

OpenVGR

座標系変換ツール説明書

はじめにお読みください

V e r . 0.9.0

2011 年 12 月 21 日

独立行政法人 産業技術総合研究所

本文書の取り扱いについて

- 本書に掲載する情報は、使用者に有用なものであるように万全を期していますが、内容の正確性、最新性、その他一切の事項について保証をするものではありません。
- 本書の著者および著作権者は、使用者がこの文書から得たまたは得られなかった情報から生じる損害に対しても、一切責任を負いません。
- 本書は使用者への事前の予告なしに変更、削除、公開の中止を行うことがあります。

【連絡先】

(独) 産業技術総合研究所 知能システム研究部門 タスクビジョン研究グループ
〒305-8568 茨城県つくば市梅園 1-1-1 中央第二

E-Mail: openvgr-contact@m.aist.go.jp

目次

1. 概要	1
2. 導入方法	1
3. 座標系変換RTコンポーネント例の実行方法	2
4. 座標系変換行列データの作成	5
5. ツールプログラムについて	9
6. 座標系変換RTコンポーネント例の仕様	10

1. 概要

OpenVGR 作業対象認識コンポーネントが出力する認識結果はカメラ座標系の値です。ロボットが対象物を掴んだりするためには、カメラ座標系からロボット座標系への変換が必要になります (図 1)。本書では、この座標系変換を行う RT コンポーネント例と、変換行列データを作成するツールプログラムについて説明します。

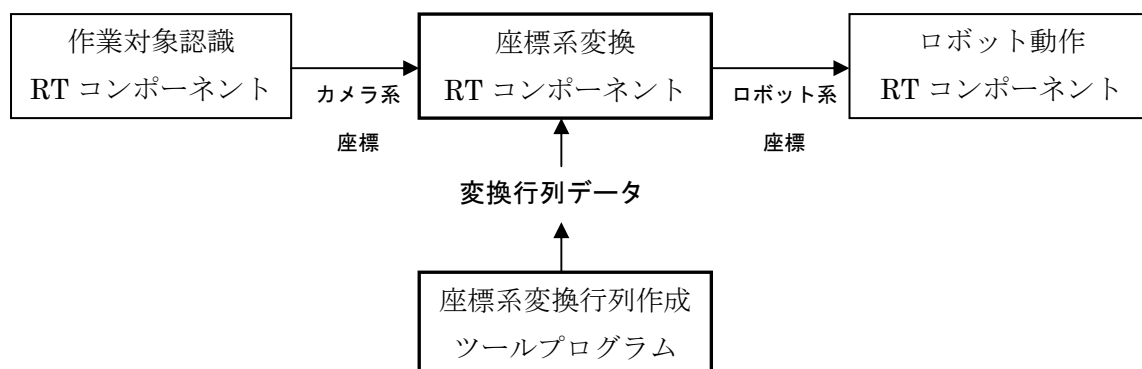


図 1 座標系変換 RT コンポーネントと変換行列作成ツールプログラム

本プログラムは以下の環境で動作します。

動作環境：

PC : CPU Core2Duo 以上, メモリ 1GB 以上, ハードディスク 10GB 以上

OS : Ubuntu 10.04 LTS Desktop 日本語 Remix (32bit 環境)

カメラ : IEEE 1394b カメラ (IIDC 1.31 準拠)

Point Grey Research 社製 Flea2 カメラで動作確認

開発環境：

OpenRTM-aist 1.0.0-RELEASE C++版

OpenRTM Eclipse tools 1.0-RELEASE

GNU Compiler Collection 4.4.3

OpenCV 2.0

libdc1394-2

OpenVGR-0.9.0 以降

2. 導入方法

2.1 プログラムのビルド

以下のように **OpenVGRextra-バージョン番号.tgz** を展開後 **make** を実行します。ここでは OpenVGR ファイルセットが既に **~/opt/OpenVGR** に展開されているものと想定しています。

```
$ tar xzf OpenVGRextra-バージョン番号.tar.gz -C ~/opt/OpenVGR
$ cd ~/opt/OpenVGR/robotvision/src
$ make
```

この make によって robotvision/src ディレクトリに以下のプログラムがビルドされます。

- component/CoordTrans/CoordTransComp 座標系変換 RT コンポーネント例
- tool/cr2xyz ステレオ対応点 3 次元座標復元
- tool/cutroi クロスマーカーテンプレート画像作成
- tool/findx クロスマーカー検出
- tool/transmat 座標系変換行列生成

