RT コンポーネント操作マニュアル

屋内地図モデルの簡易生成コンポーネント

2014 年 12 月 2 日版

芝浦工業大学

ロボティクスシステムデザイン研究室

立川将　土屋彩茜　奥野万丈　竹村友晃　佐々木毅

更新履歴

2014 年 10 月 31 日版　第 1 版.

2014 年 11 月 02 日版　第 2 版.

　　第 2 章 5 節、第 3 章 3 節をコンポーネントの修正に伴い、加筆・修正しました

2014 年 11 月 20 日版　第 3 版.

　　第 4 章を加筆・修正しました

2014 年 12 月 2 日版　第 4 版.

　　第 4 章を加筆・修正しました

目次

1. 本提案の概要

1.1 提案の背景

現在大規模商業施設の増加など、屋内環境を把握する必要がある場面は増えてきています。しかし、テナントなどにより構造が多様化するといった理由から、環境把握は困難なものとなっております。そこで、既存の地図を利用してシステム上の地図モデルを簡易生成することができれば、屋内サービスへの発展が期待されると考えました。

このような背景から、屋内地図の簡易生成アプリケーションを RT コンポーネント群として開発することとなりました。これにより屋内向けサービスロボットへの軌道生成や、Google glass など別アプリケーション上での利用など、他の RT と関連した様々な用途が可能となります。また、一つのアプリケーションとしてではなく、複数のコンポーネントとして開発を行ったため、テストケースや画像処理コンポーネントとしてそれぞれ利用していただくことができます。

コンポーネントの設計指針等につきましては、

立川　将, 土屋　彩茜, 奥野　万丈, 竹村　友晃, 佐々木　毅,
“屋内地図モデルの簡易生成コンポーネント”,
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、java 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 利用環境

本コンポーネント群は rtshell によって起動することができます。その際の必要環境は以下の通りです。rtshell のインストール方法や詳しい利用方法については（http://www.openrtm.org/openrtm/ja/node/5013#toc3）を参照してください。屋内地図モデルの簡易生成システムとして利用する場合は、rtshell で実行する方法を推奨しています。動作確認済みの環境は以下の通りです。

・RT ミドルウェア：OpenRTM-aist-1.1.0-RELEASE（Python 版）
・Python2.6
・PyYAML3.1.0
・rtshell3.1.0
・rtctree3.1.0
・rtsprofile3.1.0

1.4 開発したコンポーネント群

今回地図画像を処理する画像処理の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。

・エッジ検出コンポーネント(CVCanny)
・画像二値化コンポーネント(CVImgThreshold)
・ハフ変換コンポーネント(CVHoughThransform)
・特徴点検出コンポーネント(CVFeatureDetector)
・輪郭検出コンポーネント(CVFindContorours)
・選択領域修復コンポーネント(CVInpaint)

また、地図モデルを生成するためのツールとして、以下のコンポーネント群を開発しました。

・複数画像ファイル送信コンポーネント(OpenPicts)
・輪郭情報データを用いた画像修復コンポーネント(imageInpaint)
・ラインマップ変換コンポーネント(convToLineMap)
・地図モデル生成管理コンポーネント(mapsManager)
・UI(User Interface）コンポーネント(controlPanel)

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

2. 画像処理コンポーネント群の説明

画像処理コンポーネント群は OpenCV の各関数を元に作成されています。関数の持つ引数はコンフィギュレーションパラメータを用いて指定できるようになっています。また、処理中の画像データの表示に関しても、不要である場合が考えられるため、コンフィギュレーションパラメータより表示の ON/OFF の設定ができるようになっています。

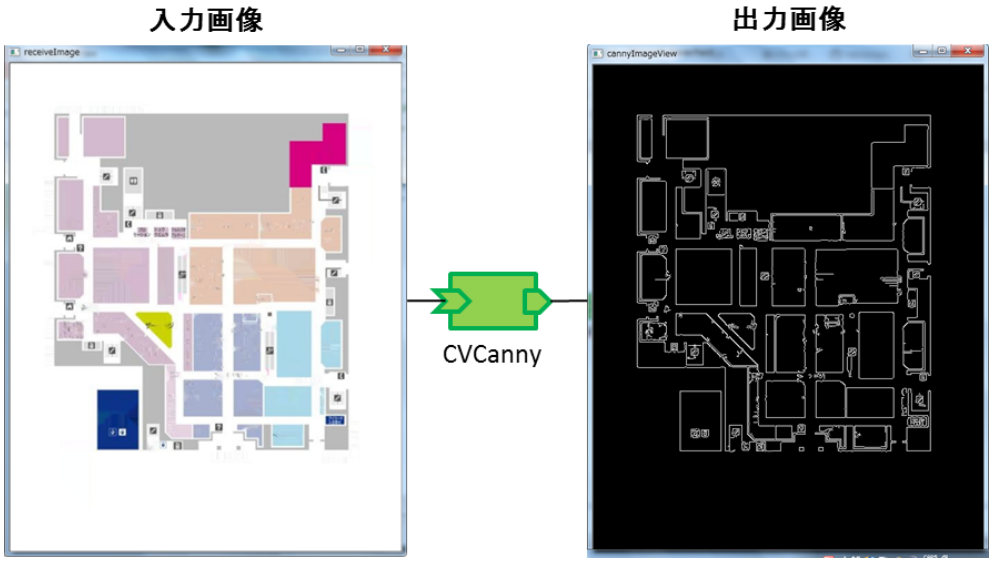
今回コンポーネント化した関数は以下のものになっています。また、関数に対する詳細

な説明につきましては、OpenCV のホームページでの確認をよろしくお願いいたします。

・canny()
・threshold()
・adaptedThreshold()
・FindContours
・FeatureDetect
・Houg
・ImageThreshold
・Inpaint

続いて、各画像処理コンポーネントの説明へ移ります。

2.1 エッジ検出コンポーネント(CVCanny)



CVCanny は、InPort の srcImage から入力された画像に対して、コンフィギュレーションパラメータを用いてエッジ検出を行うコンポーネントです。OutPort の cannyImage からは、入力画像から抽出したエッジデータの画像が出力されます。各コンフィギュレーションパラメータの説明は以下の表を参照してください。

・InPort

名称	型	説明
srcImage	CameraImage	エッジ検出を行う画像データを取得するポート。

・OutPort

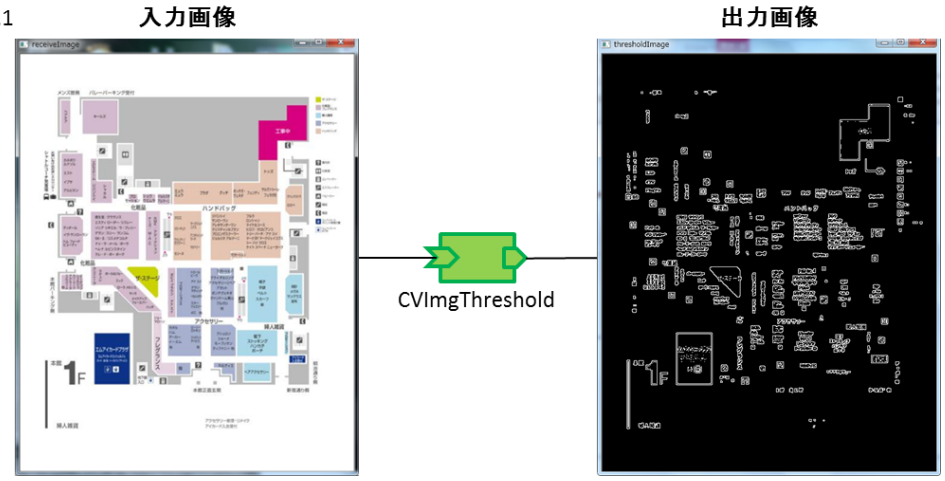
名称	型	説明
cannyImage	CameraImage	入力画像から抽出したエッジ画像を出力するポート。

・Configuration

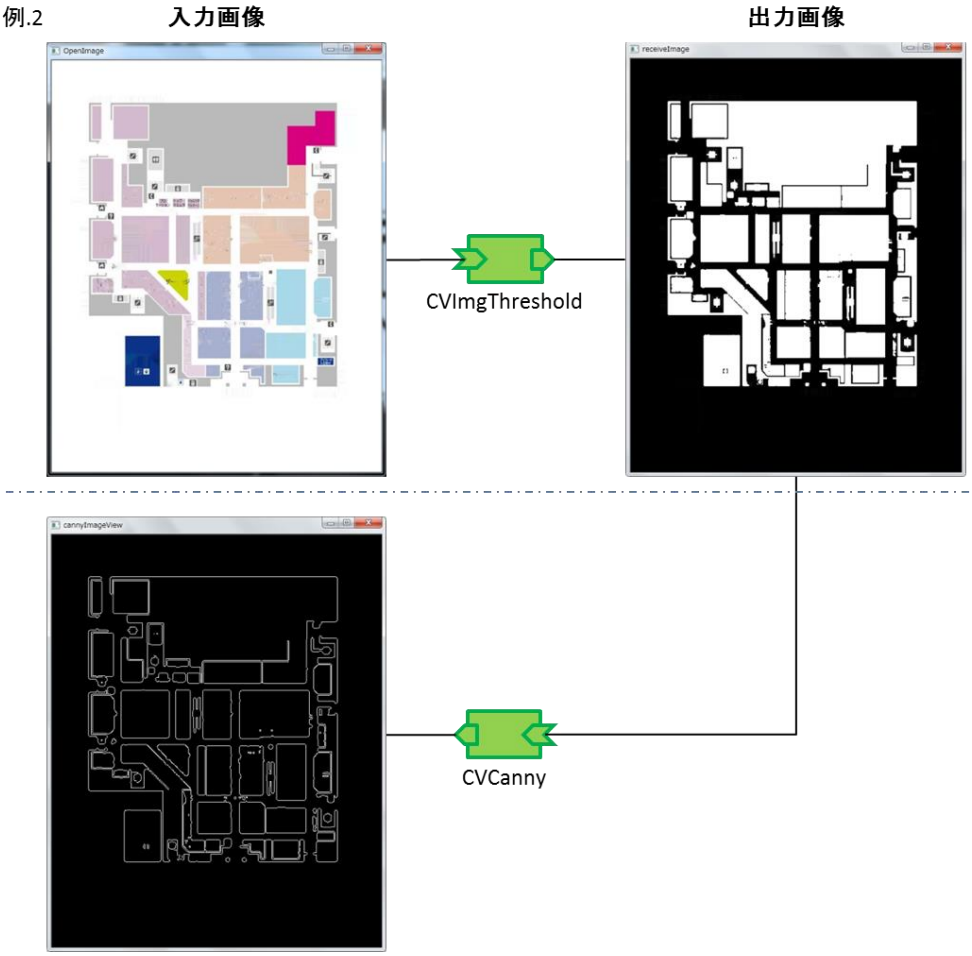
名称	型	Widget	説明
ImageView	string	radio	画像表示の ON/OFF を選択するための変数。 制約条件：ON,OFF デフォルト：OFF
threshold1	double	text	ヒステリシスが存在する処理の一番目の閾値。 制約条件：x > 0 デフォルト：50
threshold2	double	text	ヒステリシスが存在する処理の二番目の閾値。 制約条件：x > 0 デフォルト値：200
apertureSize	int	text	オペレータのアパーチャサイズ。 制約条件：x > 0 デフォルト：3
L2gradient	string	radio	画像勾配の強度取得の精度を選択するための変数。 制約条件：true,false デフォルト：true
	制約説明	true	L2 ノルムを用いて高い精度で強度取得を行う。
		false	L1 ノルムを用いて強度取得を行う。

