HALCON プロバイダ

MVTec 画像処理

Version 1.2.0

ユーザーズ ガイド

Jun 28, 2017

【備考】

【改版履歴】

バージョン	日付	内容
1.0.0.0	2009-06-22	初版. HALCON 9.0.1 に対応
1.1.0.0	2010-5-11	HALCON 8.0.4 に対応. HALCON 9.0.2 に対応. CaoTask クラスの実装追加.
1.1.0.1	2011-03-11	プロバイダ登録ツールに関する追記
1.1.0	2012-07-17	ドキュメントのバージョンルールを変更
1.1.0.43	2012-09-28	HALCON 10.0.3 に対応. HALCON 11.0.0.1 に対応
1.1.0.108	2014-08-18	HALCON 11.0.3 に対応.
1.1.0.138	2015-06-19	HALCON12, HALCON11.0.4, HALCON10.0.4, HALCON9.0.4 に対応
1.2.0.0	2016-02-25	HALCON12.0.1, HALCON11.0.5 に対応.
1.2.0.7	2016-05-11	HALCON12.0.2 に対応.
1.2.0.29	2016-12-06	HALCON13 に対応
1.2.0.56	2017-06-28	HALCON13.0.1, HALCON12.0.3 に対応

【対応機器】

機種	バージョン	注意事項
HALCON 8	8.0.4	8.0.4 以前のバージョンでは正常に動作しない恐れがあります.
HALCON 9	9.0.4	9.0.4 以前のバージョンでは正常に動作しない恐れがあります.
HALCON 10	10.0.4	10.0.4 以前のバージョンでは正常に動作しない恐れがあります.
HALCON 11	11.0.5	11.0.5 以前のバージョンでは正常に動作しない恐れがあります.
HALCON 12	12.0.3	12.0.3 以前のバージョンでは正常に動作しない恐れがあります.
HALCON 13	13.0.1	13.0.1 以前のバージョンでは正常に動作しない恐れがあります.

【ご注意】

本プロバイダを使用する場合は，別途ライセンス購入が必要です．このライセンスには，**HALCON** 本体のランタイムラインセンスは含まれていませんので注意してください．

目次

1. はじめに

本プロバイダは，MVTec 社製の画像処理ライブラリ「HALCON」を ORiN 経由で使用するために作成されたプロバイダです．本書は，そのユーザーズガイドです．

注意：HALCON プロバイダを使用するには，「HALCON」本体をインストールしなければなりません．「HALCON」の 64bit 版には対応していませんので注意してください．「HALCON」本体のインストール後にプロバイダをレジストリ登録する必要があります．レジストリ登録の方法は表 2-1 を参照してください．

2. プロバイダの概要

2.1. 概要

本プロバイダは，「HALCON」を ORiN から呼び出すためのゲートウェイプロバイダです．ORiN CAO エンジン上で「HALCON」の画像処理オペレータや，ユーザが「HALCON」の統合開発環境「HDevelop」で作成したプログラム・外部プロシージャをそのまま使用することが可能です（図 2-1）．

