RTコンポーネント操作マニュアル

効率的な入力データ生成のための
ファンクションジェネレータコンポーネント

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

2009 年 11 月 15 日版	第 1 版。
2009 年 11 月 20 日版	誤字等の修正。
2009 年 12 月 4 日版	新たに 1.2 節としてファンクションジェネレータコンポーネントの用途例を追加。 ファンクションジェネレータコンポーネント Ver.1.0.2 で利用できる関数に天井関数 ceil と床関数 floor が追加されたことに伴い、これらの記述を 3.2.1 節に追加。 4 章「FunctionGenerator の利用例」を新たに追加し、4.2 節としてビューワコンポーネントを用いた出力データの可視化を紹介。 その他、一部記述の修正。
2009 年 12 月 7 日版	4.3 節として音声入出力コンポーネントを用いた音によるデータの提示を紹介。 その他、誤字、脱字や表現の一部を修正。
2009 年 12 月 18 日版	4.1 節として本稿で利用しているコンポーネント群の一覧を掲載。 ファンクションジェネレータコンポーネントの利用例を示すものとしてパン-チルト-ズームカメラコンポーネント (PTZCameraSony) を作成。それに伴い、パン-チルト-ズームカメラコンポーネントの詳細を追加。 4.4 節としてパン-チルト-ズームカメラコンポーネントを用いた広範囲観測を紹介。 その他、誤りの修正。
2009 年 12 月 27 日版	1.1 節の参考文献にページ数を追加。
2011 年 9 月 18 日版	OpenRTM-aist-1.0.0 対応により 1.3 節と 4.1 節の記述を修正。 著者所属変更に伴う 5 章の問合せ先の変更。
2019 年 4 月 11 日版	OpenRTM-aist-1.2.0 対応による記述の修正。1.0 対応版と 0.4 対応版の記載の削除。 パン-チルト-ズームカメラコンポーネントの記述を削除し、過去作品として記載。 「FunctionGenerator の利用例」を参考とした。
2022 年 9 月 6 日版	5 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

コンポーネント指向による RT システムの開発や運用に際しては、開発したコンポーネントやそれらを統合したシステムが様々な入力に対して正常に動作するかを確認するための動作テストや、移動ロボットの経路やロボットアームの目標姿勢といったアクチュエータへの指令の生成などが必要であり、効率的に入力データを作成することが望まれます。このような背景から、ユーザが様々なパターンの入力データを生成することが可能なファンクションジェネレータコンポーネントを開発することといたしました。

- ファンクションジェネレータコンポーネント (FunctionGenerator)

  入力データ生成コンポーネント。

コンポーネントの詳細については 2 章を FunctionGenerator の使用方法については 3 章を参照してください。
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, “効率的な入力データ生成のためのファンクションジェネレータコンポーネント”, 第 10 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.29–30, 2009.

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 ファンクションジェネレータコンポーネントの用途

ファンクションジェネレータコンポーネントは、上述のようにコンポーネントの開発及び運用段階において様々に利用することができます。代表的な用途としては以下のようなものが挙げられます。

- 様々な入力データパターンに対するコンポーネントやシステムの動作テスト
- 情報処理コンポーネントの処理パラメータの影響の評価
- 複数の情報処理コンポーネントの同一の入力条件での性能比較
- アクチュエータなどへの入力データの生成

1.3 開発環境

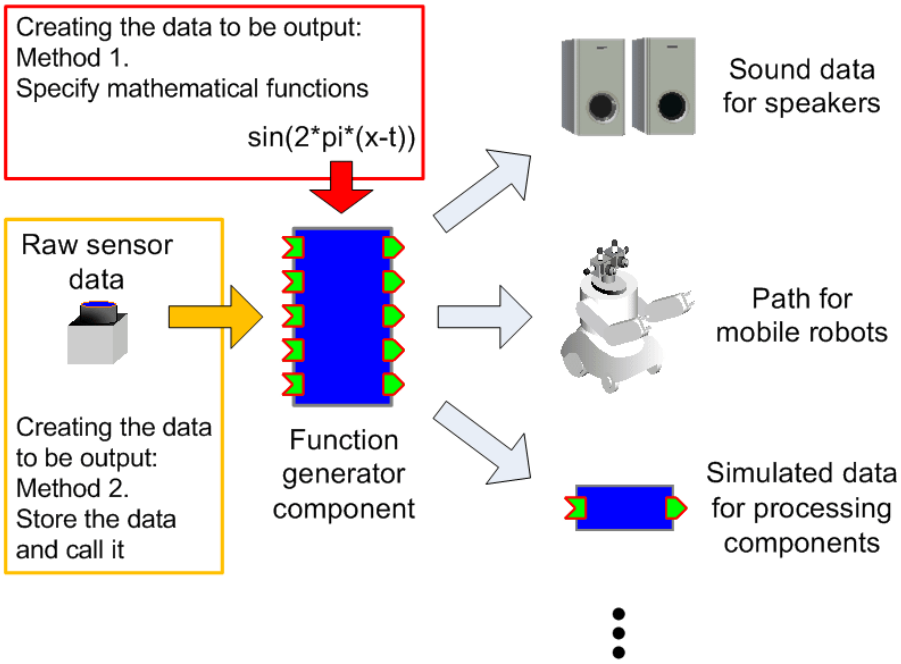
ファンクションジェネレータコンポーネントは現バージョンについては Windows にて開発・動作確認を行っていますが、過去のバージョンでは Linux でも動作することを確認しています。開発環境は以下の通りです。

- 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

2　コンポーネントの説明

2.1　ファンクションジェネレータコンポーネント (FunctionGenerator)

FunctionGenerator は、数学関数や事前に保存したデータ列を指定することで様々なパターンの入力データを生成することが可能な入力データ生成コンポーネントです。コンポーネントの動作概要を下図に示します。出力データの生成方法は大きく分けて 2 通りあります。まず、コンフィギュレーションパラメータから出力したいデータを数学関数によって指定することが可能です。また、ファンクションジェネレータコンポーネントには InPort に入力されたデータを記録する機能も備えており、予めセンサ出力などを保存しておき、そのデータを読み込み出力することも可能です。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
iOctetSeqData	TimedOctetSeq	ファイルに保存する octet 型数値列データ。
iShortSeqData	TimedShortSeq	ファイルに保存する short 型整数値列データ。
iLongSeqData	TimedLongSeq	ファイルに保存する long 型整数値列データ。
iFloatSeqData	TimedFloatSeq	ファイルに保存する float 型実数値列データ。
iDoubleSeqData	TimedDoubleSeq	ファイルに保存する double 型実数値列データ。

