弦波の位相速度 $c_s(=\omega_s/k_s)$。単位は m/s。
cWaveNum	double	1.0	余弦波の波数 $k_c(=2\pi/\lambda_c)$。単位は rad/m。
cPhaseVel	double	1.0	余弦波の位相速度 $c_c(=\omega_c/k_c)$。単位は m/s。
SampleNum	int	200	データ数。[0,2π) の範囲をこのデータ数でサンプルする。

- サービスポート

  なし

3 GnuplotViewer の使用方法

3.1 RTSystemEditor によるシステム構築の手順

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

RTSystemEditor を用いたシステム構築の手順

1. ネームサーバの起動
2. RTSystemEditor の起動
3. ネームサーバへの接続
4. コンポーネントの起動
5. システムの構築と実行

これらの手順はどのようなシステムでも共通ですので、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 GnuplotViewer の使用手順

GnuplotViewer の使用手順をまとめると以下のようになります。

GnuplotViewer の使用手順

1. Configuration を設定する
   (a) p{Short, Long, Float, Double}InPortNum から使用するポートの数を設定する
   (b) gpSeparatePos から 1 次元ベクトルデータの区切り位置を設定する
   (c) gpPeriodicExecCommand からプロットコマンドを設定する
2. コンポーネントをアクティブ化する
3. 出力を表示したいコンポーネントと接続する
4. 適宜 Command ポートからプロット設定などのコマンドを入力する

手順 1(b) および 1(c) は手順 2 のアクティブ化の後に行うことも可能です。以下の節では具体的な例を挙げながら GnuplotViewer の使い方を説明します。

3.2.1 基本的な使い方 – 1 次元配列データのプロット機能の利用 –

まず、GnuplotViewer の最も基本的な使用方法である 1 次元配列データのプロット機能について説明します。使用するコンポーネントの種類とその数は

- ビューワコンポーネント (GnuplotViewer) · · · · · · 1 つ
- コンソール文字列入力コンポーネント (ConsoleInString) · · · · · · 1 つ
- 正弦 · 余弦関数出力コンポーネント (SinCosFunction) · · · · · · 1 つ

の計 3 つです。以下に使用手順を示します。

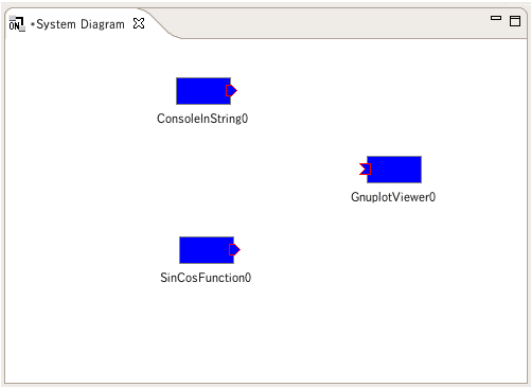
(1) コンポーネントの配置

予め以下のような設定ファイルを “rtc.conf” として SinCosFunction の実行フォルダに保存しておきます。既に rtc.conf がある場合には以下の内容を rtc.conf 内に追記してください。

rtc.conf

```
exec_cxt.periodic.rate: 5.0
```

次にコンポーネントを起動し、システムエディタ上に GnuplotViewer, ConsoleInString, SinCosFunction を配置してください。



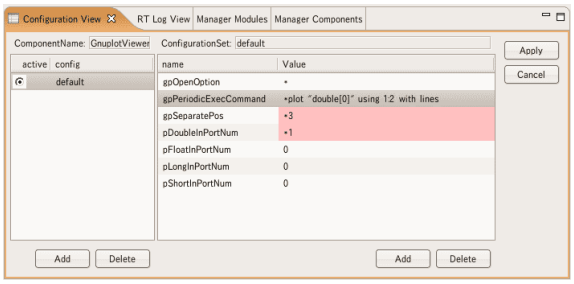
(2) Configuration 変数の設定

GnuplotViewer を選択し、ConfigurationView からパラメータを設定します。ここでは、SinCosFunction の OutPort から出力されるデータのうち、x の値と sin() の値をプロットすることを考えます。

まず、使用するポートの種類と数の設定を行います。SinCosFunction の OutPort は TimedDoubleSeq 型なので、pDoubleInPortNum の値を 1 にします。次に、1 次元データ配列の区切り位置を設定します。SinCosFunction の OutPort は xData[0], sinData[0], cosData[0], xData[1], sinData[1], cosData[1], $\cdots$ というように 3 つごとに同種のデータを出力するので gpSeparatePos の値を 3 にします。なお、gpSeparatePos の指定についての詳細は次頁の囲み枠内 “gpSeparatePos による区切り位置の指定” を参照してください。続いて gnuplot が周期実行するコマンドを指定します。ここでは、唯一の（0 番目の）double 型のポートの 1 列目 (x) と 2 列目 (sin()) の値をプロットするので、