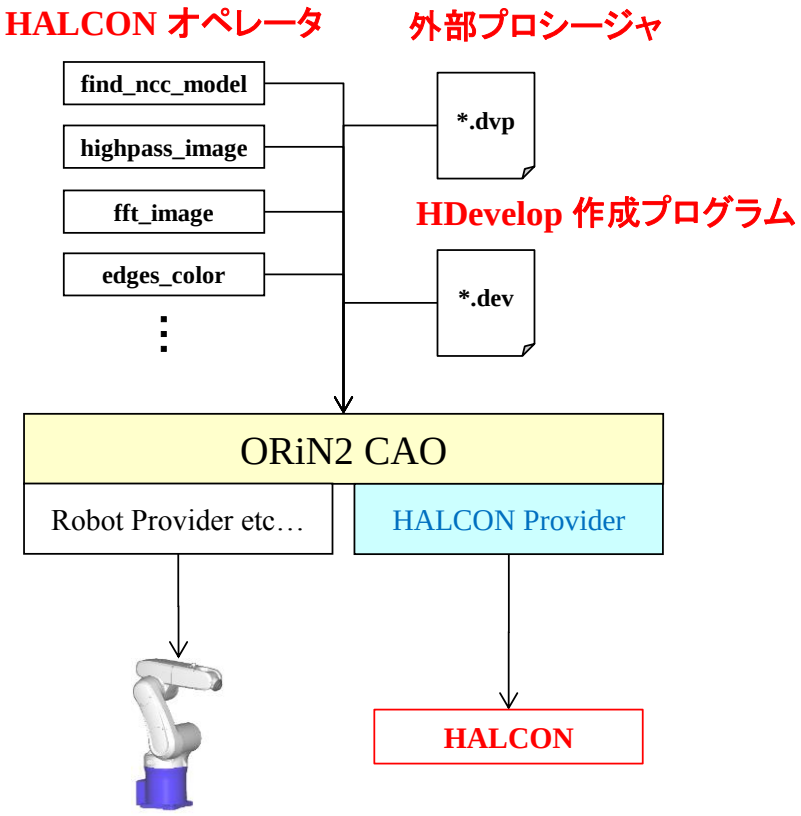

図 2-1 プロバイダ概要

2.2. インストール

プロバイダは，「HALCON」のメジャーバージョンごとにファイルが異なります．お使いの「HALCON」のバージョンに合わせて，手動で登録 / 抹消して下さい．

表 2-1 HALCON プロバイダ

ファイル名	CaoProvHALCON.dll
ProgID	CaoProv.HALCON
レジストリ登録[1]	regsvr32 CaoProvHALCON.dll
レジストリ登録の抹消	regsvr32 /u CaoProvHALCON.dll

[1] ORiN SDK でインストール／アンインストールしても，自動で登録／抹消されません．手動で登録／抹消してください．

2.3. ライセンスの追加

本プロバイダを使用可能にするには，「HALCON」本体，および，ORiN2 SDK をインストール後，別途「HALCON プロバイダ」ライセンスを入力する必要があります．下記にそのライセンスの追加手順を示します．正規ライセンスがない場合には下記の試用ライセンスをご利用ください．

HLGY-YZZ2-QL95-MDLG （3 ヶ月有効）

[追加手順]

1. CaoConfig（図 2-2）を起動し，[Cao Provider]タブを選択する
2. Provider List から[HALCON CAO Provider]項目を選択する
3. License 項目の[…]ボタンをクリックする
4. ORiN2 License Manager で[Add]ボタンをクリックする
5. 入手したライセンスキーを入力後，[OK]ボタンをクリックする
6. [Close] ボタンをクリックし，CaoConfig を終了する