- OutPort

名称	型	説明
oOctetSeqData	TimedOctetSeq	octet 型数値列の出力データ。
oShortSeqData	TimedShortSeq	short 型整数値列の出力データ。
oLongSeqData	TimedLongSeq	long 型整数値列の出力データ。
oFloatSeqData	TimedFloatSeq	float 型実数値列の出力データ。
oDoubleSeqData	TimedDoubleSeq	double 型実数値列の出力データ。

- Configuration 変数

名称	型	デフォルト値	説明
xMin	double	0.0	変数 x の最小値。
xMax	double	10.0	変数 x の最大値。
xDelta	double	1.0	変数 x の増加量。
tMin	double	0.0	変数 t の最小値。
tMax	double	10.0	変数 t の最大値。
tDelta	double	1.0	変数 t の増加量。
saveFileMode	char	a	入力データを保存するときのモード。w:アクティブ化時またはアクティブ化後の saveFileMode 再設定時に既存のファイルを上書き。a:既存のファイルに追記。
saveFileName	string	*	入力データを保存するファイル名。*とした場合、何も保存しない。
outputFunctionOrLoadFileName	string	*	出力する関数または読み込むファイル名。*とした場合、何も出力しない。

- サービスポート

  なし

3 FunctionGenerator 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、FunctionGenerator の使用手順を説明します。

3.2 FunctionGenerator の使用手順

FunctionGenerator には大きく分けて 2 つの使い方があります。

1. 数学関数を出力する
2. データをファイルに保存し、そのデータを読み込んで出力する

いずれの場合においても、FunctionGenerator は指定されたデータを順次 OutPort から出力します。このときのコンポーネントの実行周期は rtc.conf の exec_cxt.periodic.rate の値で設定することができます。以下の節では、これらのそれぞれの使い方について説明します。

3.2.1 基本的な使い方 (1) – 数学関数出力機能の利用 –

まず、数学関数を入力し、それを出力する機能について説明します。この使用手順をまとめると以下のようになります。

FunctionGenerator の使用手順 – 数学関数出力機能の利用 –

1. Configuration を設定する
   (a) 変数のパラメータ xMin, xMax, xDelta, tMin, tMax, tDelta を設定する
   (b) outputFunctionOrLoadFileName に出力する関数を設定する
2. OutPort をデータを与えたいコンポーネントと接続する
3. コンポーネントをアクティブ化する

手順 1, 2, 3 は前後しても構いません。ただし、手順 1 で関数を設定し、手順 3 のアクティブ化を行うとデータ出力が始まりますので、コンポーネントの接続が行われていないと相手のコンポーネントがその部分のデータを受け取れなくなってしまいます。したがって、コンポーネントの周期実行のたびに値が変化する変数 t（以下で説明）を利用する場合には注意が必要です。

また、出力されるデータはそれぞれの出力ポートからポートのデータ形式に従って出力されます。例えば、oShortSeqData からは short 型の整数値列が出力されますので、小数点以下の値は切り捨てられます。また、変数で表現できる値の最大値より大きい値や最小値より小さい値に対しては、それぞれその最大値、最小値が出力されます。

以下では Configuration の設定と出力されるデータについて説明します。

(a) 変数のパラメータ xMin, xMax, xDelta, tMin, tMax, tDelta の設定

これらのパラメータは出力する関数に指定することができる変数 x と t の値の範囲を決定するのに用いられます。xMin, xMax, xDelta はそれぞれ変数 x の最小値、最大値、増加量、tMin, tMax, tDelta はそれぞれ変数 t の最小値、最大値、増加量です。つまり、変数はそれぞれ初期値が xMin, tMin で、値が xMax, tMax を超えるまで xDelta, tDelta ずつ増加していきます。例えば、xMin=0, xMax=3, xDelta=2 なら、x は 0→2 と変化し、xMin=0, xMax=3, xDelta=1.5 なら、x は 0→1.5→3 と変化します。また、設定値は xMin≤xMax, xDelta>0, tMin≤tMax, tDelta>0 でなければなりません。ただし、xMin=xMax または tMin=tMax のときには、それぞれ xDelta=0, tDelta=0 でも構いません。

変数 x は各出力の配列の要素ごとに変化する変数です。各出力のデータ長は、変数 x の範囲によって決まります。例えば、xMin=0, xMax=4, xDelta=1 とし、出力する関数を $x+1$ とすれば、出力されるデータは 1, 2, 3, 4, 5 で、データ長は 5 となります。定数関数の場合にも変数 x の範囲に従ってデータ長が決定し、値が出力されます。例えば、xMin=0, xMax=4, xDelta=1 とし、出力する関数を 1 とすれば、出力されるデータは 1, 1, 1, 1, 1 で、やはりデータ長は 5 となります。次のデータを出力する際には、x の値は xMin に戻ります。

変数 t はコンポーネントが値を出力するたびに変化する変数です。ですので、実際の時間の変化と t の値は必ずしも一致しません。t の値が tMax を超えると、t の値は tMin に戻ります。例えば、xMin=0, xMax=4, xDelta=1 とし、tMin=0, tMax=2, tDelta=1、出力する関数を $x+t$ とすれば、出力されるデータはデータ長が 5 で、3 パターンのデータを繰り返し出力することになります。つまり、出力データは

最初は	0,	1,	2,	3,	4
	(=0+0)	(=1+0)	(=2+0)	(=3+0)	(=4+0)
2 番目は	1,	2,	3,	4,	5
	(=0+1)	(=1+1)	(=2+1)	(=3+1)	(=4+1)
3 番目は	2,	3,	4,	5,	6
	(=0+2)	(=1+2)	(=2+2)	(=3+2)	(=4+2)

となり、以後このデータの繰り返しとなります。

(b) 出力する関数 outputFunctionOrLoadFileName の設定

outputFunctionOrLoadFileName には、出力する関数として $f(x,t)$ の形で表現可能な陽関数を指定します。このとき、例えば $\sin(2*pi*(x-t))$ といったように、プログラミング言語などにおいても一般に用いられている記法によって出力する関数を記述することができます。outputFunctionOrLoadFileName に指定可能な関数、演算子、定数、変数は以下の通りです。