2.2 サンプルデータによるツールプログラムテスト実行

make に続けて bin ディレクトリから下のデータとシェルスクリプトを使ってプログラムの実行テストを行います。下に示すようにタイプしていきます。

maketransmat.sh を実行したときに matrix ディレクトリに生成された matrix.txt が samples ディレクトリの同名ファイルと一致すればプログラムは正常に動作しています。

下の例では最後の行で matrix.txt の一致確認をしています。一致すれば diff コマンドは何も出力せず終了します。

```
$ cd ../bin/cross
$ ./setup.sh test
  (メッセージ省略)
$ ./cr.sh
  (メッセージ省略)
$ cd ../matrix
$ ./maketransmat.sh
$ diff matrix.txt ../samples
$
```

3. 座標系変換RTコンポーネント例の実行方法

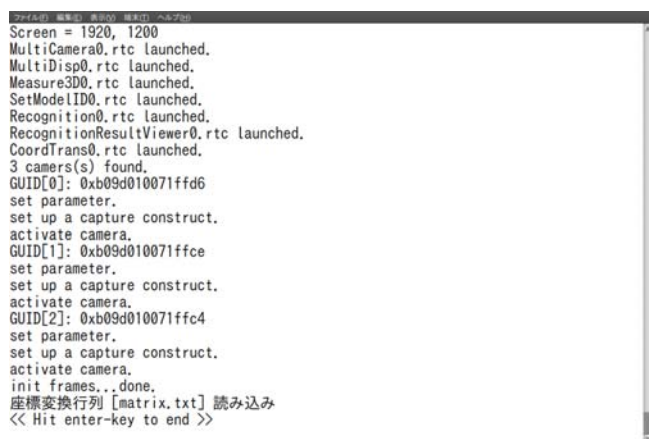
「OpenVGR 操作手順書 4.7. rtshell を利用した実行スクリプト例について」を参照し、OpenVGR/example/script にある captrecog.sh で作業対象認識ができるように準備します。その後 robotvision ディレクトリ下にある CoordTransComp, vision.sh, matrix.txt を以下のように OpenVGR/example/script にコピーします。

vision.sh は座標系変換 RT コンポーネント CoordTransComp を OpenVGR の認識コ

ンポーネントと接続して実行する **rtshell** スクリプト例です。ただし、ロボットを動作させる部分は含まれていません。実際には認識コンポーネントにモデル ID を与える部分とロボット系の認識結果を受け取る部分に利用者のコンポーネントを接続する必要があります。

```
$ cd ~/opt/OpenVGR/robotvision
$ cp src/component/CoordTrans/CoordTransComp ../example/script
$ cp bin/matrix/matrix.txt ../example/script
$ cp bin/samples/vision.sh ../example/script
```

ファイルをコピーした後 **OpenVGR/example/script** ディレクトリで **vision.sh** を実行します。するとターミナルに図 2 のようなメッセージが出力され、**captrecog.sh** の実行時と同様な状態になります。以降の操作は **captrecog.sh** と同じなので、詳細は「**OpenVGR 操作手順書 4.7.2.2. カメラキャプチャ画像の認識**」を参照してください。



```
Screen = 1920, 1200
MultiCamera0.rtc launched.
MultiDisp0.rtc launched.
Measure3D0.rtc launched.
SetModelID0.rtc launched.
Recognition0.rtc launched.
RecognitionResultViewer0.rtc launched.
CoordTrans0.rtc launched.
3 cameras(s) found.
GUID[0]: 0xb09d010071ffd6
set parameter.
set up a capture construct.
activate camera.
GUID[1]: 0xb09d010071ffce
set parameter.
set up a capture construct.
activate camera.
GUID[2]: 0xb09d010071ffc4
set parameter.
set up a capture construct.
activate camera.
init frames...done.
座標変換行列 [matrix.txt] 読み込み
<< Hit enter-key to end >>
```

図 2 vision.sh の実行開始時メッセージ

vision.sh を実行中に **RTSystemEditor** の **System Diagram** で見ると図 3 のようになっています。**CoordTrans** コンポーネントの **RecogResult** ポートを **Recognition** コンポーネントの **RecognitionResultOut** ポートに接続することで、認識結果の座標系を変換できます。変換後の結果を利用するには **CoordTrans** コンポーネントの **Transformed** ポートに接続をします（図 4）。このポートは **Recognition** コンポーネントの **RecognitionResultOut** ポートと同じ型の出力を行います。