plot ”double[0]” using 1:2 with lines

というコマンドを指定します（上記のコマンドは p ”double[0]” u 1:2 w l などと省略することも可能です）。また、gpOpenOption は gnuplot を開く際のオプションですが、ここでは特に指定する必要はないと思います。なお、ここではすべての設定をアクティブ化前に行いましたが、前述の通り gpSeparatePos と gpPeriodicExecCommand はアクティブ状態にしてからの設定も可能です。設定が終了したら変更を確定するために Apply をクリックします。



gpSeparatePos による区切り位置の指定

GnuplotViewer の Configuration 変数である gpSeparatePos は、1 次元配列データの区切り位置を指定し、2 次元プロットや 3 次元プロットを簡単に行えるようにするための値です。例えば、元の 1 次元配列データが 1 2 3 4 5 6 7 8 9 10 であるとき、gpSeparatePos の値が 3 であれば、このデータは

1 列目	2 列目	3 列目
1	2	3
4	5	6
7	8	9
10		

のように整列しているものと見なされます。これにより、gnuplot の using オプションで例えば 2:3 のように指定すれば、2 列目と 3 列目のデータの 2 次元プロットなどが可能となります。

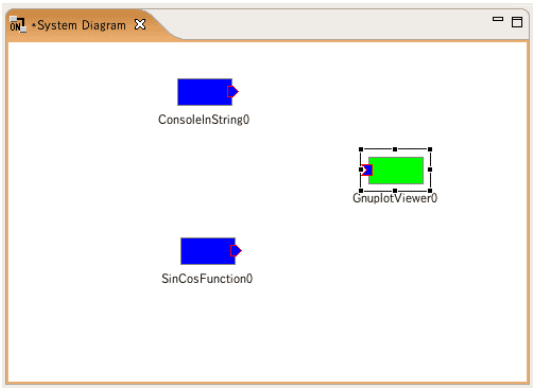
また、複数ポートがある場合には、Short, Long, Float, Double のポートの順にコロン (:) を区切り文字としてそれぞれの区切り位置を指定します。例えば、ShortSeq 型のポートが 2 つ、FloatSeq 型のポートが 1 つあり、それぞれ区切り位置を 1,2,3 としたいときには 1:2:3 と入力してください。

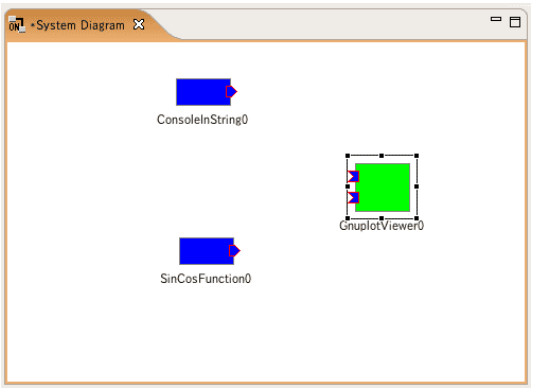
(3) コンポーネントのアクティブ化

GnuplotViewer を選択し、右クリックメニューの Activate をクリックしてコンポーネントをアクティブ化します。GnuplotViewer のコンソールに

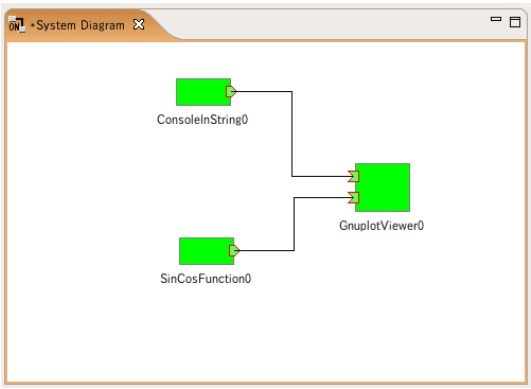
Set separate position list.
3
Set command: plot ”double[0]” using 1:2 with lines

と表示されれば gnuplot のオープンとプロット設定の初期化は完了です*。アクティブ化すると追加したポートが現れます。





ConsoleInString の OutString ポートと GnuplotViewer の Command ポート、SinCosFunction の SinCosData ポートと GnuplotViewer の DoubleSeqData ポートをそれぞれ接続し、ConsoleInString と SinCosFunction もアクティブ化します。

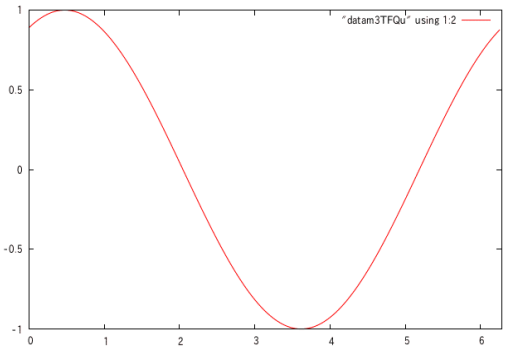


すると、SinCosFunction からデータが送られてきたことによって GnuplotViewer の gpPeriodicExecCommand が実行され、gnuplot のウインドウ内に x 軸方向に移動する正弦波が表示されます。なお、ConsoleInString のコンソールから

set xrange [0:2*pi]
set yrange [-1:1]