- 関数

  ここでは、定義域の説明において引数を x で表しています。複数の引数がある場合には第 1 引数から順に $x_1, x_2, \cdots$ としています。

関数名	引数の数	定義域	説明
abs	1	$-\infty < x < \infty$	引数 x の絶対値を計算する。
acos	1	$-1 \leq x \leq 1$	引数 x の逆余弦の主値 $[0, \pi]$ [rad] を計算する。
asin	1	$-1 \leq x \leq 1$	引数 x の逆正弦の主値 $[-\frac{\pi}{2}, \frac{\pi}{2}]$ [rad] を計算する。
atan	1	$-\infty < x < \infty$	引数 x の逆正接の主値 $[-\frac{\pi}{2}, \frac{\pi}{2}]$ [rad] を計算する。
atan2	2	$x_1 \neq 0$ または $x_2 \neq 0$	引数 x_1/x_2 の逆正接の主値 $[-\pi, \pi]$ [rad] を計算する。このとき、引数の符号を用いて返却値の象限を決定する。
ceil	1	$-\infty < x < \infty$	引数 x 以上の最小の整数値を計算する。
cos	1	$-\infty < x < \infty$	引数 x [rad] の余弦を計算する。
cosh	1	$-\infty < x < \infty$	引数 x の双曲線余弦を計算する。
exp	1	$-\infty < x < \infty$	ネイピア数（自然対数の底）e のべき乗 e^x を計算する。
floor	1	$-\infty < x < \infty$	引数 x 以下の最大の整数値を計算する。
ln	1	$0 < x < \infty$	引数 x の自然対数を計算する。
log	1	$0 < x < \infty$	引数 x の常用対数を計算する。
random	0	–	乱数 $[0, 1]$ を返す。
sign	1	$-\infty < x < \infty$	引数 x の符号を表す値を返す。関数の値は x が負のとき -1、0 のとき 0、正のとき 1 となる。
sin	1	$-\infty < x < \infty$	引数 x [rad] の正弦を計算する。
sinh	1	$-\infty < x < \infty$	引数 x の双曲線正弦を計算する。
sqrt	1	$0 \leq x < \infty$	引数 x の非負の平方根を計算する。
tan	1	$x \neq \frac{\pi}{2} + n\pi$ (n は整数)	引数 x [rad] の正接を計算する。
tanh	1	$-\infty < x < \infty$	引数 x の双曲線正接を計算する。

- 演算子

優先順位	演算子	結合	説明
1	^	右	べき乗。0 の非正数乗および負数の非整数乗の場合は定義されない。
2	単項 +	右	正の符号。
	単項 −	右	負の符号。
3	∗	左	乗算。
	/	左	除算。除数が 0 である場合は定義されない。
	%	左	剰余算。除数が 0 である場合は定義されない。
4	+	左	加算。
	−	左	減算。

また、この他に、演算順序を指定するための左括弧 '(' および右括弧 ')' を使用することができます。

- 定数

定数名	説明
pi	円周率 $\pi = 3.141592...$。
e	ネイピア数（自然対数の底）$e = 2.71828...$。

- 変数

変数名	説明
x	1 組の出力データにおいて、xMin を初期値とし、値が xMax を超えるまで xDelta ずつ増加する。
t	tMin を初期値とし、コンポーネントの周期ごとに tDelta ずつ増加する。値が tMax を超えると tMin に戻る。

0 除算や sqrt(-1) など、定義されない入力がある場合の要素の値は、浮動小数点型のデータ (double, float) に関しては NaN、 その他の型については 0 となります。

また、RT コンポーネントでは、位置と姿勢 (x, y, θ) を一つの入力にする、といったように、必ずしもお互いに相関があるとは限らないデータをひとまとめにすることがあります。このような場合に対応するため、コロン ':' で区切ることで複数の関数を指定し、出力を行うことができます。例えば、xMin=0, xMax=3, xDelta=1 とし、出力する関数を x:x+1:x^2 とすれば、出力されるデータは、データ長が 4×3=12 で、

0, 1, 0, 1, 2, 1, 2, 3, 4, 3, 4, 9
(=0+1) (=0^2) (=1+1) (=1^2) (=1+2) (=2^2) (=1+3) (=3^2)

となります。

3.2.2 基本的な使い方 (2) – ファイル出力機能の利用 –

次に、出力データをファイルに保存しておき、それを出力する機能について説明します。ファイルを自分で作成することも可能ですが、FunctionGenerator にはファイル作成の手間を低減するため、他のコンポーネントから入力ポートに入力されたデータを保存できる機能も備えています。これにより、予めセンサ出力などを保存しておき、センサ情報処理コンポーネントの動作を同一の入力条件で処理パラメータを変えながらテストしたり、複数の処理コンポーネントの性能を比較したりすることも可能です。この使用手順をまとめると以下のようになります。

FunctionGenerator の使用手順 – ファイル出力機能の利用 –

- ファイルの作成
  1. Configuration を設定する
     (a) ファイルの保存形式 saveFileMode を設定する
     (b) 保存するファイル名 saveFileName を設定する
  2. InPort をデータを保存するコンポーネントと接続する
  3. コンポーネントをアクティブ化する
- ファイルの読み込み
  1. Configuration を設定する
     (a) outputFunctionOrLoadFileName に読み込むファイル名を設定する
  2. OutPort をデータを与えたいコンポーネントと接続する
  3. コンポーネントをアクティブ化する

それぞれの手順 1, 2, 3 は前後しても構いません。ただし、数学関数出力の場合と同様、ファイルの読み込みにおいては手順 1 でファイル名を設定し、手順 3 のアクティブ化を行うとデータ出力が始まりますので、コンポーネントの接続が行われていないと相手のコンポーネントがその部分のデータを受け取れなくなってしまいます。したがって、時間とともに異なる値を出力するような場合には注意が必要です。

以下ではファイルの作成、読み込みのそれぞれにおける Configuration の設定について説明します。

(a) ファイルの作成

ファイルの作成時には、入力されたデータを全て受け取れるように、コンポーネントの実行周期をデータの受信周期より早くしておく必要があります。コンポーネントの実行周期は rtc.conf の exec_cxt.periodic.rate の値で設定することができますので、十分大きな値を設定してからコンポーネントを起動してください。

まず、ファイルの保存形式を saveFileMode に指定します。w を指定すると、指定したファイルを読み込みモードで生成または開きます。outputFunctionOrLoadFileName に指定したものと同名のファイルが存在した場合には、その内容を消去し、データの書き込みを開始します。a を指定すると、指定したファイルを追記モードで生成または開きます。outputFunctionOrLoadFileName に指定したものと同名のファイルが存在した場合には、その内容の最後にデータを追加します。