2.2 画像二値化コンポーネント(CVImgThreshold)

例.1



例.2　　入力画像　　出力画像



CVImgThreshold は、InPort の srcImage から入力された画像に対して、コンフィギュレーションパラメータで与えられた値を用いて二値化処理を行い、OutPort の thresholdImage から作成した二値画像を出力するコンポーネント。

二値化処理アルゴリズムは、cvThreshold と cvAdaptiveThreshold を使うことができ、一つの画像に複数回の処理を行うことが可能です。

・InPort

名称	型	説明
srcImage	CameraImage	二値化処理を行う画像データを取得するポート。

・OutPort

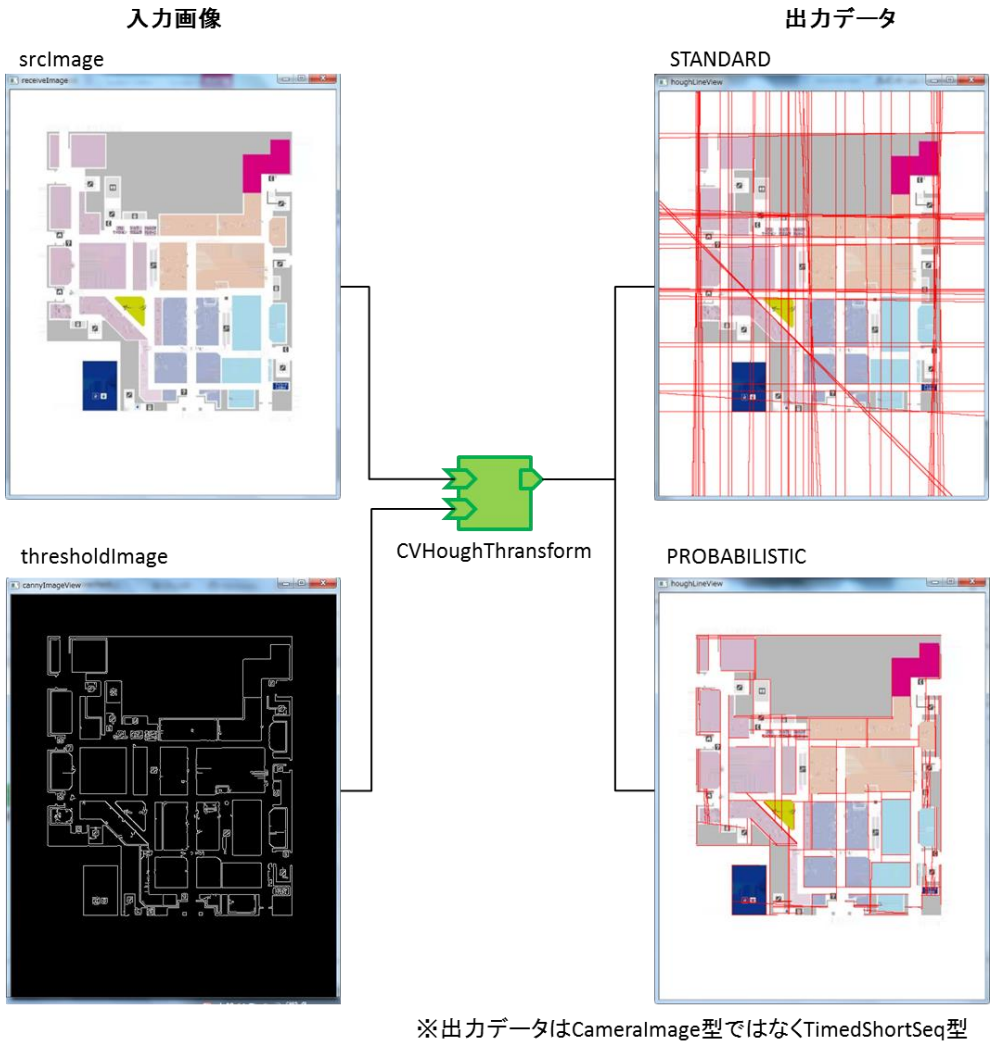
名称	型	説明
cannyImage	CameraImage	二値化処理された二値画像を出力するポート。

・Configuration

名称	型	Widget	説明
ImageView	string	radio	画像表示の ON/OFF を選択するための変数。 制約条件：ON,OFF デフォルト：OFF
Algorithm	vector <string>	ordered_list	二値化処理アルゴリズムを指定するための変数。 制約条件： NON,Binary,BinaryInv,Trunc,ToZero,ToZeroInv,Adaptive_MEAN_CCV,Adaptive_GAUSSIAN_C デフォルト： NON
Parameter	vector <double>	text	cvThreshold()と cvAdapticeThreshold()の引数。 cvThreshold()での引数は[0]~[2]を cvAdapticThreshold()での引数は[0]~[4]を使用。 また、入力に不備がある場合はデフォルト値を 用いて処理を行う。 デフォルト：0,0,255,0,0,255 ----------引数の説明---------- threPara[0] – usedFlag 0：デフォルト値で実行。 1：引数を用いて実行。 cvThreshold()の場合 threPara[1] – double threshold 処理の閾値として使用。 制約：0<=x<=255 デフォルト：0 threPara[2] – double maxValue threAlgo が BINARY と BINARY_INV の場合のみ使用する。二値化の MAX 値として使用。 制約：0<=x<=255 デフォルト：255 cvAdaptedThreshold()の場合 threPara[1] – double maxValue 二値化の MAX 値として使用。

			制約：0<=x<=255 デフォルト：255 threPara[2] – int threshold_type 閾値処理の種類を 0 か 1 かで選択する。 CV_THRESH_BINARY – 0 CV_THRESH_BINARY_INV – 1 制約：0 , 1 デフォルト：0 threPara[3] – int blockSize ピクセルに対する閾値を計算するために利用する、そのピクセルの近傍領域のサイズ(3, 5, 7 など) 制約：1 < x かつ奇数 デフォルト：3 threPara[4] – double param1 利用する手法にパラメータ デフォルト：10
GaussianBlur	string	radio	GaussianBlur を行うかどうかを決めるための変数。 制約条件：ON,OFF デフォルト：ON
GaussianSize	vector <short>	text	画像平均化のガウシアンカーネルサイズ。 制約条件：x[0] >= 0, x[1] >= 0 かつ奇数 デフォルト：3,3
BitwiseNot	string	radio	Bitwise_not を行うかどうかを決めるための変数。 制約条件：ON,OFF デフォルト：ON

2.3 ハフ変換コンポーネント(CVHoughThransform)



CVHoughThransform は、InPort の thresholdImage から入力された二値画像データに対して、コンフィギュレーションパラメータで設定した変換メソッドや値を用いてハフ変換を行い、その結果を InPort の srcImage から取得した画像データに書き込んでウィンドウに表示するコンポーネントです。InPort の srcImage を、二値化する前の画像を出力するポートに接続することで、ハフ変換の結果を二値化する前の画像に描画することができるため、変換閾値などを確認することができます。OutPort の houghLines からは、ハフ変換の結果となる Lines を配列化したデータが出力されます。各コンフィギュレーションパラメータの説明は以下の表を参照してください。

・InPort

名称	型	説明
srcImage	CameraImage	二値化する前の画像を取得するポート。 この画像に対し、ハフ変換の結果を書き込む。
thresholdImage	CameraImage	ハフ変換を行うための二値画像を取得するポート。 この接続がない場合、InPort の srcImage の画像を 固定値で二値化し、ハフ変換を行う。

・OutPort

名称	型	説明
houghLines	TimedShortSeq	ハフ変換によって得た線のデータを出力するポート。 線は line の端点として用いる Point p1,p2 の座標を - l1.p1.x [0] - l1.p1.y [1] - l1.p2.x [2] - l1.p2.y [3] として配列に格納したデータである。 [0]~[3]によって一本の線が表されている。

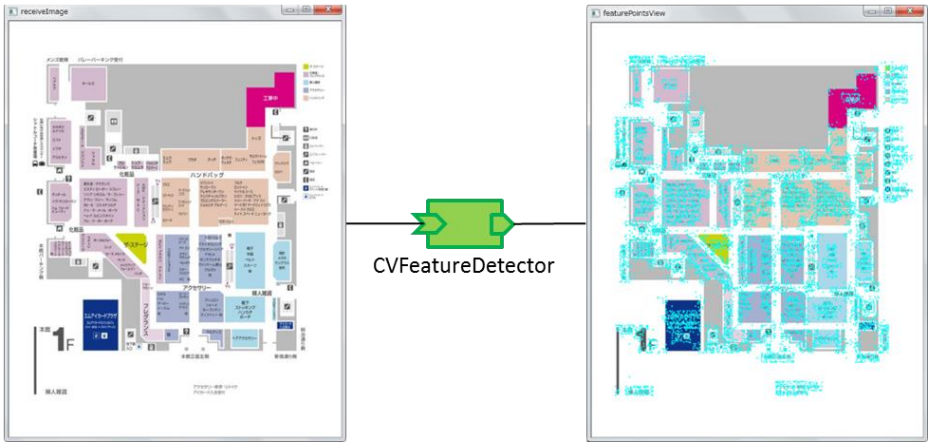
・Configuration

名称	型	Widget	説明
ImageView	string	radio	画像表示の ON/OFF を選択するための変数。 制約条件：ON,OFF デフォルト：OFF
HoughMethod	string	radio	ハフ変換の method を選択するための変数。 制約条件： STANDARD,MULTI_SCALE,PROBABILISTIC デフォルト：STANDARD
	制約説明	STANDARD	標準的ハフ変換
		MULTI_SCALE	マルチスケール型の古典的ハフ変換
		PROBABILISTIC	確率的ハフ変換
rho	double	text	ピクセル単位での投票空間の距離分解能。 制約条件：x > 0 デフォルト：1
theta	double	text	ラジアン単位での投票空間の角度分解能。 制約条件：x > 0

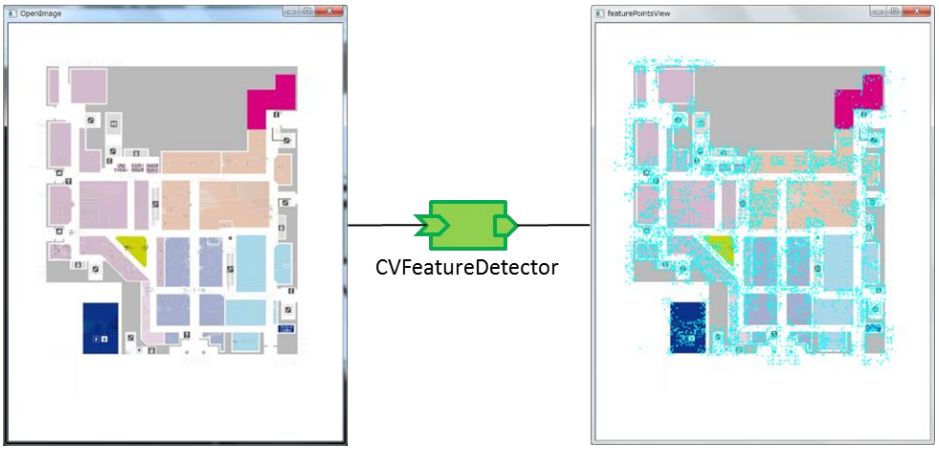
			デフォルト：0.0174533 (CV_PI/180)
threshold	int	text	投票の閾値。十分な票（ >threshold ）を得た直線のみが出力される。 制約条件：x > 0 デフォルト：100
srn	double	text	マルチスケールハフ変換において、 距離分解能 rho の除数となる値。 投票空間の粗い距離分解能は rho となり、 細かい分解能は rho / srn となる。 srn=0 かつ stn =0 の場合は，古典的ハフ変換が利用され、そうでない場合は、 両方のパラメータが正値である必要がある。 制約条件：x >= 0 デフォルト：0
stn	double	text	マルチスケールハフ変換において、 角度分解能 theta の除数となる値。 制約条件：x >= 0 デフォルト：0
minLineLength	double	text	最小の線分長。 これより短い線分は棄却される。 制約条件：x > 0 デフォルト：30
maxLineGap	double	text	2 点が同一線分上にあると見なす場合に許容される最大距離。 制約条件：x > 0 デフォルト：10