と入力して x 軸、y 軸の表示範囲を変更するとより見やすくなります。

*エラーが発生する場合には 5 章の内容をご確認ください。



なお、ConsoleInString から送るコマンドが長い場合や、gpPeriodicExecCommand に設定したいコマンドが複数行にわたる場合などは、そのコマンドを予めファイルに保存しておき、gnuplot の load コマンドや call コマンド（こちらだと引数を与えられる）によって呼び出しを行うと便利です。

この他にも、ConsoleInString から送るコマンドや各コンポーネントの Configuration の設定を変更することで様々な表示が可能です。以下に例を示します。

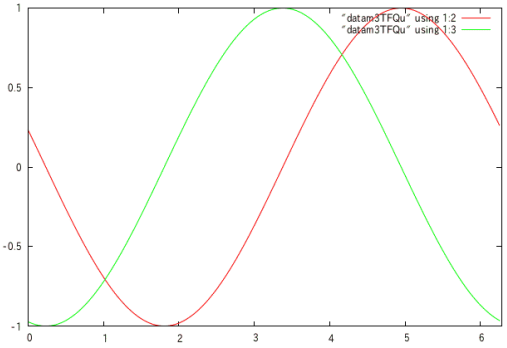
2 つの系列を同時に表示（第 2 系列と第 3 系列を同じグラフ上に表示 $y = \sin(x), y = \cos(x)$）

- ConsoleInString から送るコマンド

  set xrange [0:2*pi]
  set yrange [-1:1]

- GnuplotViewer の gpPeriodicExecCommand

  p ”double[0]” u 1:2 w l, ”double[0]” u 1:3 w l



演算子の適用、系列同士の演算（第 2 系列と第 3 系列の積を 2 倍して表示 $y = 2\sin(x)\cos(x) = \sin(2x)$）

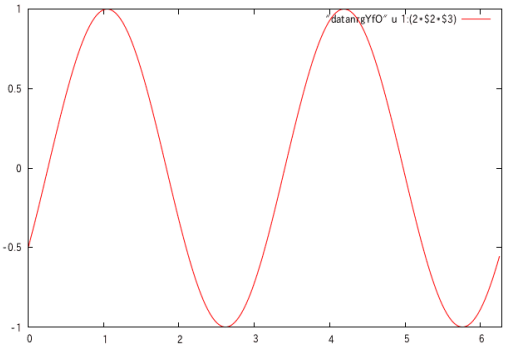
- ConsoleInString から送るコマンド

  set xrange [0:2*pi]
  set yrange [-1:1]

- GnuplotViewer の gpPeriodicExecCommand

  p ”double[0]” u 1:(2*$2*$3) w l

  * この他にも対数 (log) や正の平方根 (sqrt) などが使用できます。



極座標プロット（第 2 系列を極座標表示 $r = \sin(5t)$）

- ConsoleInString から送るコマンド

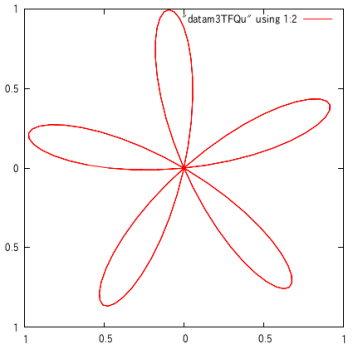
  set size square
  set xrange [-1:1]
  set yrange [-1:1]
  set polar

- SinCosFunction の Configuration

  sWaveNum 5

- GnuplotViewer の gpPeriodicExecCommand

  p ”double[0]” u 1:2 w l



3.2.2 発展的な使い方 – テキスト処理ツールの利用 –

テキスト処理を行える Unix コマンドや、sed、AWK などのツールを用いることで、1 次元配列データのプロット機能だけでは困難な様々なプロット機能を実現できます。ここではその例として、2 つの異なるコンポーネントから得られた出力から 1 系列ずつをそれぞれ x,y の値として 2 次元プロットを行います。使用するコンポーネントの種類とその数は

- ビューワコンポーネント (GnuplotViewer) · · · · · · 1 つ
- コンソール文字列入力コンポーネント (ConsoleInString) · · · · · · 1 つ
- 正弦 · 余弦関数出力コンポーネント (SinCosFunction) · · · · · · 2 つ