次に、保存するファイル名を saveFileName に設定します。拡張子は.txt, .csv, .dat のいずれかを指定してください。その他の拡張子を指定した場合には自動的に.txt の拡張子が付けられます。

アクティブ化した状態でいずれかの入力ポートが情報を受け取ると、指定されたファイルにそのデータが書き込まれます。

ファイルを自分で作成する場合

ファイルを自分で作成する場合には、データをカンマ、タブ、スペース、改行のいずれかで区切って入力します。また、改行のみの行を入力することで、データをブロックに分けることができます。データ出力はブロック単位で行われ、コンポーネントの実行周期で順にブロックを出力していきます。ファイルの終端に到達すると、ファイルの最初から出力を再開します。

例として、以下のようなデータをファイルに保存したとします。

```
1,  2,  3
4,  5,  6

1,  4,  9

3,  6,  9
12,  15
```

このとき、出力されるデータは

```
最初は     1,  2,  3,  4,  5,  6
2 番目は   1,  4,  9
3 番目は   3,  6,  9,  12,  15
```

となり、以後このデータの繰り返しとなります。

(b) ファイルの読み込み

ファイルを読み込む際には、outputFunctionOrLoadFileName にファイル名を指定してください。ファイルの作成と同様、拡張子は.txt, .csv, .dat のいずれかを指定してください。

4 (参考) FunctionGenerator の利用例

本章では、FunctionGenerator の利用例に関して、他のコンポーネントとの連携を中心に説明します。

4.1 本章で利用しているコンポーネント群

ファンクションジェネレータコンポーネントの利用例の紹介では、これまでの 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。

- ビューワコンポーネント (GnuplotViewer)

  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

- 音声入力コンポーネント (VoiceIn), 音声出力コンポーネント (VoiceOut)

  MIKS, “VoiceCell”, RT ミドルウエアコンテスト 2007 応募作品 [2007103109]

- 画像表示コンポーネント (ShowComponent)

  田窪朋仁, 大原賢一, 吉岡健伸, “画像処理学習用 RT コンポーネントライブラリ”, RT ミドルウエアコンテスト 2008 応募作品 1L4-4

- パン-チルト-ズームカメラコンポーネント (PTZCameraSony)

  佐々木毅, 橋本秀紀, “効率的な入力データ生成のためのファンクションジェネレータコンポーネント”, RT ミドルウエアコンテスト 2009 応募作品 1A3-6

これらのコンポーネントの詳細については、各応募作品に同梱されているマニュアルや開発者のサポートページ等を参照してください。また、これらのコンポーネントは例えば OpenRTM-aist のバージョン 0.4.x のみに対応している場合があり、最新バージョンで動作させる場合にはソースの変更が必要となる可能性がありますのでご了承ください。

4.2 ビューワコンポーネントを用いた出力データの可視化

FunctionGenerator の出力データをビューワコンポーネントを利用して可視化する例を紹介します。ここではビューワコンポーネントとして、RT ミドルウエアコンテスト 2008 の応募作品 1L3-2

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”

を利用します。また、GnuplotViewer 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。

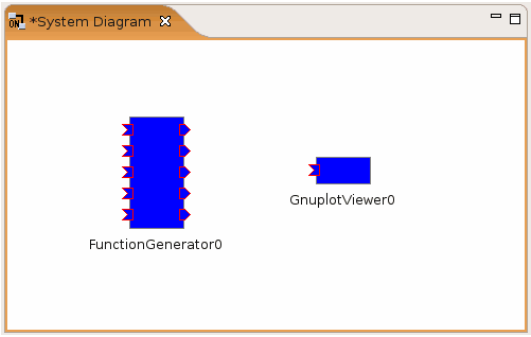
この例で使用するコンポーネントの種類とその数は

- ファンクションジェネレータコンポーネント (FunctionGenerator) · · · · · · 1 つ
- ビューワコンポーネント (GnuplotViewer) · · · · · · 1 つ

の計 2 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

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



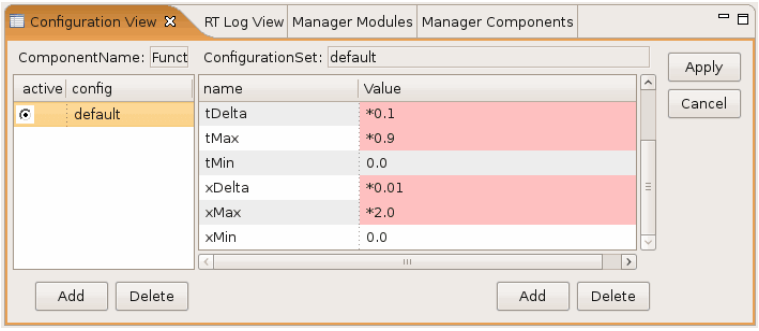
(2) Configuration 変数の設定

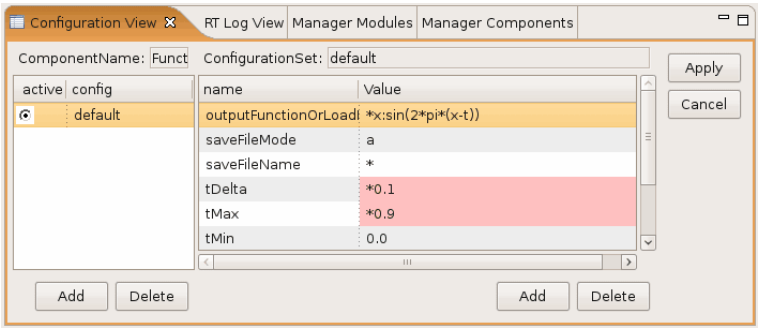
ここでは、FunctionGenerator から時間とともに x 軸方向に移動する正弦波を出力し、それを GnuplotViewer で視覚化する例を示します。

まず、FunctionGenerator を選択し、ConfigurationView からパラメータを設定します。最初に変数の範囲を決定します。ここでは変数 x の範囲を xMin=0.0, xMax=2.0, xDelta=0.01、変数 t の範囲を tMin=0.0, tMax=0.9, tDelta=0.1 とします。次に出力する関数を指定します。ここでは、変数 x の値と正弦波の値を出力することとし、outputFunctionOrLoadFileName に以下の関数を設定します。

x:sin(2*pi*(x-t))