2.4 特徴点検出コンポーネント(CVFeatureDetector)

例.1　　入力画像　　出力データ



例.2　　入力画像　　出力データ



※出力データはCameraImage型ではなくTimedFloatSeq型

CVFeatureDetector は、InPort の srcImage から入力された画像に対して、コンフィギュレーションパラメータで選択されたメソッドやアダプタ、入力された閾値を用いて特徴点検出を行うコンポーネントです。OutPor の featurePoint からは、特徴点検出の結果となる KeyPoint を配列化したデータを出力します。各コンフィギュレーションパラメータの説明は以下の表を参照してください。

・InPort

名称	型	説明
srcImage	CameraImage	特徴点検出を行う画像データを取得するポート。

・OutPort

名称	型	説明
featurePoint	TimedFloatSeq	特徴点検出で得た KeyPoint のデータを出力するポート。 KeyPoint 型のデータを、以下のように分解して出力する。 KeyPoint - Point pt (pt.x [0] , pt.y [1]) - float size [2] - float angle [3] - float response [4] - int octave [5] - int class_id [6] これら 7 個のデータがそれぞれ配列の一要素として用いられている。

・Configuration

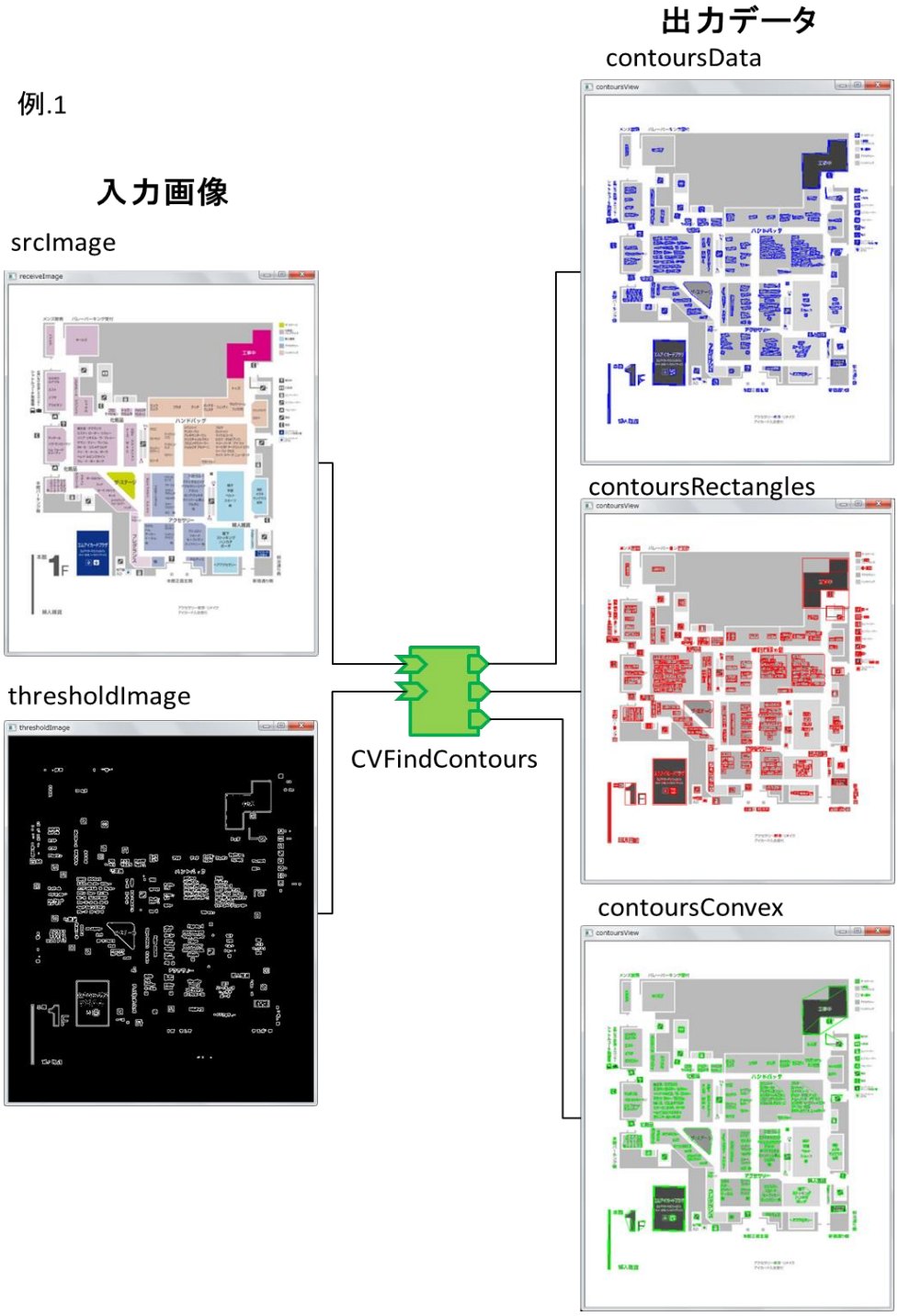
名称	型	Widget	説明
ImageView	string	radio	画像表示の ON/OFF を選択するための変数。 制約条件：ON,OFF デフォルト：OFF
Method	string	radio	特徴検出のメソッドを選択するための変数。 制約条件：FAST, Good, GoodHarris, Star, SIFT, SURF, MSER デフォルト：FAST
	制約説明	FAST	FAST()メソッドを用いた特徴検出
		Good	GoodFeaturesToTrack()クラスを用いた特徴検出
		GoodHarris	GoodFeaturesToTrack()クラスにおいて、HarrisDetector に true を与えた特徴検出
		Star	StarDetector()クラスを用いた特徴検出
		SIFT	SIFT()クラスを用いた特徴検出
		SURF	SURF()クラスを用いた特徴検出
		MSER	MSER()クラスを用いた特徴検出
Adapter	string	radio	特徴検出のアダプタを選択するための変数。 制約条件：NON, Grid, Pyramid, Dynamic デフォルト：NON
	制約説明	NON	アダプタを使用せず検出
		Grid	入力画像をグリッド状に分割して

			各セルでポイント検出
		Pyramid	ガウシアンピラミッドの複数レベルから検出
		Dynamic	指定した数の特徴が見つかるまで繰返し検出
FAST-threshold	int	text	FAST()のパラメータ。 制約条件：x > 0 デフォルト：1
Good-maxCorners	int	text	GoodFeatureToTrack()のパラメータ。 制約条件：x > 0 デフォルト：1000
Good-qualityLevel	double	text	GoodFeatureToTrack()のパラメータ。 制約条件：x > 0 デフォルト：0.01
Good-minDistance	double	text	GoodFeatureToTrack()のパラメータ。 制約条件：x > 0 デフォルト：1
Good-blockSize	int	text	GoodFeatureToTrack()のパラメータ。 制約条件：x > 0 デフォルト：3
Good-k	double	text	GoodFeatureToTrack()のパラメータ。 制約条件：x > 0 デフォルト：0.04
Star-maxSize	int	text	StarDetector()のパラメータ。 制約条件：x > 0 デフォルト：16
Star-responseThreshold	int	text	StarDetector()のパラメータ。 制約条件：x > 0 デフォルト：30
Star-lineThresholdProjected	int	text	StarDetector()のパラメータ。 制約条件：x > 0 デフォルト：10
Star-lineThresholdBinarized	int	text	StarDetector()のパラメータ。 制約条件：x > 0 デフォルト：8
Star-suppressNonmaxSize	int	text	StarDetector()のパラメータ。 制約条件：x > 0 デフォルト：5

SIFT-threshold	double	text	SIFT()のパラメータ。 制約条件：x > 0 デフォルト：0.05
SIFT-edge Threshold	double	text	SIFT()のパラメータ。 制約条件：x > 0 デフォルト：10.0
SURF-hessian Threshold	double	text	SURF()のパラメータ。 制約条件：x > 0 デフォルト：400.0
Grid-max TotalKeypoints	int	text	画像から検出されるキーポイントの最大数。 制約条件：x > 0 デフォルト：200
Grid-gridRows	int	text	グリッドの行数。 制約条件：x > 0 デフォルト：10
Grid-gridCols	int	text	グリッドの列数。 制約条件：x > 0 デフォルト：10
Pyramid-levels	int	text	スケーリングレベル。 制約条件：x > 0 デフォルト：3
Dynamic_ minFeatures	int	text	出力キーポイント個数の最小数。 制約条件：x > 0 デフォルト：400
Dynamic_ maxFeatures	int	text	出力キーポイント個数の最大数。 制約条件：x > 0 デフォルト：500
Dynamic_ maxIters	int	text	特徴検出処理の最大繰り返し回数。 制約条件：x > 0 デフォルト：10

2.5 輪郭検出コンポーネント(CVFindContours)

例.1

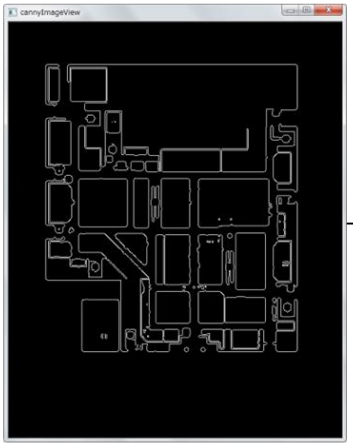


例.2

入力画像

srcImage

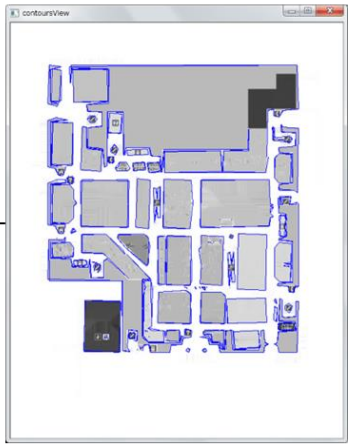
thresholdImage



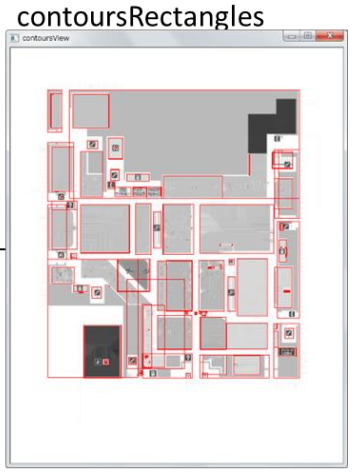
CVFindContours

出力データ

contoursData



contoursRectangles



contoursConvex

※出力データはCameraImage型ではなくTimedShortSeq型

CVFindContours は、InPort の thresholdImage から入力された二値画像データに対して、コンフィギュレーションパラメータで設定した変換メソッドや値を用いて輪郭検出を行い、その結果を InPort の srcImage から取得した画像データに書き込んでウィンドウに表示するコンポーネントです。InPort の srcImage を、二値化する前の画像を出力するポートに接続することで、輪郭検出の結果を二値化する前の画像に描画することができるため、変換閾値などを確認することができます。OutPort からは輪郭検出の結果となる Contours を配列化したデータが出力され、contoursData からは無変換の輪郭点データ、contoursRectangles からは長方形化した輪郭データ、contoursConvex は凸図形化した輪郭データが出力されます。