の計 4 つです。以下に使用手順を示します。

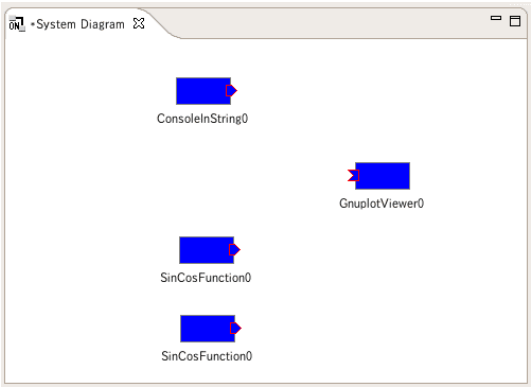
(1) コンポーネントの配置

予め以下のような設定ファイルを “rtc.conf” として SinCosFunction の実行フォルダに保存しておきます。既に rtc.conf がある場合には以下の内容を rtc.conf 内に追記してください。

rtc.conf

```
exec_cxt.periodic.rate: 5.0
```

次にコンポーネントを起動し、システムエディタ上に GnuplotViewer, ConsoleInString, SinCosFunction を配置してください。



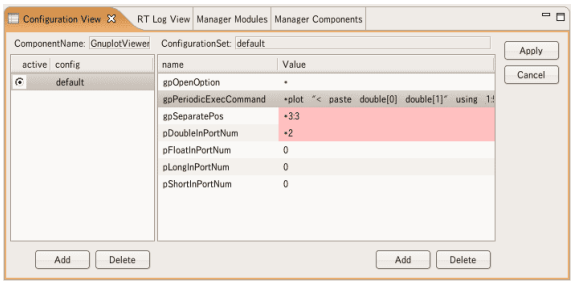
(2) Configuration 変数の設定

GnuplotViewer を選択し、ConfigurationView からパラメータを設定します。ここでは、Unix コマンドの paste を用いて 2 つの異なるポートから入力されたデータを結合し、1 つめの SinCosFunction の OutPort から出力されるデータの x の値と、2 つめの SinCosFunction の OutPort から出力されるデータの sin() の値をプロットすることを考えます。

まず、使用するポートの種類と数の設定を行います。SinCosFunction の OutPort は TimedDoubleSeq 型なので、pDoubleInPortNum の値を 2 にします。次に、1 次元データ配列の区切り位置を設定します。SinCosFunction の OutPort は xData[0], sinData[0], cosData[0], xData[1], sinData[1], cosData[1], $\cdots$ というように 3 つごとに同種のデータを出力し、それが 2 つあるので gpSeparatePos の値を 3:3 にします。続いて gnuplot が周期実行するコマンドを指定します。ここでは、2 つの（0 番目と 1 番目の）double 型のポートのそれぞれ 1 列目 (x) と 2 列目 (sin()) の値を paste コマンドにより結合してプロットするので、gpPeriodicExecCommand に指定するコマンドは以下のようになります。

plot ”< paste double[0] double[1]” using 1:5 with lines

ここで、using の設定が 1:5 なのは、結合を行うと 1 つめのコンポーネントの 3 列が挿入されることにより 2 つめのコンポーネントの 2 列目が $3+2=5$ 列目になるためです。また、gpOpenOption は gnuplot を開く際のオプションですが、ここでは特に指定する必要はないと思います。なお、ここではすべての設定をアクティブ化前に行いましたが、前述の通り gpSeparatePos と gpPeriodicExecCommand はアクティブ状態にしてからの設定も可能です。設定が終了したら変更を確定するために Apply をクリックします。

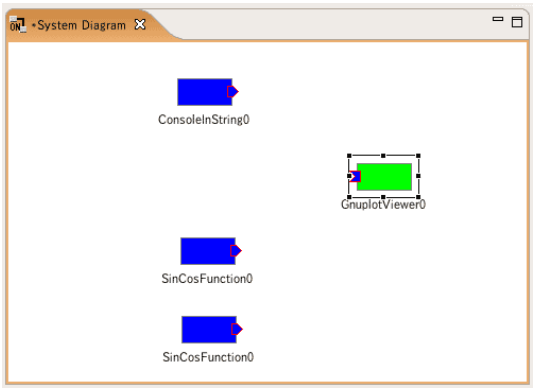


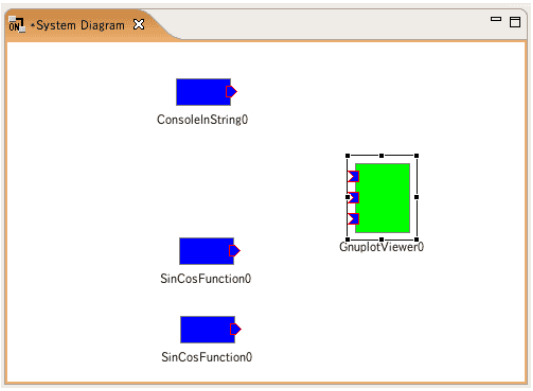
(3) コンポーネントのアクティブ化

GnuplotViewer を選択し、右クリックメニューの Activate をクリックしてコンポーネントをアクティブ化します。GnuplotViewer のコンソールに