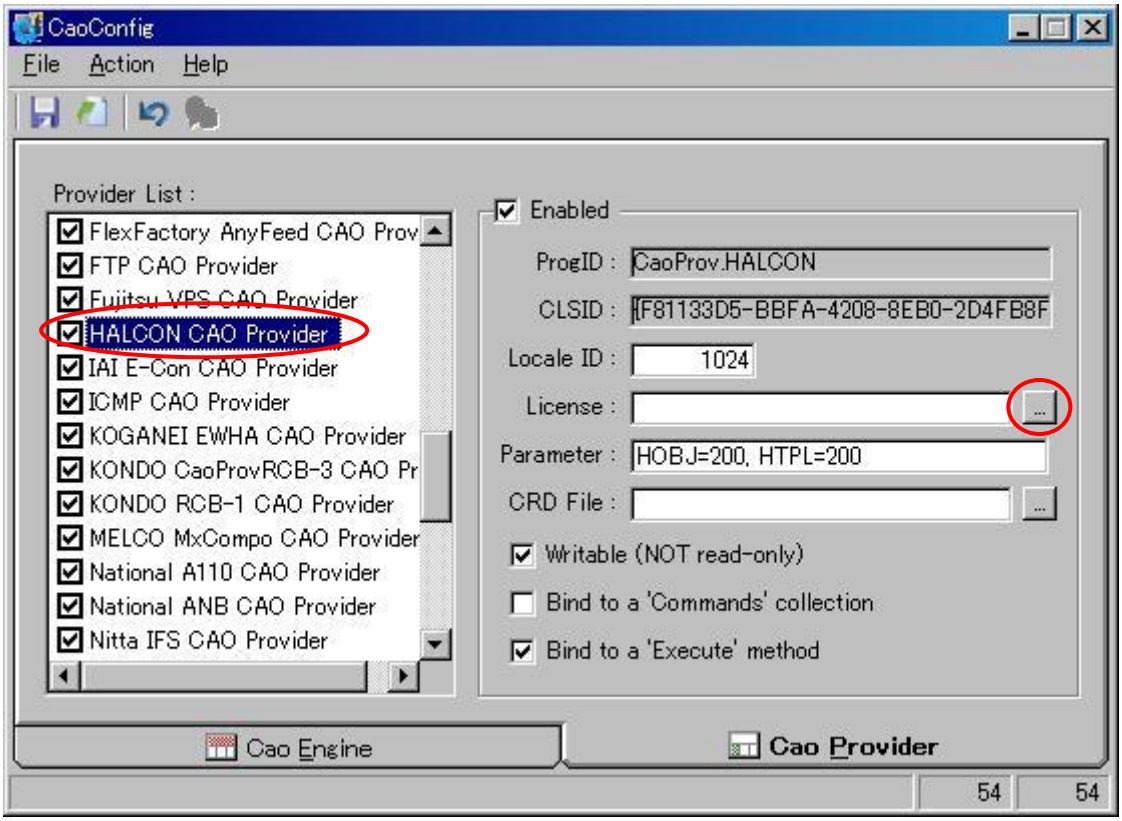

図 2-2 CaoConfig によるライセンス追加

2.4. レジストリの設定

HALCON プロバイダでは，内部メモリ等の設定をレジストリに登録します．登録内容の変更は CaoConfig（図 2-2）を使用して行います．以下に設定内容を示します．

表 2-2 CaoConfig の Parameter 文字列

パラメータ	意味
HOBJ=＜配列サイズ＞	Hobject 配列のサイズを指定します. 1 以上の値を指定します. （デフォルト:200）
HTPL=＜配列サイズ＞	HTuple 配列のサイズを指定します. 1 以上の値を指定します. （デフォルト:200）

2.5. 内部メモリ

HALCON プロバイダは, その内部に,「Hobject」,「HTuple」というデータ型[2]の固定長メモリ領域をそれぞれ1つずつ持っています. プロバイダ内では, 呼び出し元から入力値として指定されたメモリ位置（管理番号）にある「Hobject」,「HTuple」データを使って,「HALCON」の画像処理オペレータを実行します. 実行結果は, プロバイダの呼び出し元が出力として指定したメモリ位置（管理番号）に格納されます.

2.6. HDevelop 外部プロシージャ呼び出し

「HALCON」の開発環境「HDevelop」で作成したプログラムを外部プロシージャファイル（*.dvp）化して, CaoController::Execute メソッドでそのまま呼び出すことができます.「HALCON」の外部プロシージャファイル（*.dvp）の作成方法については「HALCON」の「HDevelop Users Guide」を参照して下さい. 外部プロシージャファイルの呼び出しの際には, 外部プロシージャファイルが置いてあるフォルダのパスを登録する必要があります. パスを新たに登録する際は CaoController::Execute（”AddProcedurePath”, <bstrPath:BSTR >）を使用します.

2.7. HDevelop プログラムの呼び出し

「HALCON」の開発環境「HDevelop」で作成したプログラムファイル（*.dev）を, CaoTask のインスタンスとして生成し, CaoTask::Start メソッドでそのまま呼び出すことができます.

2.8. メッセージ転送機能

CAO エンジンのメッセージ転送機能により, 他のプロバイダから送信された画像データを, プロバイダの内部メモリに格納することができます. 転送メッセージの画像データは, DIB（Device Independent Bitmap）形式の VARIANT 型データ（VT_UI1 | VT_ARRAY）である必要があります. メッセージを受信した際, 受信状態を通知する CaoController クラスの OnMessage イベントが発行されます. 受信したデータを内部メモリのどの位置に格納するかは, システム変数「@MSG」の CaoVariable::put_ID（）プロパティで設定します.

[2] Hobject, HTuple は,「HALCON」で定義されているデータ型です.