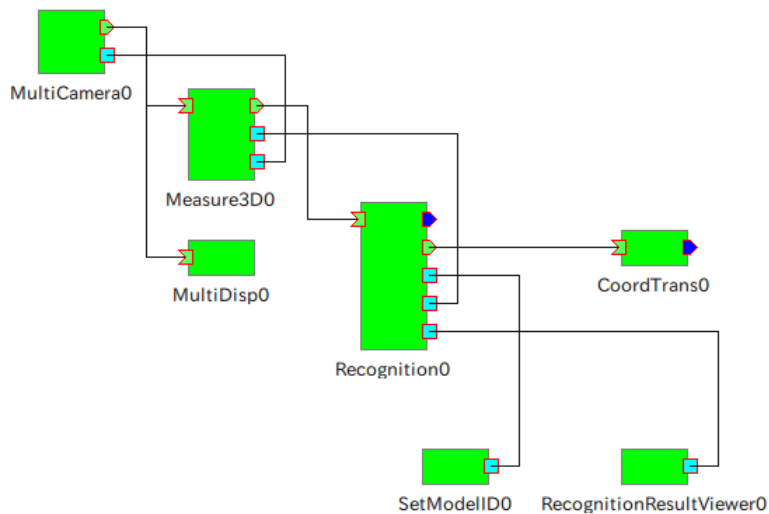


図 3 vision.sh の実行時コンポーネント状態

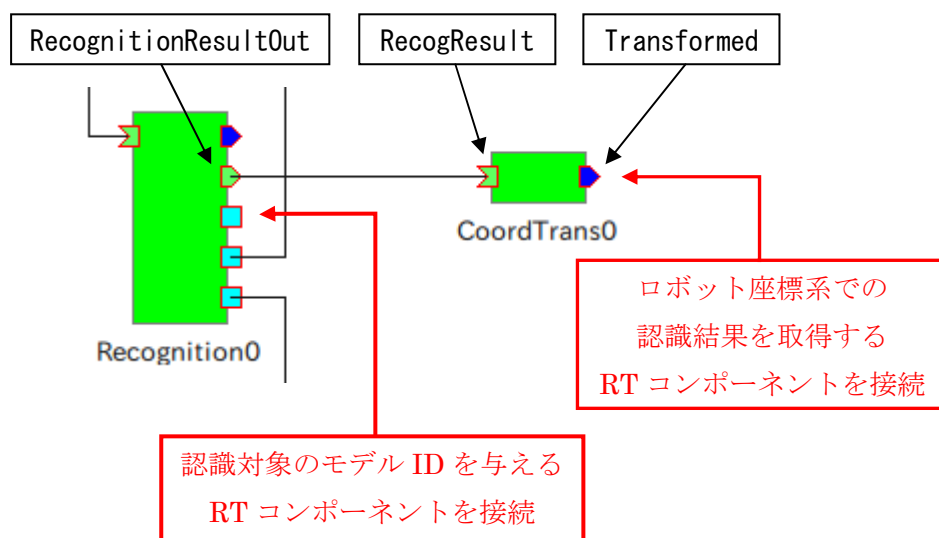


図 4 利用者の RT コンポーネントを接続するポート

CoordTrans コンポーネントのデフォルト時動作は同じディレクトリにある matrix.txt を座標系変換行列として読み込みます。matrix.txt は 4 4 という行で始まる 4×4 同次行列のテキストデータです。コンポーネントのコンフィグレーションパラメータ MatrixFile を変更すると、読み込むファイル名やパスを変更することができます。指定の場所に変換行列データファイルが見つからない場合は、単位行列を使用して Recognition コンポーネントと同じ結果を出力します。

4. 座標系変換行列データの作成

前節で使用した変換行列データ (matrix.txt) は仮のものです。実際にロボットと組み合わせるための変換行列を新たに作る必要があります。作成手順の概略を図 5に示します。



図 5 座標系変換行列データ作成の概略

以降で作成手順の詳細を説明します。

4.1 クロスマーカ－画像の撮影

robotvision/samples ディレクトリにある画像のように、ロボットの作業空間内にクロスマーカ－を置いて撮影を行います。また、このクロスマーカ－中央点の 3 次元位置をロボット座標系で記録しておきます。このデータを異なる位置で 14 点用意します。マーカ－や作業空間内での配置についての詳細は、同梱の文書「クロスマーカ－検出プログラムの使い方」を参照してください。