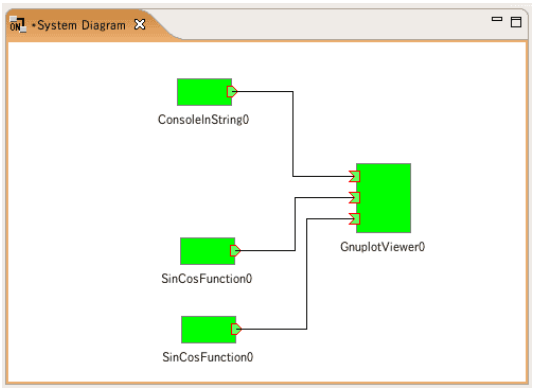
Set separate position list.
3:3
Set command: plot ”< paste double[0] double[1]” using 1:5 with lines

と表示されれば gnuplot のオープンとプロット設定の初期化は完了です。アクティブ化すると追加したポートが現れます。





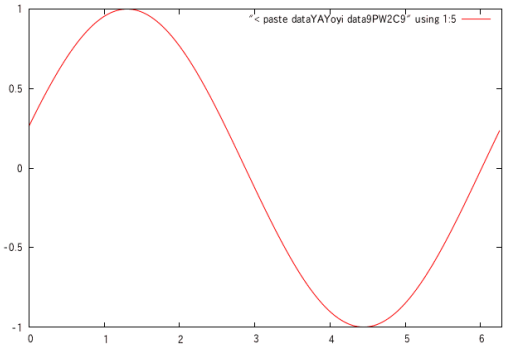
ConsoleInString の OutString ポートと GnuplotViewer の Command ポート、SinCosFunction の SinCosData ポートと GnuplotViewer の DoubleSeqData ポートをそれぞれ接続し、ConsoleInString と 2 つの SinCosFunction もアクティブ化します。



すると、2 つの SinCosFunction からデータが送られてきたことによって GnuplotViewer の gpPeriodicExecCommand が実行され、前節と同様 gnuplot のウインドウ内に x 軸方向に移動する正弦波が表示されます。なお、ConsoleInString のコンソールから

```
set xrange [0:2*pi]
set yrange [-1:1]
```

と入力して x 軸、y 軸の表示範囲を変更するとより見やすくなります。



4 (参考) GnuplotViewer の実用例

本章では、北陽電機株式会社製のレーザレンジファインダ URG-04LX を用いた人間トラッキングを取り上げ、GnuplotViewer の実用例を示します。

4.1 本章で利用しているコンポーネント群

ここでは、過去の RT ミドルウエアコンテストに出品された以下のコンポーネントを利用しています。

佐々木毅, 橋本秀紀, “効率的な RT システム開発および運用のための汎用ビューワコンポーネント”, RT ミドルウエアコンテスト 2008 応募作品 1L3-2 より

- レーザレンジファインダコンポーネント (LRFComponent)

  北陽電機株式会社製のレーザレンジファインダ URG-04LX を RT コンポーネント化したもの。

- 移動体トラッキングコンポーネント (SimpleTracker)

  レーザレンジファインダのスキャンデータから移動物体の位置を出力する。

これらのコンポーネントを含め、使用するコンポーネントの種類とその数は

- ビューワコンポーネント (GnuplotViewer) · · · · · · 1 つ
- コンソール文字列入力コンポーネント (ConsoleInString) · · · · · · 1 つ
- レーザレンジファインダコンポーネント (LRFComponent) · · · · · · 1 つ
- 移動体トラッキングコンポーネント (SimpleTracker) · · · · · · 1 つ

の計 4 つです。

まず LRFComponent が出力する距離データをプロットする例を示し、続いて SimpleTracker が出力する位置データをプロットする例を示します。

4.2 距離データの表示

(0) 設定ファイルの作成

予め、以下の 2 つの設定ファイルを作成し、それぞれ、“plot_lrfdata.gp”, “addindex_lrf.awk” として保存しておきます。

plot_lrf.data.gp

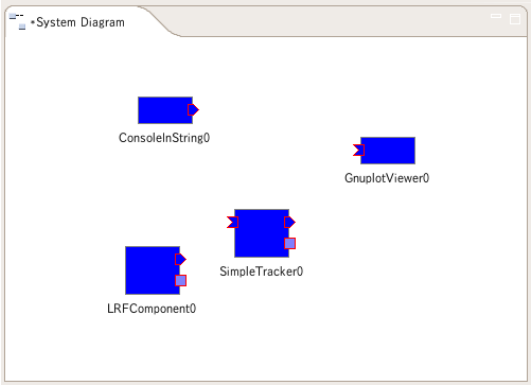
```
set polar
set angles degrees
plot ”<awk -f addindex_lrf.awk $0” u 1:($$2/1000) w p
set angles radians
set nopolar
```

addindex_lrf.awk

```
BEGIN{
  stp = 0
  ar = 270/768 #angular resolution of URG-04LX [deg]
}

NR==1{
  st = -135 + $1*ar #start angle [deg]
}

NR==2{
  ed = -135 + $1*ar #end angle [deg]
}

NR>2{
  line[NR-2]=$0 #store data
}

END{
  if(NR>3){
    stp=(ed-st)/(NR-2-1)
  }
  angle = st
  for(i=0;i<NR-2;i++){
    print angle,line[i] #output index + range data
    angle += stp #update index
  }
}
```