2.9. メソッド・プロパティ

2.9.1. CaoWorkspace::AddController メソッド

プロバイダ内部に「Hobject」「HTuple」の内部メモリをそれぞれ確保するとともに，「HALCON」の初期化を行います．内部メモリの確保は，初回の AddController 時のみ行われ，初回以降は，どのアプリケーションから呼ばれても，最初に確保した内部メモリを共有して使用することになります．

書式 AddController(<bstrCtrlName:BSTR>,<bstrProvName:BSTR>,
<bstrPcName:BSTR > [,<bstrOption:BSTR>])

bstrCtrlName : [in] コントローラ名
bstrProvName : [in] プロバイダ名．固定値 = “CaoProv.HALCON“
bstrPcName : [in] プロバイダの実行マシン名
bstrOption : [in] オプション文字列＝“<オプション 1>,<オプション 2>,…”

表 2-3 CaoWorkspace::AddController のオプション文字列

オプション	説明
HOBJ=＜配列サイズ＞	Hobject 配列のサイズを指定します．1 以上の値を指定します． 指定しない場合は，レジストリに登録してある同名パラメータからサイズを決定します（デフォルト:200）
HTPL=＜配列サイズ＞	HTuple 配列のサイズを指定します．1 以上の値を指定します．指定しない場合は，レジストリに登録してある同名パラメータからサイズを決定します． （デフォルト:200）

2.9.2. CaoController::AddVariable メソッド

表 4 に示すシステム変数が実装されています．

書式 AddVariabler(<bstrVarName:BSTR> [,<bstrOption:BSTR>])

bstrVarName : [in] 変数名
bstrOption : [in] オプション文字列＝“<オプション 1>,<オプション 2>,…”

表 2-4 CaoController::AddVariable のオプション文字列

オプション	説明
ID=＜内部メモリ位置＞	システム変数「@MSG」専用．格納する「Hobject」内部メモリの位置を指定します．指定しない場合は，データ配列の最後尾が指定されます． （デフォルト 200）

2.9.3. CaoController::AddTask メソッド

HALCON の統合開発環境「HDevelop」で作成したプログラムの実行タスクを生成します.

書式 AddTask(<bstrTaskName:BSTR> [,<bstrOption:BSTR>])

bstrTaskName : [in] 「HDevelop」で作成したプログラムファイル(*.dev)の絶対パス

bstrOption : [in] オプション文字列(未使用)

2.9.4. CaoController::Execute メソッド

コマンド名で指定した「HALCON」の画像処理オペレータ・外部プロシージャを実行します. パラメータ配列は, 4 つの要素を持ち, 先頭から順に「HALCON」オペレータの 4 種の引数 “input object”, “output object”, “input control”, “output control” に対応しています. パラメータ配列の各要素には, パラメータが格納されているメモリ位置を指定する必要があり, メモリ位置を指定する場合は VT_I4 (| VT_ARRAY)で, 空文字を指定する場合は, VT_BSTR で指定します.「HALCON」の画像処理オペレータのパラメータ仕様に関しては,「HALCON」付属の「HDevelop Operator Reference Manual」 を参照してください.

書式 Execute(<bstrCmd:BSTR > [,<vntParam:VARIANT>])

bstrCmd : [in] HALCON オペレータ名

vntParam : [in] パラメータ配列

vntParam = Array(< vntP1:VARIANT >, < vntP2:VARIANT >,
< vntP3:VARIANT >, < vntP4:VARIANT >)

vntP1 : [in] 入力 Hobject. 複数の場合は配列で指定. 不要な場合は空文字を設定.

vntP2 : [out] 出力 Hobject. 複数の場合は配列で指定. 不要な場合は空文字を設定.

vntP3 : [in] 入力 HTuple. 複数の場合は配列で指定. 不要な場合は空文字を設定.

vntP4 : [out] 出力 HTuple. 複数の場合は配列で指定. 不要な場合は空文字を設定.

例 :

```
Execute( “reduce_domain”, Array( Array( 1, 2 ), 10, "", "" ) )
Execute( “set_ncc_model_origin”, Array( "", "", Array( 11, 12, 13 ), "" ) )
Execute( “get_image_pointer3”, Array( 1, "", "", Array( "", "", "", 5, 6, 7 ) )
```

2.9.5. CaoController::get_VariableNames プロパティ

CaoController オブジェクトで対応している変数名のリストを返します.