OpenVGR のインストール・準備後には OpenVGR/build ディレクトリにキャリブレーションパターンを撮影するときに使う ichimatsu というプログラムがあるので、クロスマーカ－画像撮影にはこれを用います。このプログラムを以下のように実行します。


```
$ ./i chimatsu
```

カメラ画像が表示されているウインドウを選択している状態で、アルファベットの **p** キーを押すと、その時のカメラ画像が **cap_日付-時刻_0[012].png** というファイル名で保存されます。全ての撮影が終了したら **q** キーを押すとプログラムが終了します。

4.2 ロボット座標系マーカー位置リストの作成

次に記録しておいた各画像のマーカー中央点位置座標の全部を、**3d.txt** という名前のファイルを作って記述します。このファイルは各点の **x, y, z** 座標（ロボット座標系）をスペースで区切って一行ずつ書いたものです。なお、ツールプログラムの扱う 3 次元座標系はすべて右手系です。点座標の順番は対応するマーカー画像ファイルの順番に合わせます。実例が **samples** ディレクトリにあるので参照してください。

4.3 テンプレート画像の作成

撮影画像のうちから一つを選び、それをもとにクロスマーカーのテンプレート画像を作成します。ファイル名は **template.png** とします。テンプレート作成方法の詳細については「クロスマーカー検出プログラムの使い方」を参照下さい。

4.4 マーカー位置の検出

例えば **robotvision/bin** の下に **work** というディレクトリを作って上記 4.1～4.3 で作成した画像、マーカー位置リスト、カメラキャリブレーションファイル **camera_calib.yaml** をまとめて置きます。このあと、**robotvision/bin/cross** ディレクトリに移動し下の手順でクロスマーカーの検出を行います。

```
$ ./setup.sh ../work
(メッセージ省略)
$ ./cr.sh
(メッセージ省略)
```

setup.sh の引数として、データを置いてあるディレクトリを指定して実行します。次に **cr.sh** を実行します。このとき

```
testin***.png : クロス線の検出に失敗しました.
```

というメッセージが出なければ何らかの検出ができています。ここで、

```
$ ./mon.sh
```

とタイプすると図 6 のような out00t.png～out13t.png というファイルを生成するので、display コマンドなどを使って目視でこれら 14 個のファイルについて検出結果の確認をします。黄色の点がクロスマーカの中央を正しく示していれば検出成功です。この画像が小さくてわかりにくいなど結果を詳細に見る場合は、例えば以下のように out*[0-2].png を確認してください。これは out00t.png～out13t.png の元になったファイルです。

```
$ display out*[0-2].png
```



図 6 検出結果確認画像

cr.sh を disp オプションと共に実行すると、処理をするひとつの画像毎に検出結果としてこれらの画像をウインドウ表示するので、処理を実行しながら結果の確認ができます。また、中間結果として以下の三種類の画像を生成しているので、クロス線の検出ができていないか、マーカ中央点を正しく見つけていない場合はそれを参考にできます。

- out***-extr. bmp テンプレートマッチ結果 2 値化画像
- out***-edge. bmp マーカ一部エッジ検出結果画像
- out***.line. bmp クロス線検出結果画像

この画像をもとに -bthr, -lmin, -gmax, -smin オプションのパラメータを調節します。これらのオプションは cr.sh の引数として指定することができます。例えば bthr の値を変更してその場で結果を確認するときは以下のようになります。

```
$ ./cr.sh -bthr 90 -disp
```

指定したオプションは cr.sh から呼ばれる findx プログラムにそのまま与えられます。オプションの意味と調整の方法は「クロスマーカ検出プログラムの使い方」を参照下さい。

4.5 変換行列データファイルの作成

前節 4.4 でマーカ位置の検出を行った `cross` ディレクトリから `matrix` ディレクトリに移動し、`maketransmat.sh` を実行します。

```
$ cd ../matrix  
$ ./maketransmat.sh
```

ここで生成された `matrix.txt` が目的の変換行列データファイルとなります。座標系変換コンポーネントの実行時にこのファイルが読み込まれるようにしてください。