(1) コンポーネントの配置

コンポーネントを起動し、システムエディタ上に GnuplotViewer, ConsoleInString, LRFComponent, SimpleTracker を配置してください。



(2) Configuration 変数の設定

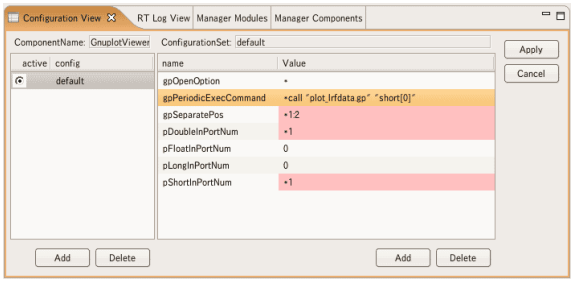
GnuplotViewer を選択し、ConfigurationView からパラメータを設定します。

まず、使用するポートの種類と数の設定を行います。ここでは、TimedShortSeq 型の OutPort をもつ LRFCom-

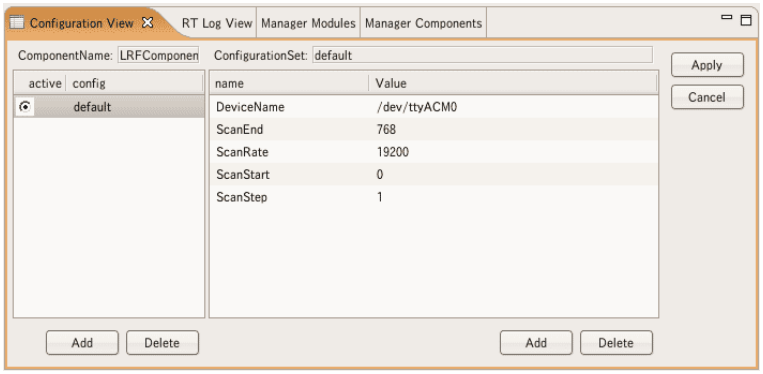
ponent と TimedDoubleSeq 型の OutPort をもつ SimpleTracker の出力データを表示させるため、pShortInPortNum と pDoubleInPortNum の値を 1 にします。次に、1 次元データ配列の区切り位置を設定します。LRFComponent の OutPort は距離データのみを出力し、SimpleTracker の出力は xData[0], yData[0], xData[1], yData[1], · · · というように 2 つごとに同種のデータを出力するので gpSeparatePos の値を 1:2 にします。続いて gnuplot が周期実行するコマンドを指定します。前述のとおり最初は LRFComponent が出力する距離データをプロットしますので、用意したプロット設定ファイルを call します。gpPeriodicExecCommand に以下のコマンドを指定してください。

call ”plot_lrfdata.gp” ”short[0]”

また、gpOpenOption は gnuplot を開く際のオプションですが、ここでは特に指定する必要はないと思います。なお、ここではすべての設定をアクティブ化前に行いましたが、前述の通り gpSeparatePos と gpPeriodicExecCommand はアクティブ状態にしてからの設定も可能です。設定が終了したら変更を確定するために Apply をクリックします。



さらに、LRFComponent を選択し、ConfigurationView から測定パラメータ等を設定してください。通常はデフォルトのままでよいと思いますが、必要に応じて DeviceName を変更してください。

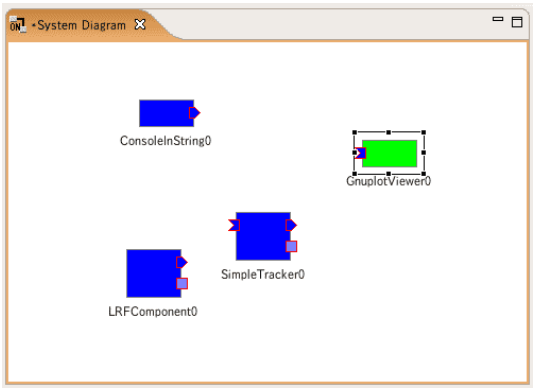


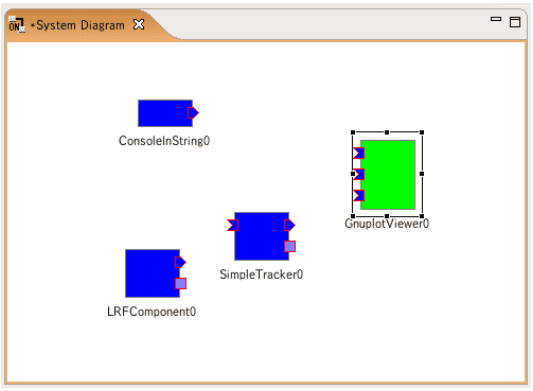
(3) コンポーネントのアクティブ化

GnuplotViewer を選択し、右クリックメニューの Activate をクリックしてコンポーネントをアクティブ化します。GnuplotViewer のコンソールに