・InPort

名称	型	説明
srcImage	CameraImage	二値化する前の画像を取得するポート。 この画像に対し、輪郭検出の結果を書き込む。
thresholdImage	CameraImage	輪郭検出を行うための二値画像を取得するポート。 この接続がない場合、InPort の srcImage の画像を 固定値で二値化し、輪郭検出を行う。

・OutPort

名称	型	説明
contoursData	TimedShortSeq	輪郭検出によって得た輪郭点群を出力するポート 輪郭点群は、contour1 として用いる vect<Point>の データである Point p1,p2,p3,p4 を - contour1.size() [0] - contour1.p1.x [1] - contour1.p1.y [2] - contour1.p2.x [3] - contour1.p2.y [4] - contour1.p3.x [5] - contour1.p3.y [6] - contour1.p4.x [7] - contour1.p4.y [8] として配列に格納したデータである。 [0]~[8]で一つの図形が表されている。 受け取り側は、[0]にあたるデータ数分の Point が 来ると認識し、その Point 数分格納したら次の vector の処理へ進む。

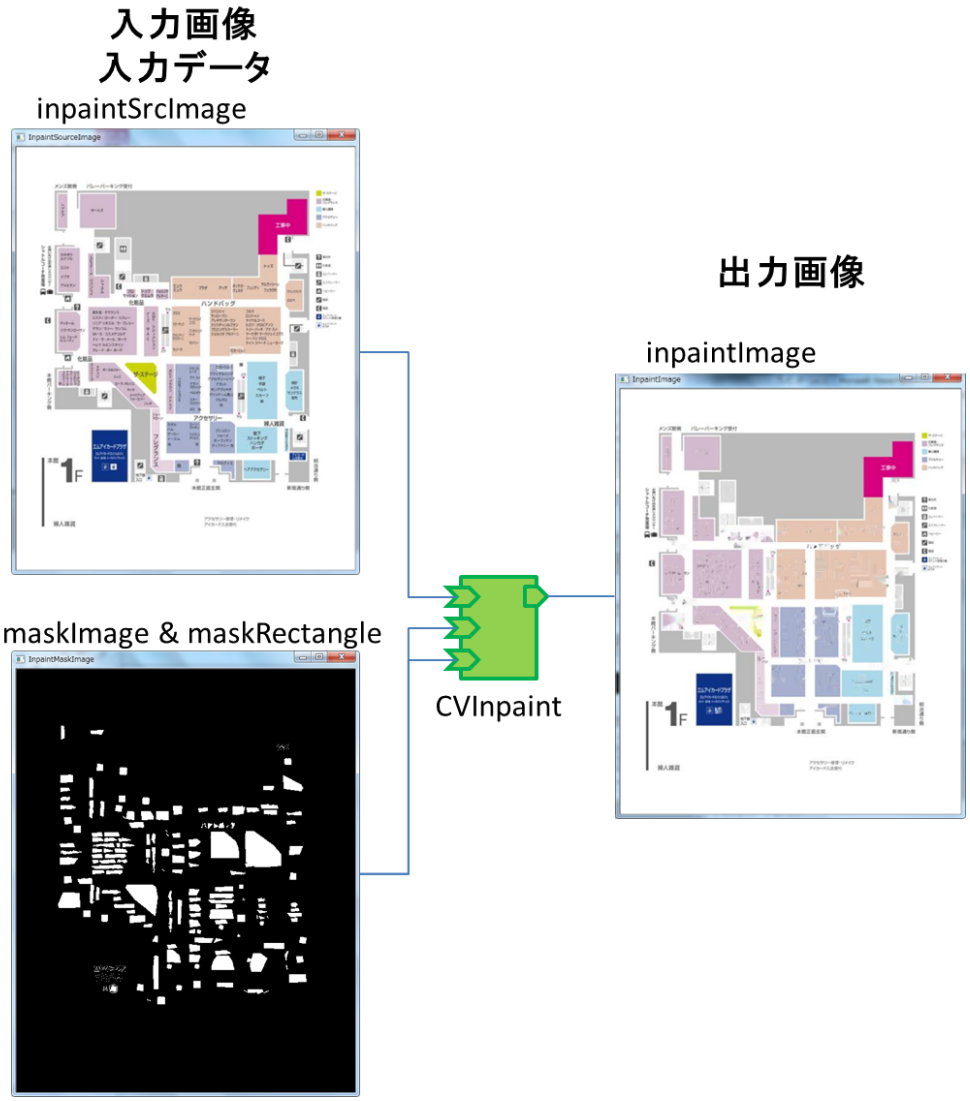
contoursRectangles	TimedShortSeq	輪郭検出によって得た輪郭点群からなる、長方形のデータを出力するポート。 長方形は Rectangle r1 の左上の点 p1 の座標と width,height を - r1.p1.x [0] - r1.p1.y [1] - r1.width [2] - r1.height [3] として配列に格納したデータである。 [0]~[3]によって一つの長方形が表されている。
contoursConvex	TimedShortSeq	輪郭検出によって得た輪郭点群からなる、凸図形のデータを出力するポート。 凸図形は、convex1 として用いる vect<Point>のデータである Point p1,p2,p3,p4 を - convex1.size() [0] - convex1.p1.x [1] - convex1.p1.y [2] - convex1.p2.x [3] - convex1.p2.y [4] - convex1.p3.x [5] - convex1.p3.y [6] - convex1.p4.x [7] - convex1.p4.y [8] として配列に格納したデータである。 [0]~[8]で一つの図形が表されている。 受け取り側は、[0]にあたるデータ数分の Point が来ると認識し、その Point 数分格納したら次の vector の処理へ進む。

・Configuration

名称	型	Widget	説明
ImageView	string	radio	画像表示の ON/OFF を選択するための変数。 制約条件：ON,OFF デフォルト：OFF
FindContour Mode	string	radio	輪郭検出のモードを選択するための変数。 制約条件：EXTERNAL,LIST,CCOMP,TREE

			デフォルト：EXTERNAL
	制約説明	EXTERNAL	最も外側の輪郭のみを抽出する
		LIST	すべての輪郭を抽出し、リストに保存する
		CCOMP	すべての輪郭を抽出し、 それらを 2 階層構造として保存する *上のレベルには，連結成分の外側の境界線が，下のレベルには，連結成分の内側に存在する穴の境界線が属する
		TREE	すべての輪郭を抽出し、入れ子構造になった輪郭を完全に表現する階層構造を構成する
FindContour Method	string	radio	輪郭検出のメソッドを選択するための変数。 制約条件：NONE,SIMPLE,TC89_L1, TC89_KCOS,LINK_RUNS デフォルト：NON
	制約説明	NONE	チェーンコードの全ての点を、 通常の点群に変換する
		SIMPLE	水平・垂直・斜めの線分を圧縮し、 それらの端点のみを残す処理を行う
		TC89_L1	チェーン近似アルゴリズムを適用する
		TC89_KCOS	チェーン近似アルゴリズムを適用する
		LINK_RUNS	値が 1 のセグメントを水平方向に探索し、 接続する輪郭を抽出するアルゴリズムを適用する *CV_LINK_RUNS は CVRETR_LIST が選択されていなければ使うことができないため、モードが LIST でない場合は、CODE を用いるように変更される
offset	vector <int>	text	オプションのオフセット。 各輪郭点はこの値の分だけシフトする。 デフォルト：0,0

2.6 選択領域修復コンポーネント(CVInpaint)



CVInpaint は、入力された画像に対して、コンフィギュレーションパラメータで与えられた値をから得られる領域に対して修復処理を行い、修復された画像を OutPort の inpaintImage から出力するコンポーネントです。

InPort は、修復処理の対象画像を取得する inpaintSrcImage と、修復マスク画像を取得する maskImage と、修復領域を示す長方形のデータを取得する maskRectangle の 3 つがあります。

・InPort

名称	型	説明
inpaintSrcImage	CameraImage	画像データを受け取るポート。

		この画像対して画像修復処理が行われる。 修復の基になる領域は、InPort の maskImage、maskRectangle または コンフィギュレーションパラメータより選択する。
maskImage	CameraImage	修復マスク画像を受け取るポート。
maskRectangle	TimedShortSeq	修復領域を示す長方形のデータ。 左上の x 座標，左上の y 座標，横幅，縦幅を持つ配列。

・OutPort

名称	型	説明
inpaintImage	CameraImage	修復処理後の画像を出力するポート。

・Configuration

名称	型	Widget	説明
ImageView	string	radio	画像表示の ON/OFF を選択するための変数。 制約条件：ON,OFF デフォルト：OFF
InpaintSpace	vector <long>	text	修復したい領域のパラメータ。 矩形修復領域の左上の x 座標，左上の y 座標，横幅，縦幅を持つ long の配列。 制約条件：x[2] >= 0, x[3] >= 0 デフォルト：0,0,0,0
InpaintRadius	double	text	修復される各点を中心とする近傍円形領域半径。 制約条件：x[0] >= 0, x[1] >= 0 デフォルト：0,0
InpaintFlags	string	radio	inpaint の修復手法。 制約条件：INPAINT_NS,INPAINT_TELEA デフォルト：INPAINT_NS
	制約説明	INPAINT_NS	ナビエ・ストークスベース手法
		INPAINT_ TELEA	Alexandru Telea による手法。

3. 地図モデル生成のためのツールコンポーネント群の説明

3.1 複数画像ファイル送信コンポーネント(OpenPicts)

OpenPicts は、読み込んだ画像ファイルを CameraImage 型に変換し OutPort の sendCameraImage から出力するコンポーネントです。読み込む画像ファイルは、コンフィギュレーションパラメータの FileName、又は入力ポート FileName 指定できます。また、InPort の backCameraImage は送り先からのフィードバック処理を受けるためのポート、OutPort の sendFlagData は画像の送信終了・送信のやり直しなどの状態を伝えるためのポートとなっており、フィードバック処理を模した構造となっています。これにより、送り先のコンポーネントと相互通信を模したやり取りを行うことができ、複数枚の画像を CameraImage 型で送ることができます。

フィードバック処理を適切に行うため、OutPort の sendCameraImage のデータ送信ポリシーは、New を選択することを推奨しています。

・InPort

名称	型	説明
FileName	TimedString	読み込む画像データのファイル名を取得するポート。 ","で区切ることで複数取得可能。
backCameraImage	CameraImage	送り先から受け取った画像データを取得するポート。 このポートの CameraImage と OutPor の sendCameraImage から出力される CameraImage が合致した場合、次画像へのフラグ処理へ進む。