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。

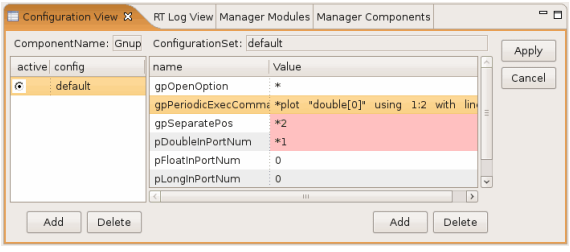




続いて、GnuplotViewer を選択し、ConfigurationView からパラメータを設定します。まず、使用するポートの種類と数の設定を行います。この例では、TimedDoubleSeq 型でデータを受け取ることとしますので、pDoubleInPortNum の値を 1 にします。次に、1 次元データ配列の区切り位置を設定します。FunctionGenerator の OutPort からは、変数 x の値と sin() に相当する値が、xData[0], sinData[0], xData[1], sinData[1], $\cdots$ というように出力されます。つまり、2 つごとに同種のデータが出力されることになりますので、gpSeparatePos の値を 2 にします。最後に 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()) の値をプロットしますので、

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）。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。

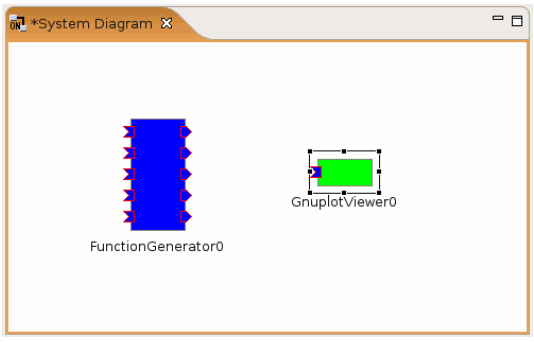


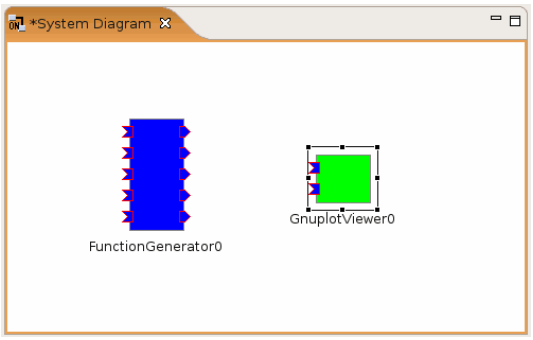
(3) コンポーネントのアクティブ化

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

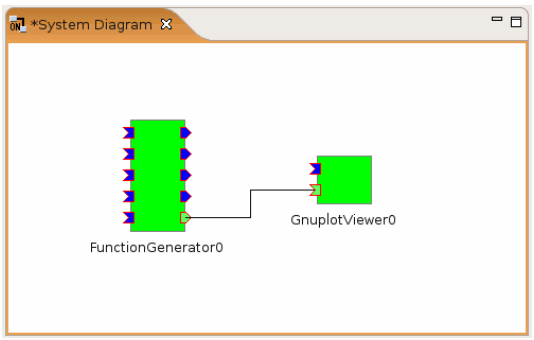
Set separate position list.
2
Set command: plot ”double[0]” using 1:2 with lines

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。

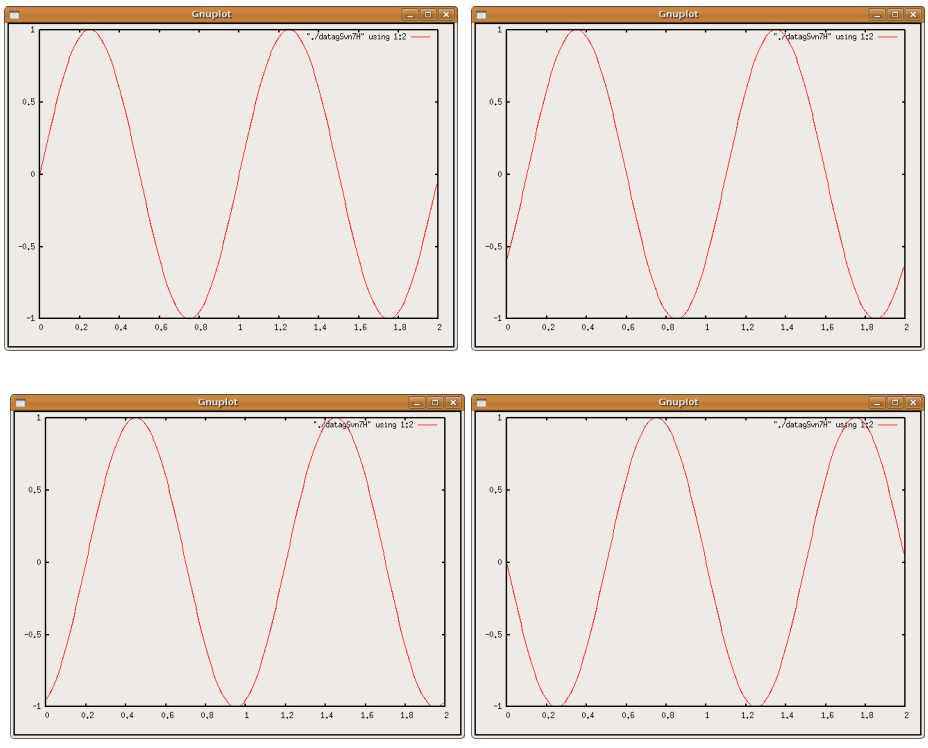




FunctionGenerator の oDoubleSeqData ポートと GnuplotViewer の DoubleSeqData ポートを接続し、FunctionGenerator もアクティブ化します。



すると、FunctionGenerator からデータが送られてきたことによって 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。



4.3 音声入出力コンポーネントを用いた音によるデータの提示

FunctionGenerator の出力データを音声出力コンポーネントを用いて音として提示したり、音声入力コンポーネントからのデータを FunctionGenerator を利用して保存・読み込みを行い、音声出力コンポーネントで再生したりする例を紹介します*。ここでは音声入出力コンポーネントとととして、RT ミドルウエアコンテスト 2007 の応募作品 [2007103109]

MIKS, “VoiceCell”