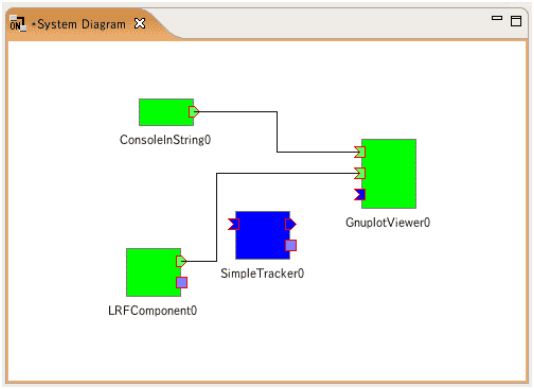
Set separate position list.
1:2
Set command: call ”plot_lrfdata.gp” ”short[0]”

と表示されれば gnuplot のオープンとプロット設定の初期化は完了です。アクティブ化すると追加したポートが現れます。

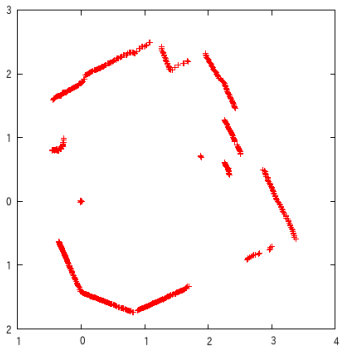




ConsoleInString の OutString ポートと GnuplotViewer の Command ポート、LRFComponent の ScanData ポートと GnuplotViewer の ShortSeqData ポートをそれぞれ接続し、ConsoleInString と LRFComponent もアクティブ化します。



すると、LRFComponent からデータが送られてきたことによって GnuplotViewer の gpPeriodicExecCommand が実行され、gnuplot のウインドウ内にスキャン結果が表示されます。なお、ConsoleInString のコンソールから x 軸、y 軸の表示範囲を変更するとより見やすくなります。



4.3 位置データの表示

前節に続き、今度は SimpleTracker が出力する位置データをプロットします。

(0) 設定ファイルの作成

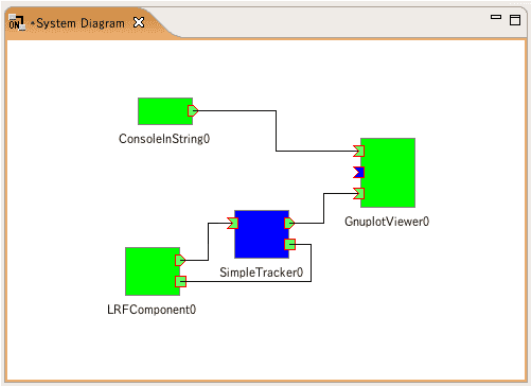
予め、以下の設定ファイルを作成し、“tracking_lrf.conf” として保存しておきます。

tracking_lrf.conf

```
reset
set nokey
set lmargin 5
set bmargin 2
set xrange [-1:4]
set yrange [-2:3]
set xtics 1
set ytics 1
set size ratio -1
set multiplot
plot ”-” u 1:2 w p lt rgb ”blue” pt 5 ps 2
0 0
end
set border 0
set format x ””
set format y ””
set noxtics
set noytics
set xtics nomirror
set ytics nomirror
```

(1) コンポーネントの接続の変更

LRFComponent の ScanData ポートの接続を SimpleTracker の ScanData ポートに変更し、またこれらのコンポーネントのサービスポートを接続します。さらに、SimpleTracker の Position2D ポートと GnuplotViewer の DoubleSeqData ポートを接続します。

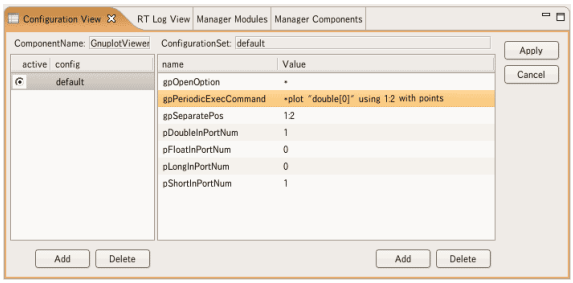


(2) Configuration 変数の設定変更

GnuplotViewer を選択し、ConfigurationView から gnuplot が周期実行するコマンドを変更します。gpPeriodicExecCommand に以下のコマンドを指定してください。

plot ”double[0]” using 1:2 with points

設定が終了したら変更を確定するために Apply をクリックします。



GnuplotViewer のコンソールに

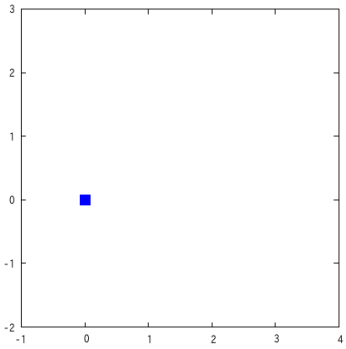
Set command: plot ”double[0]” using 1:2 with points