スクリプト `maketransmat.sh` の内部では、`cr.sh` で検出したマーカ中央点画素位置を記したファイル `Cdata.[01]` とカメラキャリブレーションデータ `camera_calib.yaml` を使ってマーカ位置の 3 次元復元計算を行います。その後マーカのロボット座標系位置を記したファイル `3d.txt` との対応からカメラ系→ロボット系座標変換行列の計算をします。参照するファイルについては、`setup.sh` を実行したときにリンクが作成されるようになっています。

5. ツールプログラムについて

5.1 プログラム間関係

使用するツールプログラムとデータの関係を図 7に示します。

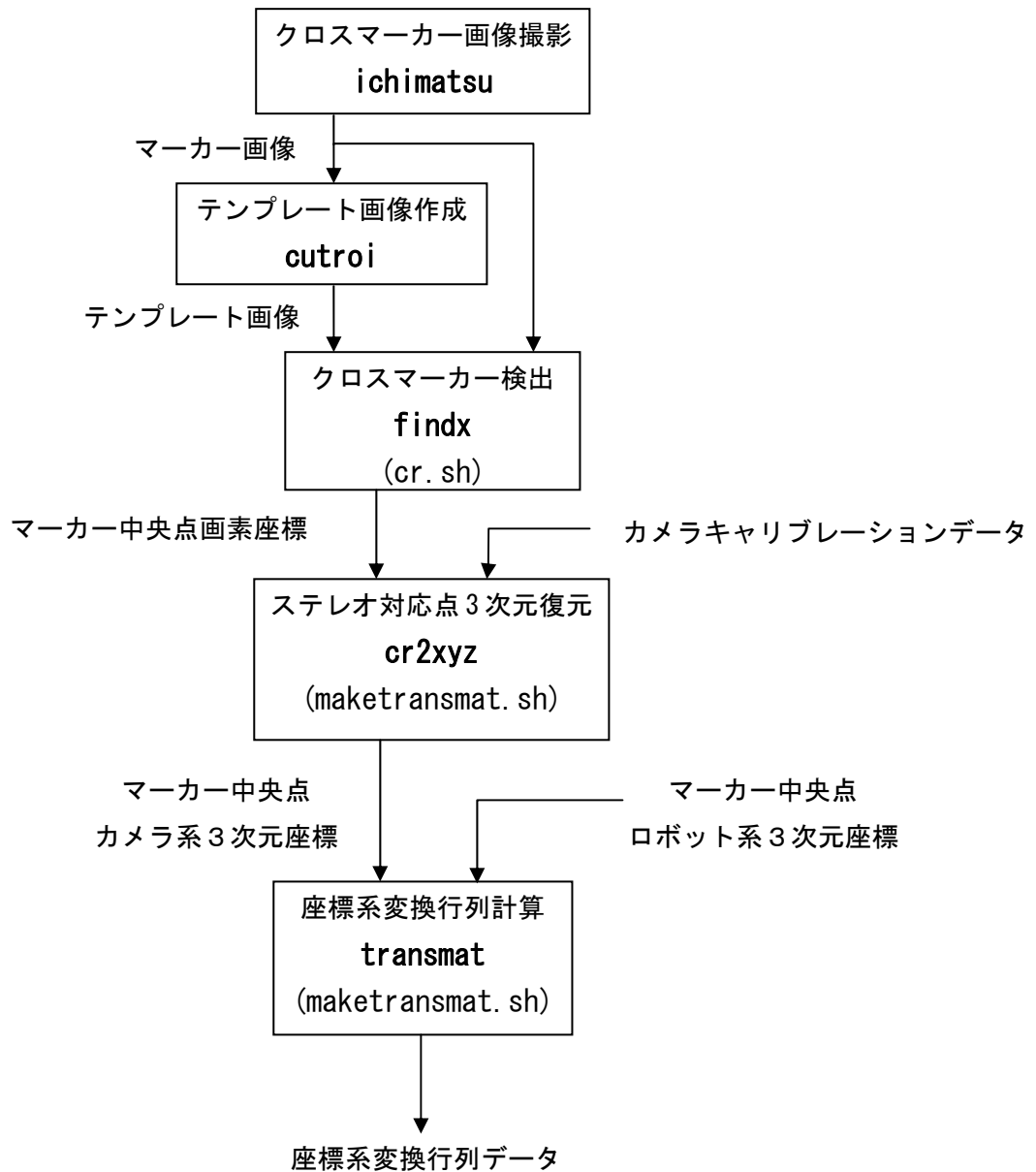


図 7 ツールプログラムの関係

5.2 単体プログラム実行方法の詳細について

ツールプログラム単体の実行方法の詳細については以下の文書を参照してください。

- ichimatsu

OpenVGR 操作手順書 4.6 節 カメラキャリブレーションツール
(OpenVGR ファイルセットに含まれる)