・OutPort

名称	型	説明
sendCameraImage	CameraImage	読み込んだ画像データを出力するポート。 データ送信ポリシー：New 推奨。
sendFlagData	TimedShort	送り先へ現在の送信状態を出力するポート。 <フラグ処理の説明> -2:ファイルの元データが書き換わったことを示すフラグ（要初期化）。 -1:すべての画像を送信したことを示すフラグ。

・Configuration

名称	型	デフォルト値	説明
01_ImageView	string	OFF	画像表示を選択するための変数。

			radio 型で宣言されており、制約条件は(ON,OFF)。
02_FileName	string	*.jpg	読み込みたいファイルのアドレス。 ","で区切ることで複数取得可能。 text 型で宣言されている。

3.2 輪郭情報を用いた画像修復コンポーネント(imageInpaint)

imageInpaint は、入力ポートから受け取った輪郭点データ群の情報を用いて画像修復を行うコンポーネントです。受け取った輪郭点データ群に対し、UI コンポーネント（controlPanel）を用いて削除等の操作を行うことができます。また、修復に対しても、UI コンポーネント（controlPanel）でのドラッグ入力を用いることで、処理に用いるマスクイメージを作り、修復処理を行うコンポーネント（CVInpaint）に出力することができます。これにより、容易に指定領域の画像修復を行うことができます。

InPort には修復処理を行いたい画像データを取得する srcImage、輪郭点群がベースとなっている輪郭長方形データ群を取得する contoursRectangles、同じく輪郭点群がベースとなっている輪郭凸図形データ群を取得する contoursConvex の 3 つのポートに加え、UI コンポーネント（controlPanel）のクリック情報を取得する clickPoint、ドラッグによって生成された長方形情報を取得する draggedRect、画像情報送信に用いる一時保存領域を指定する tempFolderPath、修復処理の実行結果となる画像を取得する inpaintImage の計 7 つがあります。contoursRectangles と contoursConvex は輪郭検出コンポーネント（CVFindContour）と、clickPoint と draggedRect は UI コンポーネント（controlPanel）コンポーネントとの接続が前提とされています。

OutPort は、UI コンポーネントへ表示する画像及び画像の保存アドレスを出力する modifyImage、modifyImagePath、選択領域修復コンポーネント（CVInpaint）へ修復に用いる情報を送るための inpaintSrcImage、inpaintMaskImage、 inpaintMaskArea、全ての処理が終わった結果の画像を送るための processedImage の計 6 つがあります。

・InPort

名称	型	説明
srcImage	CameraImage	修復処理を行う画像データを受け取るポート。 この画像は修復処理以外にも、UI コンポーネント(controlPanel)上に表示される画像としても用いる。
contoursRectangles	TimedShortSeq	輪郭検出によって得た輪郭点群からなる長方形のデータを取得するポート。 長方形は Rectangle r1 の左上の点 p1 の座標と width,height を

		- r1.p1.x [0] - r1.p1.y [1] - r1.width [2] - r1.height [3] として配列に格納したデータである。 [0]~[3]によって一つの長方形が表されている。
contoursConvex	TimedShortSeq	輪郭検出によって得た輪郭点群からなる凸図形のデータを出力するポート。 凸図形は convex1 として用いる vect<Point>のデータである Point p1,p2,p3,p4 を - convex1.size() [0] - convex1.p1.x [1] - convex1.p1.y [2] - convex1.p2.x [3] - convex1.p2.y [4] - convex1.p3.x [5] - convex1.p3.y [6] - convex1.p4.x [7] - convex1.p4.y [8] として配列に格納したデータである。 [0]~[8]で一つの図形が表されている。 受け取り側は、[0]にあたるデータ数分のPointが来ると認識し、そのPoint数分格納したら次のvector処理へ進む。
tempFolderPath	TimedString	大きい画像データを送受信する場合に用いる画像パスの保管場所。
clickPoint	TimedPoint3D	UI コンポーネント(controlPanel)上でクリックされた座標を取得するポート。 データは point.x：controlPanel 上の x 座標 point.y：controlPanel 上の y 座標 point.z：操作に対するフラグ情報 を意味する。
draggedRect	TimedShortSeq	UI コンポーネント(controlPanel)上でドラッグ指定した長方形情報を取得するポート。 データは左上の x 座標、左上の y 座標、横幅,縦幅を

		持つ配列となっている。
inpaintImage	CameraImage	修復処理が行われた画像を受け取るポート。 CVInpaint の OutPort に繋がることを前提としている。

・OutPort

名称	型	説明
modifyImage	CameraImage	ユーザの修正処理のために UI コンポーネント(controlPanel)上で表示する画像を出力するポート。 抽出領域が書き込まれた画像や修復処理が終わった後の画像が出力される。
modifyImagePath	TimedString	ユーザの修正処理のために UI コンポーネント(controlPanel)上で表示する画像のパスを出力するポート。
inpaintSrcImage	CameraImage	修復処理に用いる画像を出力するポート。 CVInpaint の InPort: : inpaintSrcImage に繋がることを前提としている。
inpaintMaskImage	CameraImage	修復処理に用いるマスク画像を出力するポート。 CVInpaint の InPort: : maskImage に繋がることを前提としている。 （同タイミングで長方形座標データを送った場合こちらからの情報が優先して処理される）
inpaintMaskArea	TimedShortSeq	マスク画像作成に用いる座標群を出力するポート。 CVInpaint の InPort: : maskRectangle に繋がることを前提としている。
processedImage	CameraImage	すべての処理が終わった画像を出力するポート。

・Configuration

名称	型	デフォルト値	説明
01_contoursType	string	Rectangle	処理に用いる輪郭データタイプを選択する。 長方形と凸図形の両方の形式のデータをポートから受け取っている場合に用いる。 Rectangle：長方形データ型の輪郭データ。 Convex：凸図形データ型の輪郭データ。

			radio 型で宣言されており、制約条件は (Rectangle, Convex)。
02_MaskDataSelect	string	Image	CVInpaint へ送るマスクの情報を選択する。 Image：マスク画像を作成して inpaintMaskImage から CVInpaint へ出力する。 Rectangle：マスク領域を inpaintMaskArea から CVInpaint へ出力する。 radio 型で宣言されており、制約条件は (Image, Rectangle)。
03_MaskTypeSelect	double	Single	02_MaskDataSelect で maskImage を送る場合、全輪郭点データを一度に処理するかどうかを選択する。 Single：単体のデータを送る All：すべてのデータを送る radio 型で宣言されており、制約条件は (Single, All)。

3.3 ラインマップ変換コンポーネント(convToLineMap)

convToLineMap は、InPort から得られるデータ群を用いて、ラインマップに変換し、OutPort の compLineMap から出力するコンポーネントです。ここでのラインマップは「地図の頂点を示す座標群と、その座標同士の関係性」を持つ点と線の情報群を指します。

InPort の contoursData、clickPoint、draggedRect、featurePoints からは、それぞれ輪郭点データ群、クリックされた座標データ、ドラッグ指定した長方形情報、特徴点検出で得たデータを取得し、これらを基にラインマップへの変換を行います。

また、UI コンポーネント(controlPanel）上に OutPort の modifyImage と modifyImagePath から出力した画像を表示し、それを見て修正を行うことができます。

・InPort

名称	型	説明
tempFolderPath	TimedString	画像情報送信に用いる一時作業領域のパスを受け取るためのポート。
srcImage	CameraImage	修復処理を行う画像データを受け取るポート。

		この画像に対し修復処理が行われる。 また、UI コンポーネント(controlPanel）上で表示される画像もこの画像が用いられる。
contoursData	TimedShortSeq	輪郭点データ群を取得するポート。 contour1 として用いる vect<Point>のデータである Point p1,p2,p3,p4 を - convex1.size() [0] - convex1.p1.x [1] - convex1.p1.y [2] - convex1.p2.x [3] - convex1.p2.y [4] - convex1.p3.x [5] - convex1.p3.y [6] - convex1.p4.x [7] - convex1.p4.y [8] として配列に格納したデータである。 [0]~[8]で一つの図形が表されている。 受け取り側は、[0]にあたるデータ数分の Point が来ると認識し、その Point 数分格納したら次の vector 処理へ進む。
clickPoint	TimedPoint3D	UI コンポーネント(controlPanel)上でクリックされた座標を取得するポート。 データは point.x：controlPanel 上の x 座標。 point.y：controlPanel 上の y 座標。 point.z：操作に対するフラグ情報。 を意味する。
draggedRect	TimedShortSeq	UI コンポーネント(controlPanel)上でドラッグ指定した長方形情報を取得するポート。 データは左上の x 座標、左上の y 座標、横幅,縦幅を持つ配列となっている。
featurePoints	TimedFloatSeq	特徴点検出で得た KeyPoint のデータを取得するポート。 KeyPoint 型のデータを､以下のように分解して出力する。 KeyPoint - float angle [0] - int class_id [1]

	- int octave [2] - Point pt (pt.x [3] pt.y [4]) - float response [5] - float size [6] これら 7 個のデータがそれぞれ配列の一要素として用いられている。

・OutPort

名称	型	説明
modifyImage	CameraImage	UI コンポーネント(controlPanel)上で表示する画像を出力するポート。 抽出領域を書き込んだ画像や修復処理後の画像を出力する。
modifyImagePath	TimedString	UI コンポーネント(controlPanel)上で表示する画像（modifyImage）のパスを出力するポート。
compLineMap	TimedShortSeq	変換が完了した地図のライン情報を取得するポート。 ライン情報は convex1 として用いる vect<Point>のデータである Point p1,p2,p3,p4 を - convex1.size() [0] - convex1.p1.x [1] - convex1.p1.y [2] - convex1.p2.x [3] - convex1.p2.y [4] - convex1.p3.x [5] - convex1.p3.y [6] - convex1.p4.x [7] - convex1.p4.y [8] として配列に格納したデータである。 [0]~[8]で一つの図形が表されている。 受け取り側は、[0]にあたるデータ数分の Point が来ると認識し、その Point 数分格納したら次の vector 処理へ進む。

・Configuration

名称	型	デフォルト値	説明
modifyType	string	erase	輪郭データ群に対するユーザからの操

	作の種類を選択する。 erase : クリックされた一番近い凸図形頂点を、輪郭データ群から消す add : クリック操作によって隣り合う凸図形頂点を選択してもらい、その二つの頂点の間に新たに頂点を追加する追加する頂点は特徴点の中から選択する radio 型で宣言されており、制約条件は(erase,add)。

3.4 地図モデル生成管理コンポーネント(mapsManager)

mapsManager は、地図画像から地図モデルへの変換を管理し、結果の保存を行うコンポーネントです。

何枚目の画像に対し地図モデル生成を行うか、処理アルゴリズムが無事完了したかなど、UI コンポーネント(controlPanel)を介してユーザの指示を受け付けることができます。

複数画像ファイル送信コンポーネント(OpenPicts)から画像を取得し、1 枚目から順に地図モデル化処理を行う。OutPort の backCamImg から出力した画像と処理が完了した画像の TimeStamp を比較し、同一なら UI コンポーネント(controlPanel）上に処理前と処理後の画像を表示する。InPort の receiveFlagData で UI コンポーネント(controlPanel）からの完了指示を取得したら来たら次の画像、もう一度の指令が受け渡されたら、もう一度同じ画像に処理を行う。最終処理でラインマップが完成したと判断されたら、コンフィギュレーションパラメータの 03_SaveSpace で指定された場所へ OutPort の lineMapsPath から全座標群情報を、compLineImg から描画イメージを出力する。