2.9.6. CaoController::OnMessage イベント

CaoController クラスの OnMessage イベントの実装状況を下表に示します.

表 2-5 OnMessage イベントの実装状況一覧

番号	データ型	通知内容	意味
1	VT_BSTR	エラーメッセージ	「HDevelop」プログラムタスクの実行状態を返します. 正常終了:NULL, 異常終了:エラーメッセージ
2	VT_I4	メッセージ受信状態	インプロセスメッセージの受信状態を返します. 受信成功:0, 不正データ[3]を受信:1, 受信失敗:0 以外

2.9.7. CaoVariable::get_ID プロパティ

システム変数「@HOBJ」,「@HTPL」,「@MSG」の場合, 現在の Variable オブジェクトに割当てられたメモリ位置を取得します. データ型は VT_I4 です. CaoTask オブジェクトから生成された場合は, 使用できません.

2.9.8. CaoVariable::put_ID プロパティ

システム変数「@HOBJ」,「@HTPL」,「@MSG」の場合, 現在の Variable オブジェクトに割当てられているメモリ位置を指定した値に直接変更します. CaoTask オブジェクトから生成された場合は, 指定した値のメモリ位置に, 現在の Variable オブジェクトの内容をコピーします.

2.9.9. CaoVariable::get_Value プロパティ

現在の Variable オブジェクトに割当てられたメモリに格納されている値を取得します. データタイプが「Hobject」の場合, 格納されているデータが HALCON の画像データ型であれば DIB(Device Independent Bitmap)形式の VARIANT 型データ(VT_UI1 | VT_ARRAY)に変換されて出力されます. それ以外の HALCONデータ型の場合はエラーとなります. データタイプが「HTuple」の場合, VARIANT型データに変換されて出力されます.

2.9.10. CaoVariable::put_Value プロパティ

現在の Variable オブジェクトに割当てられたメモリに値を設定します. データタイプが「Hobject」の場合, 設定データは DIB 形式の VARIANT 型データ(VT_UI1 | VT_ARRAY)である必要があります. それ以外のデータ型の場合はエラーとなります. データタイプが「HTuple」の場合, VARIANT 型データ(整数, 小数, 文字列)がそのまま設定可能です. CaoTask オブジェクトから生成された場合は, 使用できません.

2.9.11. CaoTask::AddVariable メソッド

タスクとして生成した「HDevelop」プログラムの内部で使用されている変数と同名の変数が実装されています. オプション文字列を指定しない場合, 指定した変数名は, ローカル変数の名前として認識されます. オブジェクトの操作に関しては, CaoVariable オブジェクトの項目を参照して下さい.

[3] DIB(Device Indepent Bitmap)を表現する VARIANT 型データ(VT_UI1 | VT_ARRAY)以外のデータ

書式 AddVariable(<bstrVarName:BSTR> [,<bstrOption:BSTR>])

bstrVarName : [in] 「HDevelop」プログラムの内部で使用されている変数名

bstrOption : [in] オプション文字列

表 2-6 CaoTask::AddVariable のオプション文字列

オプション	説明
GLOBAL	指定した変数名は，グローバル変数の名前として認識されます.

2.9.12. CaoTask::Start メソッド

タスクとして生成した「HDevelop」プログラムを実行します. オプション文字列を指定することによって，実行の同期・非同期を設定することができます. 非同期実行の場合，タスクの実行完了時に CaoController クラスの OnMessage イベントが発行されます.

書式 Start(<lMode:I4> [,<bstrOption:BSTR>])

lMode : [in] 開始モード（1： 1サイクル，2： 繰り返し，3： ステップ，4： ステップバック）

bstrOption : [in] オプション文字列＝“<オプション 1>,<オプション 2>,…”

表 2-7 CaoTask::Start のオプション文字列

オプション	説明
ASync	タスクを非同期実行します.（デフォルト設定）
Sync	タスクを同期実行します.
＜タイムアウト時間＞	タイムアウト時間 [ms] を過ぎると呼び出し元に制御を返します.