を利用します。

4.3.1 音情報の提示

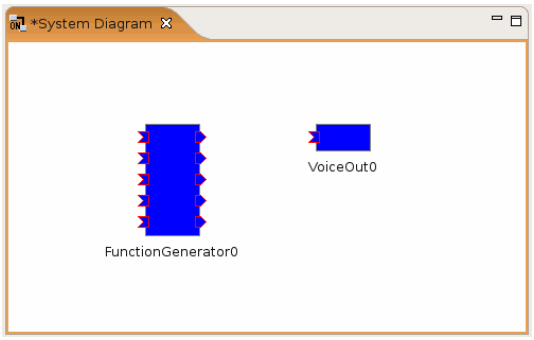
この例で使用するコンポーネントの種類とその数は

- ファンクションジェネレータコンポーネント (FunctionGenerator) ・・・・・・ 1 つ
- 音声出力コンポーネント (VoiceOut) ・・・・・・ 1 つ

の計 2 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

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



(2) Configuration 変数の設定

ここでは、FunctionGenerator を用いて 440Hz の正弦波を発生させ、それを VoiceOut で再生します。

VoiceOut の Configuration については設定する必要はありませんので、FunctionGenerator の Configuration のみを考えます。FunctionGenerator の Configuration の設定に際して、まず VoiceOut のインタフェースと内部のパラメータについて確認する必要があります。VoiceOut の InPort の型は TimedOctetSeq となっています。また、VoiceOut では、“libsound.h” でサンプリングパラメータを以下のように設定しています。

パラメータ	設定
サウンドフォーマット SIZE	量子化ビット数 16bits、符号付き、リトルエンディアン
サンプリング周波数 RATE	16.0kHz
チャンネル数 CHANNELS	2（ステレオ）
再生時間 LENGTH	1sec

これらを基に、最初に変数の範囲を決定します。ここでは 1 秒分のデータを送ることとし、サンプリング周波数が 16.0kHz のため、変数 x の範囲を xMin=0.0, xMax=15999.0, xDelta=1.0 とします。変数 t の範囲は特に設定する必要はありません（有効な値であれば何でも構いません）。

*ただし、本節の例はリアルタイム OS を使用していない場合、遅れや音飛びを生じることがあります。また、実行に際してはスピーカの音量にお気をつけください。

次に出力する関数を指定します。出力する正弦波に用いる式は振幅を A、周波数を f とすると

$$A\text{*sin(2*pi*}f\text{*x/16000)}$$

となりますが、VoiceOut では前述のように各々が符号付き 16bits のデータを octet 型（符号なし 8bits）データで受信します。また、チャンネル数が 2 となっているため、出力するデータには左チャンネル、右チャンネルのデータをそれぞれ含む必要があります。つまり、出力データは以下のような形式になります。

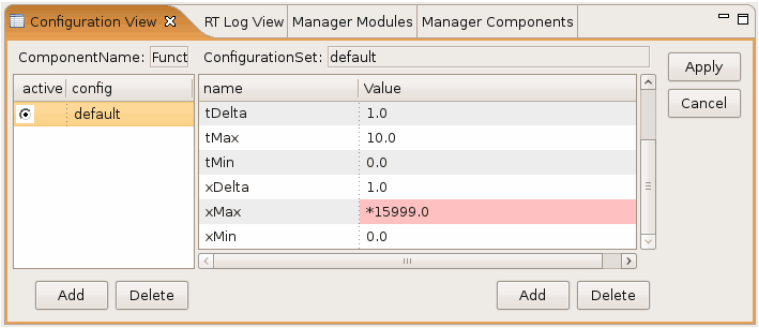
$f_{ll}(x,t)$	:	$f_{lh}(x,t)$	:	$f_{rl}(x,t)$	:	$f_{rh}(x,t)$
左チャンネル		左チャンネル		右チャンネル		右チャンネル
下位バイト		上位バイト		下位バイト		上位バイト

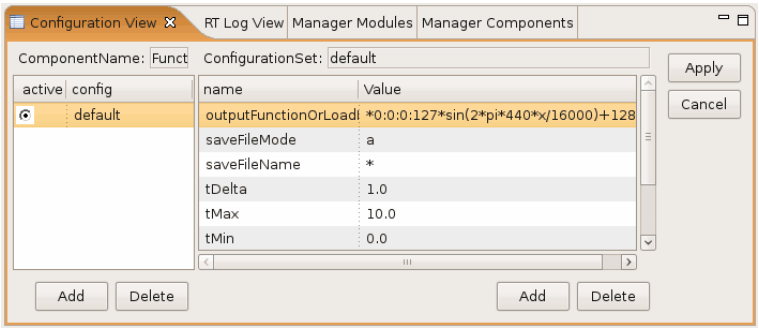
ここでは、出力は右チャンネルのみとし、また正弦波のデータも上位ビットのみで近似することとします。また、ビット列として符号なしデータを符号付きのデータと一致させる必要があるため、そのための補正も必要となります。以上のことから、outputFunctionOrLoadFileName に以下の関数を設定します。

0:0:0:127*sin(2*pi*440*x/16000)+128*(sign(sin(2*pi*440*x/16000))-1)*sign(sin(2*pi*440*x/16000))

第 2 項の 128*(sign(sin(2*pi*440*x/16000))-1)*sign(sin(2*pi*440*x/16000)) の部分が符号の補正部分に相当します。

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。





VoiceOut のサンプリングパラメータを変更した場合

例えば “libsound.h” のサンプリングパラメータを CHANNELS=1（モノラル）に変更した場合、左右のチャンネルのデータを分ける必要がなくなりますので、出力データの形式は以下のようになります。

$f_l(x,t)$	:	$f_h(x,t)$
両チャンネル		両チャンネル
下位バイト		上位バイト

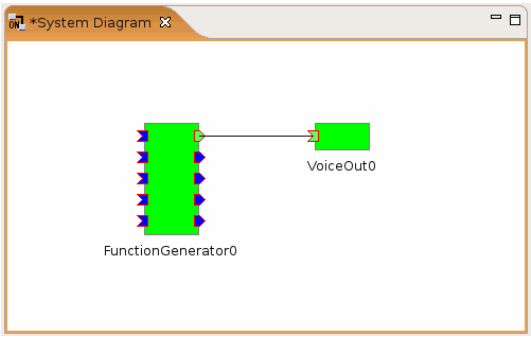
また、CHANNELS=1 に加えて、さらに SIZE=8 とした場合には、量子化ビット数 8bits、符号なしとなりますので、出力データの形式は $f_(x,t)$ のみとなります。この場合、本節の例では、outputFunctionOrLoadFileName に設定する関数は以下のようになります。

127*(sin(2*pi*440*x/16000)+1)