・InPort

名称	型	説明
receiveCamImg	CameraImage	地図モデル生成に用いる画像データを取得するポート。
receiveFlagData	TimedShort	送り先コンポーネントの画像データ送信状況を取得するポート。 <フラグ処理の説明> -2：受け取り状態の初期化のフラグ。 -1：全画像データ送信済みのフラグ。
tempFloderPath	TimedString	一時作業領域のパスを取得するポート。

		大きい画像データを入出力するときに用いる。
clickPoint	TimedPoint3D	UI コンポーネント(controlPanel)上でクリックされた座標を取得するポート。 データは point.x：controlPanel 上の x 座標。 point.y：controlPanel 上の y 座標。 point.z：操作に対するフラグ情報。 を意味する。
compNormalizedMap	CameraImage	正規化された地図画像データを取得するポート。
compLineMap	TimedShortSeq	変換されたラインマップの座標群を取得するポート。 ラインマップを描画する際に用いる。 データは vector<vector<Point>>型の地図座標群で、[0][1][2][3][4]…　に対し[vector<Point>サイズ n*2][p1.x][p1.y][p2.x][p2.y]…[pn.x][pn.y][vector<Point>のサイズ m*2]… という要素を持つ。

・OutPort

名称	型	説明
backCamImg	CameraImage	送り先から受け取った地図画像を出力するポート。 フィードバッグ処理を疑似的に再現するために用いられ、送り先側でデータの一致を確認することで次画像が送られる。
tempFolderPathOut	TimedString	一時作業領域のパスを出力するポート。 大きい画像データを入出力するときに用いる。
makeNormSrcImg	CameraImage	正規化処理を行う画像データを出力するポート。
makeNormSrcImgPath	TimedString	正規化処理を行う画像データのパスを出力するポート。
compNormImg	CameraImage	正規化処理が完了した画像データを出力するポート。
compNormImgPath	TimedString	正規化処理が完了した画像データのパスを出力するポート。
makeLineSrcImg	CameraImage	ラインマップに変換する正規化処理が完了した画像を出力するポート。
makeLineSrcImgPath	TimedString	ラインマップに変換する正規化処理が完了した画

		像のパスを出力するポート。
compLineImg	CameraImage	ラインマップを描画した画像を出力するポート。
compLineImgPath	TimedString	ラインマップを描画した画像のパスを出力するポート。
lineMaps	TimedShortSeq	全地図画像のラインマップ情報を出力するポート。 全地図画像のラインマップは vector<vector<vector<Point>> map 型で、これらを一括で出力するために - map1.size() [0] - map1.floor1.size() [1] - map1.floor1.conv1.size() [2] - map1.floor1.conv1.p1.x [3] ・・・ - map1.floor1.conv1.p3.y [8] - map1.floor1.conv2.size() [9] - map1.floor1.conv2.p1.x [10] ・・・ - map1.floor1.conv2.p3.y [15] - map1.floor2.size() [16] - map1.floor2.conv1.size() [17] - map1.floor2.conv1.p1.x [18] ・・・ - map1.floor2.conv2.p3.y [30] のように 30 の要素を配列に格納して出力する。 [0]~[15]で一つのラインマップを表している。
lineMapsPath	TimedString	全地図画像のラインマップ情報のパスを出力するポート。

・Configuration

名称	型	デフォルト値	説明
01_ImageView	string	OFF	画像表示を選択するための変数。 radio 型で宣言されており、制約条件は(ON,OFF)。
02_SelectPictNum	int	0	画像処理及び線画化処理を行う画像の番号を指定できる。 先のステップから戻ることも可能。

			text 型で宣言されており、制約条件は x>=0 かつ x<=最大枚数。
03_SaveSpace	string	C:¥¥tmp¥¥maps ¥¥linemaps	完成したラインマップの座標情報群と描画画像を保存する先を指定する。 text 型で宣言されている。

3.5 UI(User Interface）コンポーネント(controlPanel)

controlPanel は、地図モデル生成中に必要となるユーザからの操作を取得するコンポーネントです。InPort の srcImage、srcImagePath から処理前の画像を取得、modifyImage、modifyImagePath から処理中の画像を取得、completeImage、completeImagePath から処理が完了した画像を取得します。ここで受け取った処理前、処理中、処理後の画像をウィンドウに表示します。

また、このウィンドウではユーザからの操作を取得することができ、所要機能としては、読み込み画像の指定・一時作業領域の指定・クリック情報の取得・ドラッグ情報の取得・ボタンによるイベント情報(決定操作やリセット操作など)の取得（情報形態はクリック情報と同一）があります。処理の状態ににあったイベントを提示し、その選択情報を OutPort から出力します。

・InPort

名称	型	説明
srcImage	CameraImage	処理前の画像を取得するポート。
srcImagePath	TimedString	処理前の画像のパスを取得するポート。
modifyImage	CameraImage	ユーザの操作を必要とする処理中の画像を取得するポート。
modifyImagePath	TimedString	ユーザの操作を必要とする処理中の画像のパスを取得するポート。
completeImage	CameraImage	処理が完了した画像を取得するポート。
completeImagePath	TimedString	処理が完了した画像のパスを取得するポート。

・OutPort

名称	型	説明
openFilePath	TimedString	生成されたウィンドウ上のメニューで指定した、読み込む画像のパスを出力するポート。
tempFolderPath	TimedString	生成されたウィンドウ上のメニューで指定した、一時作業領域のパスを出力するポート。
clickPoint	TimedPoint3D	生成されたウィンドウ上でクリックした座標のデ

		ータを出力するポート。
dragRectangle	TimedShortSeq	生成されたウィンドウ上でドラッグされた領域のデータを出力するポート。

4. コンポーネント群の使用方法 (2014/12/2 更新)

4.1 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

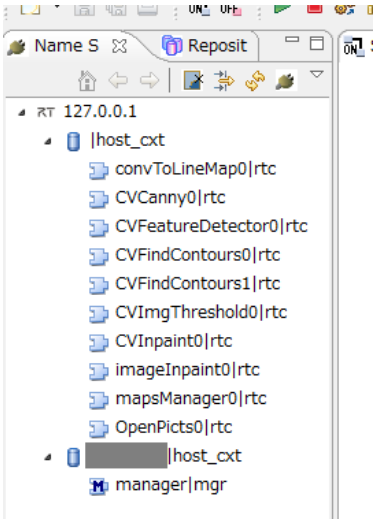
これらの手順はどのようなシステムでも共通です。RTSystemEditor の詳しい操作方法は OpenRTM-aist のホームページ（http://www.openrtm.org/）を参照してください。以下の節では手順 5 に関して説明します。

4.2 rtshell を用いたシステム構築

屋内地図モデルを生成するにあたり、大まかな流れは以下のようになります。

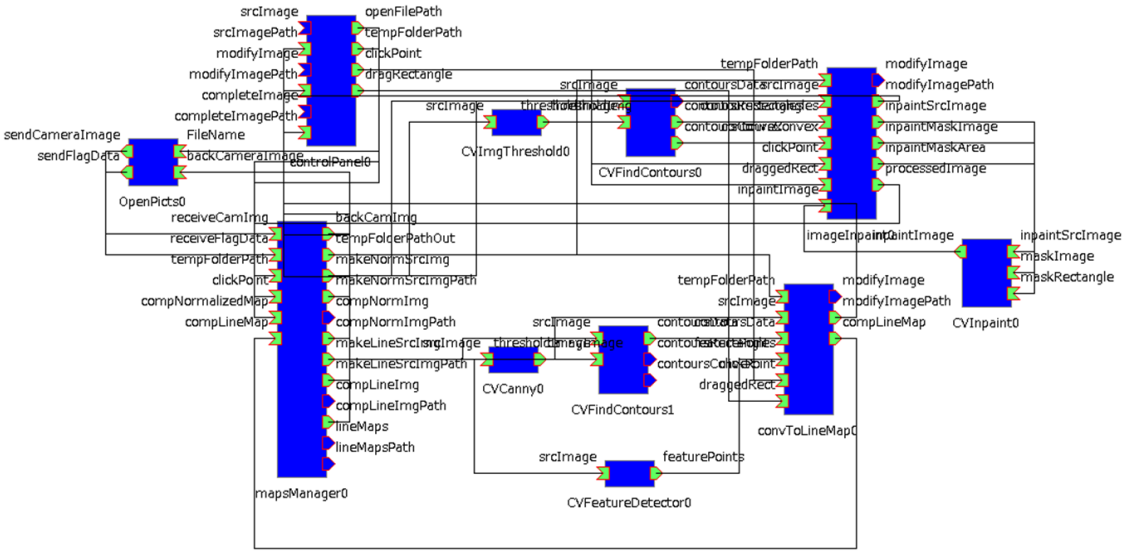
① 地図の画像を読み込む
② 地図画像からモデル生成に余計な情報を削除し正規化する
③ 正規化地図画像から地図の線情報を取得する

そのため、複数の画像処理のコンポーネントと複数のツールコンポーネントを起動・接続する必要があります。この作業を簡略化するため、rtshell によって管理を行っています。まず、コンポーネントを用意します。本稿のコンポーネントはすべて GitHub（https://github.com/Ma13055/MapModelGenerator）にありますので、そちらからダウンロードしてください。MakeLineMaps 内にある mapsManagerComp.exe を実行します。この前にネームサーバの接続等は行っておいてください。実行されるとネームサービスビューが次のようになります。



その後 controlpanelJar 内にある controlPanel.bat を実行します。これにより controlPanel が実行されます。これで起動は終了です。

次は接続を行います。MakeLineMaps 内にある makeMapsModelConect.bat を実行します。するとコンソールが立ち上がり、接続が開始されます。接続が完了したらコンソールが自動的に落ちるため、完了したらコンポーネントのイニシャライズを行ってください。配置が完了した例が以下のようになります。



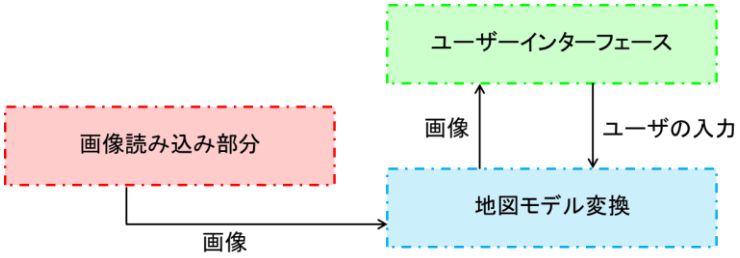
これでシステムの構築は完了です。次は今回構築したシステムの構成についての説明を行います。

4.3 システムの構成と接続

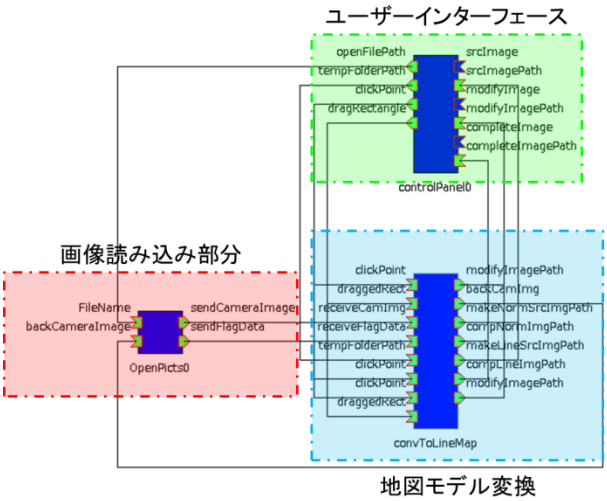
本システムは SysML を用いて設計を行っています。SysML の図は

（https://github.com/Ma13055/MapModelGenerator）で公開しているため、そちらも参照してください。本章では概要のみの説明を行います。