2.9.13. CaoTask::Stop メソッド

タスクとして生成した「HDevelop」プログラムを停止します.

書式 Stop(<lMode:I4> [,<bstrOption:BSTR>])

lMode : [in] 停止モード（0： デフォルト，1： 瞬時，2： ステップ，3： サイクル，4： 初期化）

bstrOption : [in] オプション文字列（未使用）

2.9.14. CaoTask::Delete メソッド

タスクを削除し，ロードした「HDevelop」プログラムをアンロードします.

2.9.15. CaoTask::get_FileName プロパティ

現在ロードされている「HDevelop」プログラムファイルの絶対パスを返します.

2.9.16. CaoTask::get_VariableNames プロパティ

タスクとして生成した「HDevelop」プログラム内部で使用されている変数名のリストを取得します. オプション文字列を指定しない場合,「HDevelop」プログラム内部で使用されている全てのローカル変数名を返します.

書式 get_VariableNames([<bstrOption:BSTR>])

bstrOption : [in] オプション文字列＝“<オプション 1>,<オプション 2>,…”

表 2-8 CaoTask::get_VariableNames のオプション文字列

オプション	説明
HOBJ	プログラム内部で使用されている Hobject ローカル変数名だけを返します.
HTPL	プログラム内部で使用されている HTuple ローカル変数名だけを返します.
GLOBAL_HOBJ	プログラム内部で使用されている Hobject グローバル変数名だけを返します.
GLOBAL_HTPL	プログラム内部で使用されている HTuple グローバル変数名だけを返します.
GLOBAL	プログラム内部で使用されている全てのグローバル変数名を返します.

2.10. 変数一覧

2.10.1. コントローラクラス

表 2-9 コントローラクラス システム変数一覧

変数名	データ型	説明	属性	
			get	put
@HOBJ	VT_UI1 | VT_ARRAY	Hobject 配列を直接操作するためのシステム変数	○[4]	○[5]
@HTPL	VT_VARIANT	HTuple 配列を直接操作するためのシステム変数	○	○
@HOBJMAX	VT_I4	Hobject 配列の最大サイズを取得するための変数	○	
@HTPLMAX	VT_I4	HTuple 配列の最大サイズを取得するための変数	○	
@MSG	VT_UI1 | VT_ARRAY	受信メッセージが格納されるシステム変数	○	○

[4] Hobject 型データが画像データを表現している場合のみ
[5] VARIANT 型のデータが DIB(Device Indepent Bitmap)を表現する（VT_UI1 | VT_ARRAY）のデータの場合のみ

付録A. エラーコード一覧

プロバイダのエラーコードは HRESULT 型の構造になっています. HRESULT 型の詳細については, 以下の URL を参照してください.

<http://msdn2.microsoft.com/en-us/library/bb401631.aspx>

プロバイダ内で使用しているエラーコードには, Microsoft Windows で標準的に定義されている「標準エラー」とプロバイダ独自に定義した「カスタムエラー」の 2 種類があります. 標準エラーは, Winerror.h で定義されている共通のエラーコードです. カスタムエラーはプロバイダで定義されているローカルなエラーコードです. 以下にエラーコードの一覧を示します.

表 A-1 HALCON プロバイダのエラーコード

標準エラー(抜粋)		
番号	マクロ名	内容
0x00000000	S_OK	No error occurred
0x80004001	E_NOTIMPL	Not implemented
0x80070057	E_INVALIDARG	One or more arguments are invalid
0x80004005	E_FAIL	Unspecified error
カスタムエラー		
番号	概要	内容
0x**80014**201	プロバイダ ライセンスエラー	プロバイダの使用ライセンスが有効になっていません. ライセンスを購入してください.
0x**80014**001 ～	プロバイダ 内部例外エラー	プロバイダ内で異常終了した際に発生します.
0x**8017**0000 ～	HALCON オペレータエラー	HALCON オペレータ実行時にエラーが発生した場合, HALCON オペレータが返したエラーコードを 0x**8017**0000 に加算して返します. HALCON オペレータのエラーコードについては, HALCON のマニュアルを参照してください.