この式の最後の+1 は、データが符号なしのため [-128:127] ではなく [0:255] の範囲内で値を表現するためのものです。

(3) コンポーネントのアクティブ化

FunctionGenerator の oOctetSeqData ポートと VoiceOut の in ポートを接続し、両コンポーネントをアクティブ化します。



すると、FunctionGenerator からデータが送られてきたことによって、440Hz(A4) の音が右スピーカから出力されます。

4.3.2 音声の録音 · 再生

この例で使用するコンポーネントの種類とその数は

- ファンクションジェネレータコンポーネント (FunctionGenerator) · · · · · · 1 つ
- 音声入力コンポーネント (VoiceIn) · · · · · · 1 つ
- 音声出力コンポーネント (VoiceOut) · · · · · · 1 つ

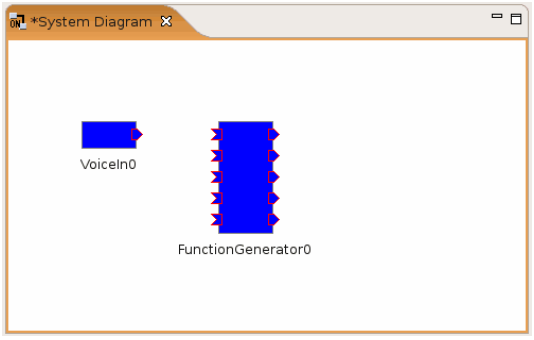
の計 3 つです。以下に使用手順を示します。

• ファイルの作成

まず、音声データを録音し、ファイルを作成する手順を説明します。

(1) コンポーネントの配置

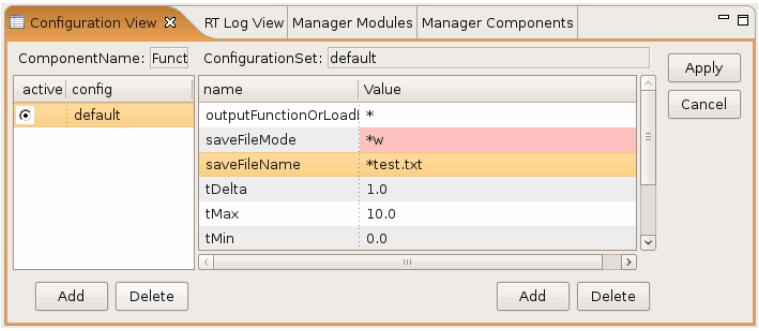
コンポーネントを起動し、システムエディタ上に FunctionGenerator, VoiceIn を配置してください。



(2) Configuration 変数の設定

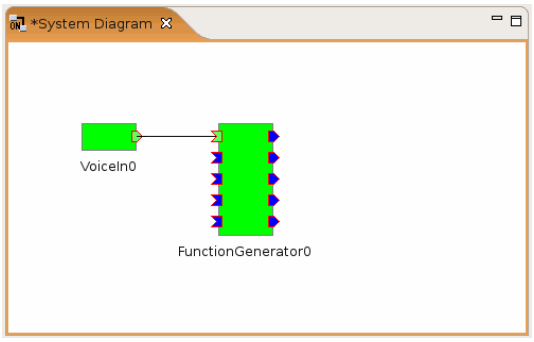
VoiceIn の Configuration については設定する必要はありませんので、FunctionGenerator の Configuration のみを考えます。FunctionGenerator を選択し、ConfigurationView からファイルの保存モードとファイル名を設定します。ここでは、saveFileMode を w（既存のファイルがある場合には上書き）、saveFileName を test.txt とします。

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

VoiceIn の out ポートと FunctionGenerator の iOctetSeqData ポートを接続し、両コンポーネントをアクティブ化します。



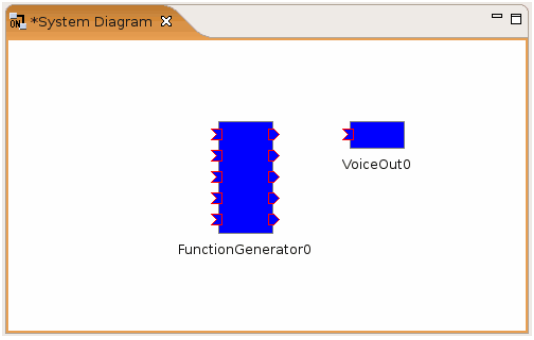
マイクから音声を入力し、終了したら両コンポーネントを非アクティブ化してください。

- ファイルの読み込み

次に、保存した音声データを読み込み、音声を再生する手順を説明します。

(1) コンポーネントの配置

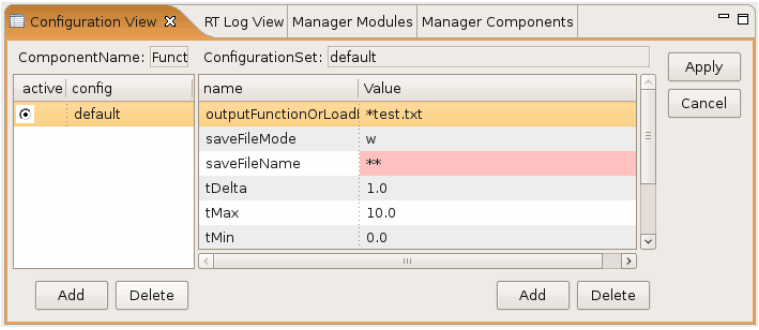
コンポーネントを起動し、システムエディタ上に FunctionGenerator, VoiceOut を配置してください。



(2) Configuration 変数の設定

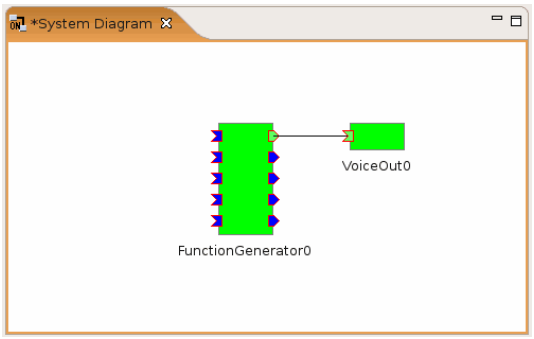
VoiceOut の Configuration については設定する必要はありませんので、FunctionGenerator の Configuration のみを考えます。FunctionGenerator を選択し、ConfigurationView から読み込むファイル名を設定します。ここでは、outputFunctionOrLoadFileName に先ほど保存した test.txt を指定します。録音に続けて再生する場合には、保存と読み込みの重複アクセスを避けるため、saveFileName をデフォルト値である*にしておきます。