本システムの大まかな構成は以下の通りです。



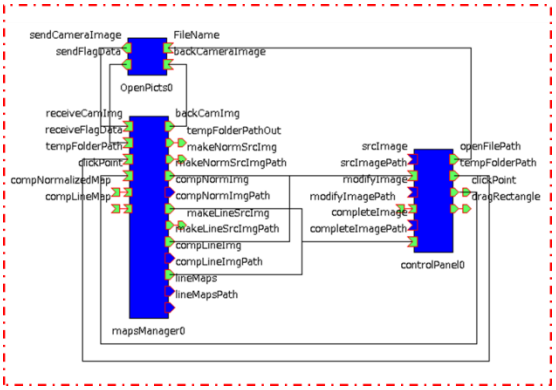
これを複合コンポーネントであらわすと以下のようになります。



さらに順に見ていきます。まず、画像読み込み部分にあたるコンポーネントは

- OpenPicts
- mapsManagers
- controllPanel

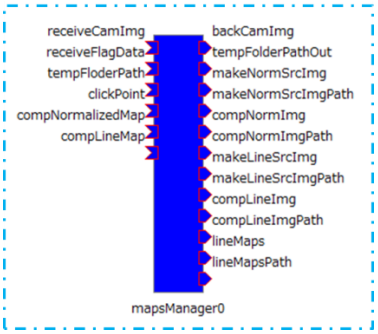
の 3 つがあります。この接続例が以下のようになります。



次に、地図モデル変換の部分です。地図モデルの変換をさらに分解すると

・画像管理
・正規マップ変換
　…与えられた画像に対し正規化処理を行い、正規地図画像を返す
・ラインマップ変換
　…与えられた画像をラインマップ（座標群）へ変換し、
　　ラインマップ（座標群）を返す

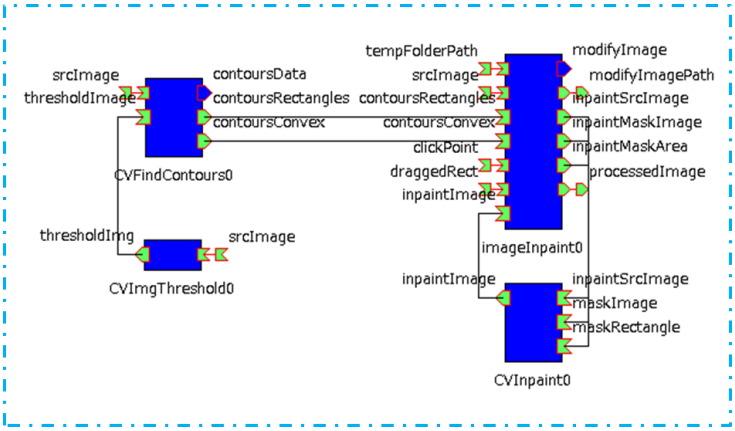
の 3 つとなります。コンポーネントの内、画像管理を行っているのが mapsManager となります。



正規マップ変換を行っているコンポーネントは

・imageInpaint
・CVInpaint
・CVFindContour
・CVImgThreshold

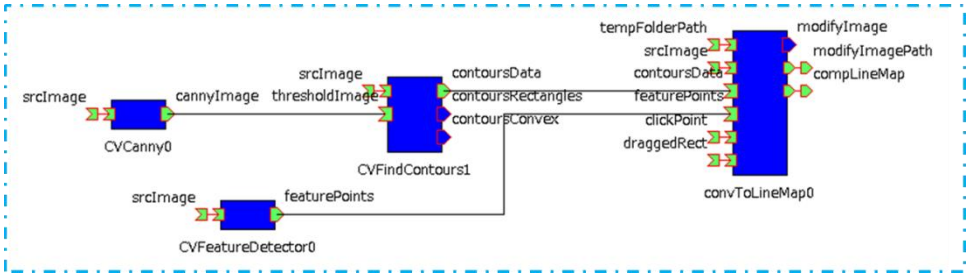
となり、これらの接続図が以下のようになります。



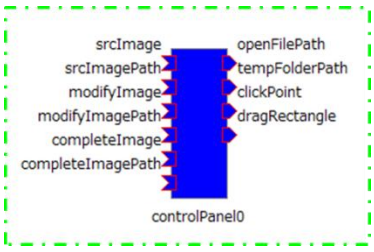
最後にラインマップ変換を行っているコンポーネントは

・CVCanny
・CVFindContour
・CVFeatureDetector

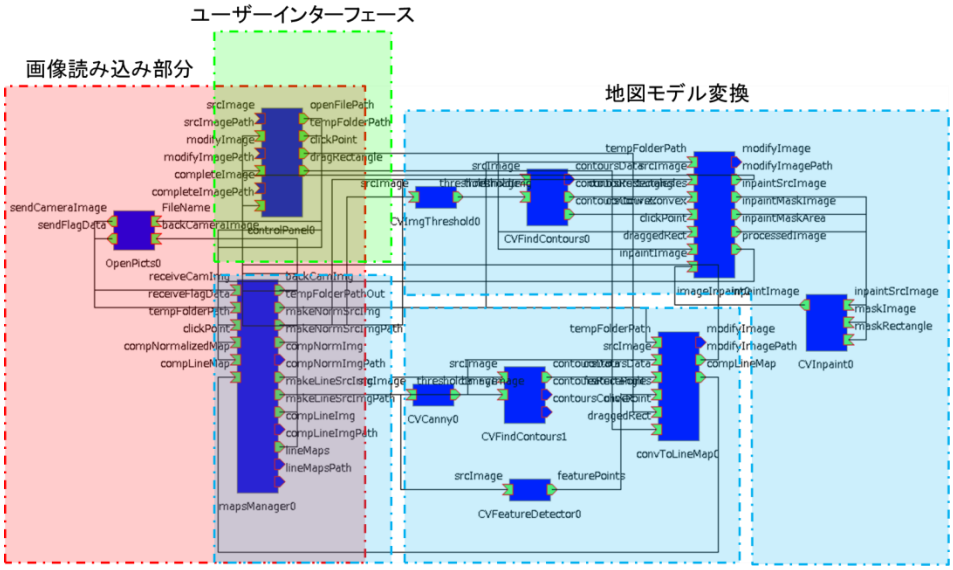
となり、これらの接続図が以下のようになります。



ユーザーインターフェースの部分のコンポーネントは controllPanel となります。



以上の接続図を全部合わせると以下のようになります。



また、本システムは地図モデル変換の部分を OpenCV による画像処理コンポーネント群によって実現しています。したがって、地図モデル変換の部分を、地図モデルにしたい地図画像に応じて変更することで、多様な地図に対応できると考えております。

システムの構成については以上です。次にシステムの流れとそれに合わせて必要な Configuration の設定方法について説明します。

4.4 システムの流れ

今回のシステムの大まかな流れは以下のようになります。

① 地図の画像を読み込む
② 地図画像からモデル生成に余計な情報を削除し正規化する
③ 正規化地図画像から地図の線情報を取得する。

これらの工程に対し、ユーザが行う作業は以下のようになります。

① 読み込む地図画像を指定する。
② 地図画像からモデル生成に余計な情報を削除する。
③ 地図画像から抽出された輪郭データに対し修正を行う。

本章では、ユーザが行う手順を追ってシステムの流れを説明します。

4.4.1 手順①：地図の画像を読み込む

起動したコンポーネントすべてアクティブにすると、controlPanel の UserInterface ウィンドウが立ち上がります。このウィンドウのメニューバーにある File(F) -> FileOpen(O) を選択すると、ファイルマネージャが立ち上がるので、適当な地図画像を選択し(複数選択

可)画像を読み込んでください。

読み込まれた画像は、MapManager の makeNormSrcImg と makeNormSrcImgPath から出力されます。コンソール画面に「SendFIN」と表示されたら読み込みは完了です。これで手順が終了となります。

送信可能な画像サイズは、最大 2MB となっており、それを超えて送信をしようとするとポートの接続が切れてしまいます。この場合、画像ファイル自体はオープンできても、送信はできていませんので注意してください。

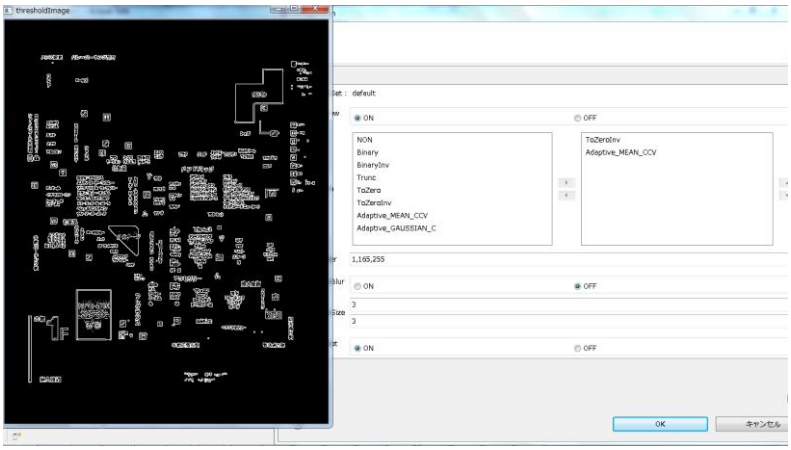
この先の手順において、作業中の画像はこの controlPanel のウィンドウに表示されます。作業前や処理後の画像を確認したいときは、ウィンドウ右上のボタンで表示を切り替えてください。また、処理の内容自体を確認したいときは、各画像処理コンポーネントに画像を表示するための Configuration があるため、そちらで画像表示を ON にしてご利用ください。

4.4.2 手順②：地図画像からモデル生成に余計な情報を削除する

画像の読み込みが完了すると controlPanel のウィンドウにグレースケールになった地図画像に輪郭データが上書きされた図が表示されます。

この輪郭データは削除する領域となります。もしも地図に必要な情報（例：壁となってほしい部分）が輪郭とされていた場合、その輪郭部分をクリックまたはドラッグすることで選択してください。この時コンソールには選択された輪郭の座標が表示されます。これにより選択した輪郭を消すことができ、地図上に残すことができます。また、そもそも輪

郭データが望む形で得られない場合は、CVImgThreshold の閾値設定で調整を行ってください。CVImgThreshold の ImageView を ON にすることで以下のように画像が表示されます。



この時、削除したい部分（例：文字）のみが白く表示されるように調節すると、スムーズに処理が行えます。

また、間違えて削除したい場所を選択してしまったときは、controlPanel のウィンドウ左上の「戻す」を選択してください。これにより図を一つ前の状態に戻すことができます。この作業中で使用できる controlPanel のウィンドウ左上のボタンの機能は以下の通りです。

戻る　　：図を一つ前の状態に戻す。
リセット：図をはじめの状態に戻す。
一時保存：作業の保存。この操作後「リセット」を押すと、この時の状態に戻る。
修正完了：作業を完了したとき選択する。