と表示されれば gnuplot の周期実行コマンドの変更は完了です。

さらに、ここではトラッキング結果の履歴を点列として表示させることを考えます。ConsoleInString から以下のコマンドを送信し、用意した設定ファイルのロードを行います。

load ”tracking_lrf.conf”

GnuplotViewer のコンソールにも同様のコマンドが表示されれば送信は完了です。するとレーザの位置 (0,0) を示したグラフが表示されます。なお、プロット範囲等は先ほどの距離データの表示結果を基に tracking_lrf.conf の内容を編集し、変更を行っておくとより見やすくなります。編集を行ったらファイルを保存後もう一度ロードを行ってください。



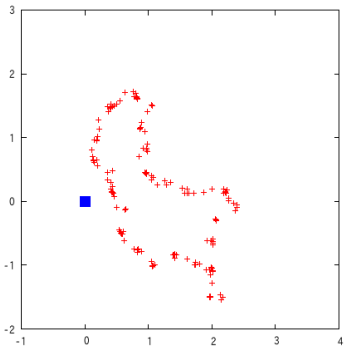
(3) コンポーネントのアクティブ化

SimpleTracker を選択し、右クリックメニューの Activate をクリックしてコンポーネントをアクティブ化します。SimpleTracker は、最初に背景情報（静止物体領域の情報）を獲得するので、アクティブ化を行うときは観測領域に人間などの移動物体が入らないように注意してください。SimpleTracker のコンソールに

Learning background...done

と表示されれば背景情報の獲得は完了です。

この状態で移動物体がレーザレンジファインダの観測領域内を移動すると gnuplot のウインドウ内にトラッキング結果が表示されます。表示をリセットしたい場合は、もう一度 ConsoleInString から tracking_lrf.conf のロードを行ってください。



5 FAQ

Q1: GnuplotViewer をアクティブ化しようとすると Error in Gnuplot::initialize(): Cannot open gnuplot と表示され、エラー状態になりアクティブ化できない。

Ans.: これには主に 2 つの原因が考えられます。

1. gnuplot がインストールされていない
2. gnuplot へのパスが通っていない

まずは gnuplot がインストールされているかを確認してください。次に、カレントフォルダで gnuplot と入力し、gnuplot が起動できるかどうか確認してください。gnuplot がインストールされているにもかかわらず gnuplot が起動できない場合には、環境変数 PATH に gnuplot の実行ファイルのパスを指定してください。

Q2: GnuplotViewer をアクティブ化しようとすると Error in Gnuplot::addData(): Cannot open file と表示され、エラー状態になりアクティブ化できない。

Ans.: GnuplotViewer が利用する一時ファイルが、そのフォルダへの書き込み許可がない等の理由により作成できないことが原因であると考えられます。デフォルトではカレントフォルダが一時ファイルの保存場所となっていますので、フォルダのパーミッションの変更を行ってください。デフォルトの一時ファイルの保存場所を変更したいという場合には、ソースファイルからこの設定を変更します。GnuplotViewer.h の 396 行目の Gnuplot クラスの変数宣言部分で一時ファイルの保存場所を指定してください。Gnuplot クラスのコンストラクタの引数が一時ファイルの保存場所です。

〈例〉

（変更前）Gnuplot gp;

（変更後）Gnuplot gp(/tmp/);

このとき、最後の/までを含めて指定してください。変更後はファイルを保存しビルドを行ってください。

Q3: Sequence 型データ以外の型や独自のデータ型をもったデータをプロットしたい。

Ans.: 可視化を行うデータは通常、単一の数値よりも数値列であると考えられるため、現状では効率の面から整数および実数値の可変長 1 次元配列データ型である TimedShortSeq, TimedLongSeq, TimedFloatSeq, TimedDoubleSeq のみをサポートしています。要望がありましたら他の型へのサポートも行いますが、現状ではこのような場合にはデータ変換用の別のコンポーネントを用意する必要があります。

6　お問い合わせ

本コンポーネントにつきましては、まだ改善の余地があるものと考えております。ご要望、バグ報告、マニュアルの記述の不備等に関しましては、芝浦工業大学デザイン工学部デザイン工学科の佐々木までご連絡ください。

問合せ先

〒 135-8548
東京都江東区豊洲 3-7-5
芝浦工業大学 本部棟 6 階 06K10 佐々木研究室
Tel:03-5859-8834
sasaki-t at ieee. org