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

FunctionGenerator の oOctetSeqData ポートと VoiceOut の in ポートを接続し、両コンポーネントをアクティブ化します。



すると、FunctionGenerator からデータが送られてきたことによって、先ほど録音した音声がスピーカから出力されます。

4.4 パン-チルト-ズームカメラコンポーネントを用いた広範囲観測

FunctionGenerator を用いてパン-チルト-ズームカメラコンポーネントにより広範囲の観測を行う例を紹介します。ここでは、カメラを左右に回転させるため RT ミドルウエアコンテスト 2008 の応募作品 1A3-6

佐々木毅, 橋本秀紀, “効率的な入力データ生成のためのファンクションジェネレータコンポーネント”

の中で開発されたパン-チルト-ズームカメラコンポーネント (PTZCameraSony) を利用します。また、画像表示コンポーネントとして、RT ミドルウエアコンテスト 2008 の応募作品 1L4-4

田窪朋仁, 大原賢一, 吉岡健伸, “画像処理学習用 RT コンポーネントライブラリ”

を利用します。上記作品では画像表示コンポーネント以外にも様々な画像処理コンポーネントを提供しています。パン-チルト-ズームカメラコンポーネントは、この作品のカメラコンポーネントと同様のインタフェースとしているため、これら画像処理コンポーネントも利用することができます。

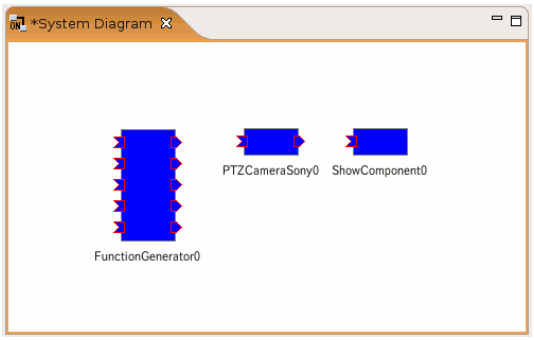
この例で使用するコンポーネントの種類とその数は

- ファンクションジェネレータコンポーネント (FunctionGenerator) · · · · · · 1 つ
- パン-チルト-ズームカメラコンポーネント (PTZCameraSony) · · · · · · 1 つ
- 画像表示コンポーネント (ShowComponent) · · · · · · 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

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



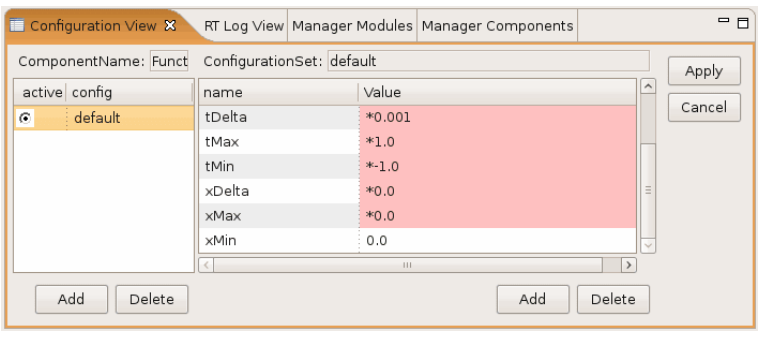
(2) Configuration 変数の設定

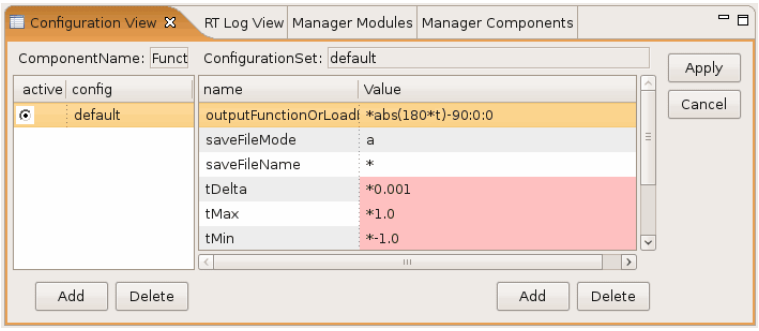
ここでは、FunctionGenerator からパン-チルト-ズームのパラメータを生成し、パン方向にカメラを等速で往復運動させる例を示します。

まず、FunctionGenerator を選択し、ConfigurationView からパラメータを設定します。最初に変数の範囲を決定します。ここでは変数 x の範囲を xMin=0.0, xMax=0.0, xDelta=0.0、変数 t の範囲を tMin=-1.0, tMax=1.0, tDelta=0.001 とします。次に出力する関数を指定します。ここでは、変数 x の値と正弦波の値を出力することとし、outputFunctionOrLoadFileName に以下の関数を設定します。

abs(180*t)-90:0:0

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。



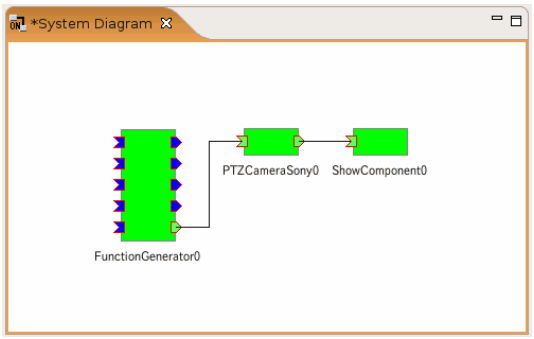


続いて、PTZCameraSony を選択し、ConfigurationView からパラメータを設定します。まず、パン-チルト-ズームユニットのデバイス名 str_ptz_dev とカメラインデックス int_camera_index の設定を行います。通常はデフォルトのままで良いと思います。次に、画像の幅 int_width と高さ int_height を設定します。そのカメラが対応しているサイズを設定してください。

最後に、ShowComponent を ConfigurationView からパラメータを設定します。画像の幅 int_width と高さ int_height を、PTZCameraSony で設定した値と同じ値に設定してください。

(3) コンポーネントのアクティブ化

FunctionGenerator の oDoubleSeqData ポートと PTZCameraSony の PanTiltZoom ポート、PTZCameraSony の ImageData ポートと ShowComponent の ImageData ポートを接続し、アクティブ化します。



すると、FunctionGenerator からデータが送られてきたことによって、カメラがパン方向に往復運動を行います。また、その時のカメラ画像がウインドウ内に表示されます。画像が表示されない場合には、Configuration の設定を見直してみてください。

5　お問い合わせ

本コンポーネントに関しましては、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