- cutroi, findx

クロスマーカー検出プログラムの使い方 (OpenVGRextra ファイルセットに同梱)

- cr2xyz, transmat

座標系変換行列計算プログラムの使い方 (OpenVGRextra ファイルセットに同梱)

6. 座標系変換RTコンポーネント例の仕様

CoordTransComp (座標系変換 RTC)

動作環境

この RTC は次の環境で動作する。

動作 OS	Ubuntu 10.04 LTS Desktop 日本語 Remix(32bit 環境)
開発言語	C, C++
コンパイラ	GNU Compiler Collection 4.4.3
RT ミドルウェア／バージョン	OpenRTM-aist-1.0.0-RELEASE (C++)
依存パッケージ	OpenCV 2.0

ポート情報



図 8 座標系変換 RTC

A) データポート (InPort)

名称	型	データ長	説明
RecogResult	TimedDoubleSeq	不定	作業対象認識 RTC が出力する認識結 果

B) データポート (OutPort)

名称	型	データ長	説明
Transformed	TimedDoubleSeq	不定	入力された認識結 果に対して座標系 変換を施したもの

コンフィグレーション

名称	型	デフォルト値	説明
MatrixFile	string	“matrix.txt”	座標系変換行列デ ータファイル名

入出力データフォーマット

A) 入出力データ構造

入出力ポートの内容は、どちらも 20 個の浮動小数点を一組のデータとする 0 組以上のデータ列である。一組のデータの内訳は、先頭から順に次の通りである。

0	カメラセット ID	撮影を行ったステレオカメラの番号
1	モデル ID	認識に用いたモデルの番号
2	認識候補番号	0 から始まる通し番号。有力な認識候補ほど若い番号。
3	座標系番号	座標系を表す番号。今は 0 のみ。
4	評価値	確かさの評価値。大きいほど有力な認識候補。
5	エラーコード	エラーコード（後述）
6	予備 1	未定義。予約されたデータ項目。
7	予備 2	未定義。予約されたデータ項目。
8	R_{00}	(後述・以下同様)
9	R_{01}	
10	R_{02}	
11	T_x	
12	R_{10}	
13	R_{11}	
14	R_{12}	
15	T_y	
16	R_{20}	
17	R_{21}	
18	R_{22}	
19	T_z	

なお R と T は次の 4×4 の行列の要素を意味する。これは座標変換行列であり、モデルからオブジェクトへの回転と移動を表す。座標系は右手系である。入力 of R と T に、設定ファイルの行列データで座標系変換を行った R と T が出力される。

$$\begin{pmatrix} R_{00} & R_{01} & R_{02} & T_x \\ R_{10} & R_{11} & R_{12} & T_y \\ R_{20} & R_{21} & R_{22} & T_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

R と T 以外のデータは入力ポートから入ったものがそのまま出力ポートにコピーされる。データ内容の詳細については「OpenVGR 機能仕様書 3.4 RecognitionComp（作業対象認識 RTC）」を参照のこと。

B) エラーコード

エラーコードは次のように定義されている。

番号	意味
-1	関数の引数が不正
-2	メモリが不足している
-3	ファイルのオープンに失敗した
-4	不正なファイルフォーマット
-101	不正な画像サイズ。画像の幅または高さが 0
-102	画像の不足。入力された画像が 1 枚以下
-103	ステレオ画像のサイズが同じではない
-104	ステレオ画像のカラー形式が同じではない
-105	モデル番号に対応するファイルが見つからない

設定ファイル

この RTC では、RTC のコンフィグレーションによって次の設定ファイルを指定する。

● 座標系変換行列データ

座標系変換行列データファイルは、先頭行で行列の行・列数を示し、以降に行列データを記述したテキストファイルである。ただし本 RTC では 4×4 行列のみ使用する。行内の数値は空白で区切る。以下に変換行列データファイルの例を示す。

```
4 4
-0.013253 -0.857027 0.515100 101.776574
-0.998863 -0.012245 -0.046073 231.416043
0.045793 -0.515125 -0.855891 1641.141210
0.000000 0.000000 0.000000 1.000000
```

RTC の活性化時に変換行列データファイルを読み込めなかった場合は、単位行列を使用する。