輪郭データの修正が完了したら、controlPanel のウィンドウ左上の「修正完了」を押してください。囲われていた輪郭データの部分が、すべて修復処理されます。この処理は自動で行われ、完了すると輪郭データで囲まれていた文字が消えた状態の図が表示されます。この図に対して、ユーザはドラッグ操作を行うことができます。ドラッグ操作によって囲まれた部分に修正処理がかかるため、先ほどの処理で残ってしまった文字などを消すことができます。

図

この修正作業が完了したら再度「修正完了」を押してください。これで、この図に対する手順②が終了します。これにより、controlPanel のウィンドウ中央上のボタンが使用できるようになります。機能は以下の通りです。

再処理　　：現在の図にもう一度処理をかける。
次画像へ　：次の図を読み込み、処理を開始する。
全処理完了*：全ての図に対して処理が完了したとき選択する。
*複数の画像が読み込まれているときは、
すべての画像を一度開かないと「全処理完了」は押せません。

すべての画像に対し処理が終了したら、「全処理完了」を押してください。これで手順②が完了です。

4.4.3 手順③：地図画像から抽出された輪郭データに対し修正を行う

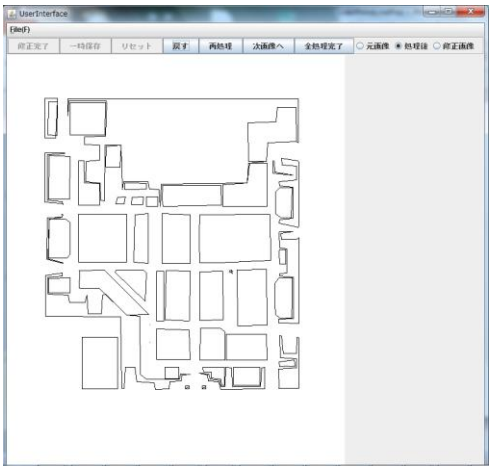
手順②が完了すると、ラインマップのもととなる輪郭データが画像に出力されます。

図

この画像に対し、輪郭データの頂点に対する修正を行います。convToLineMaps のコンフィギュレーションパラメータで操作のモードを切り替えることができます。操作の種類は以下の通りです。

defrag ：同じ頂点で囲まれている別の領域に変更する。
eraseFigure ：ドラッグされた範囲にある図形を削除する。
erasePoint ：頂点を削除する。
add ：2 つの頂点の間にある特徴点を頂点にする。（頂点を追加する。）
movePoint ： 頂点を別の特徴点に移動する。

輪郭データの妥当性はその輪郭に囲まれている領域を塗りつぶして表示されるため、地図の中で囲まれている箇所（例：部屋、店舗の領域）が正しく塗りつぶされているかどうかで判断してください。この作業が完了したら controlPanel のウィンドウ左上の「修正完了」を押してください。これでこの図に対する手順③が終了します。複数の画像を読み込んでいる場合は「次画像へ」を選択し、他の画像に対してもこの処理を終了してください。読み込んだ全ての画像に対して処理が終了したら「全処理完了」を押してください。これで手順③が完了です。これにより、ラインマップに変換された地図が画像として表示されます。



ラインマップの完成図

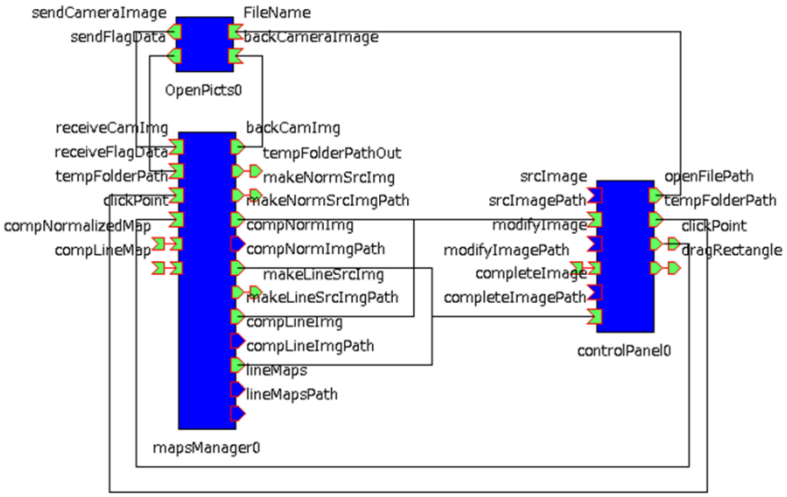
4.5 接続ポート

rtshell を用いず、個別にコンポーネントを繋ぎ、システムを構成することもできます。本章では、接続されているポートについてまとめました。今回の接続例は以前の節と同様です。各自コンポーネントを変更してシステムを構築する際は、これに加え、2 章のコンポーネントの説明を参照してください。

4.5.1 地図画像読み込み

使用するコンポーネントは以下の 3 つです。

・controlPanel
・OpenPict
・mapManager



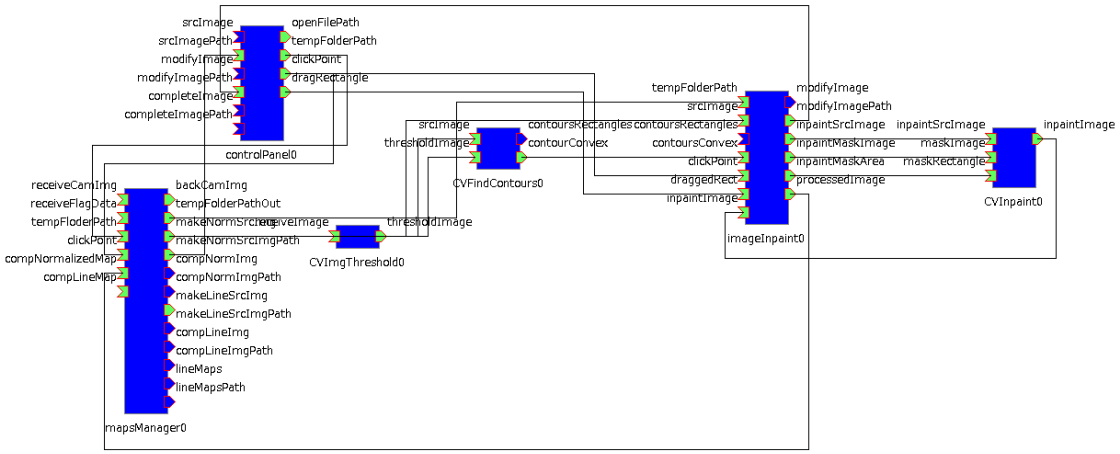
コンポーネント間の接続

InPort		OutPort	
コンポーネント名	Port 名	コンポーネント名	Port 名
OpenPicts	FileName	controlPanel	openFilePath
	backCameraImage	mapsManager	backCamImg
mapsManager	receiveCamImg	OpenPicts	sendCameraImage
	receiveFlagData		sendFlagData
	tempFolderPath	controlPanel	tempFolderPath
controlPanel	srcImagePath	mapsManager	makeNormSrcImgPath

4.2.2 正規マップ変換（地図画像からモデル生成に余計な情報を削除し正規化する）

使用するコンポーネントは以下の 6 つです。

- controlPanel
- mapManager
- imageInpaint
- CVImgThreshold
- CVFindContours
- CVInpaint



コンポーネント間の接続

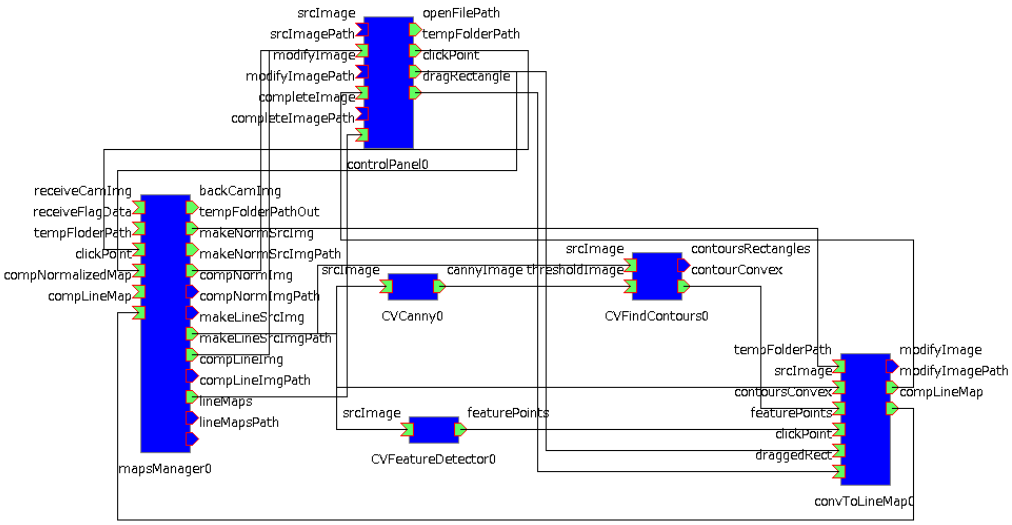
InPort		OutPort	
コンポーネント名	Port 名	コンポーネント名	Port 名
mapsManager	tempFolderPath	controlPanel	tempFolderPath

	clickPoint		clickPoint
	compNormalizedMap	imageInpaint	processedImage
controlPanel	modifyImagePath		modifyImagePath
	srcImagePath	mapsManager	makeNormSrcImgPath
CVImgThreshold	receiveImage		makeNormSrcImg
CVFindContours	srcImage		makeNormSrcImg
	thresholdImage	CVImgThreshold	thresholdImage
CVInpaint	inpaintSrcImage	imageInpaint	inpaintSrcImage
	maskImage		inpaintMaskImage
	maskRectangle		inpaintMaskArea
imageInpaint	tempFolderPath	mapsManager	tempFolderPath
	srcImage		makeNormSrcImg
	contoursRectangles	CVFindContours	contoursRectangles
	contoursConvex		contoursConvex
	clickPoint	controlPanel	clickPoint
	draggedRect		dragRectangle
	inpaintImage	CVInapint	inpaintImage

4.2.3 ラインマップ変換（正規化地図画像から地図の線情報を取得しラインマップへ変換）

使用するコンポーネントは以下の 6 つです。

- controlPanel
- mapManager
- convToLineMap
- CVCanny
- CVFindContours
- CVFeatureDetector



コンポーネント間の接続

InPort		OutPort	
コンポーネント名	Port 名	コンポーネント名	Port 名
mapsManager	tempFolderPath	controlPanel	tempFolderPath
	clickPoint		clickPoint
	compLineMap	convToLineMap	compLineMap
CVFeatureDetector	srcImage	mapsManager	makeLineSrcImg
CVCanny	srcImage		makeLineSrcImg
CVFindContours	srcImage		makeLineSrcImg
	thresholdImage	CVCanny	cannyImage
convToLineMap	tempFolderPath	mapsManager	tempFolderPath
	srcImage		makeLineSrcImg
	contoursData	CVFindContours	contoursData
	featurePoints	CVFeatureDetector	featurePoints
	clickPoint	controlPanel	clickPoint
	draggedRect		draggedRect
controlPanel	modifyImage	convToLineMap	modifyImagePath
	completeImagePath	mapsManager	compLineImgPath

5. 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
立川　将
Email:y09148@shibaura-it.ac.jp