QRCodeCalibration プロバイダ

Version 1.0.1

ユーザーズ ガイド

June 20, 2025

備考:

【改版履歴】

バージョン	日付	内容
1.0.0	2024-11-19	初版.
1.0.1	2025-06-20	- GetMarkerInfo および GetImageAcquisitionPos コマンドの位置・姿勢精度向上. - 「付録 A. QR コード位置補正の実施例」を追加.

【対応機器】

機種	バージョン	注意事項

目次

1. はじめに

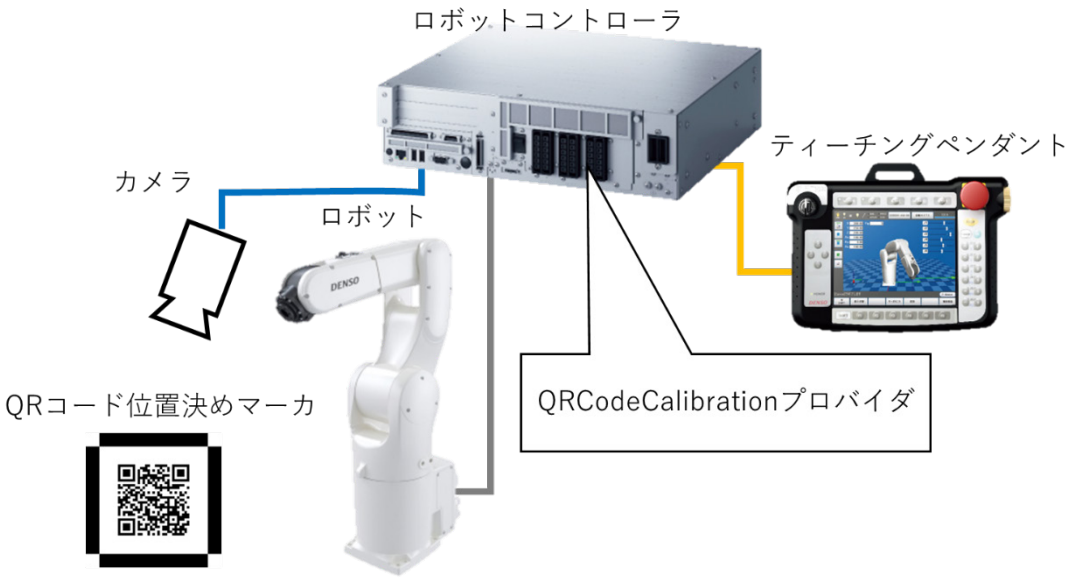
本書はQRコード位置補正に利用するQRCodeCalibrationプロバイダのユーザーズガイドです. 本書で扱うCAOプロバイダ(CaoProv DENSOQRCodeCalibration.dll)をQRCodeCalibrationプロバイダと呼びます.

次章にQRCodeCalibrationプロバイダの概要, 3章にコマンドリファレンス, 4章にサンプルプログラムを記載しています.

2. プロバイダの概要

2.1. 概要

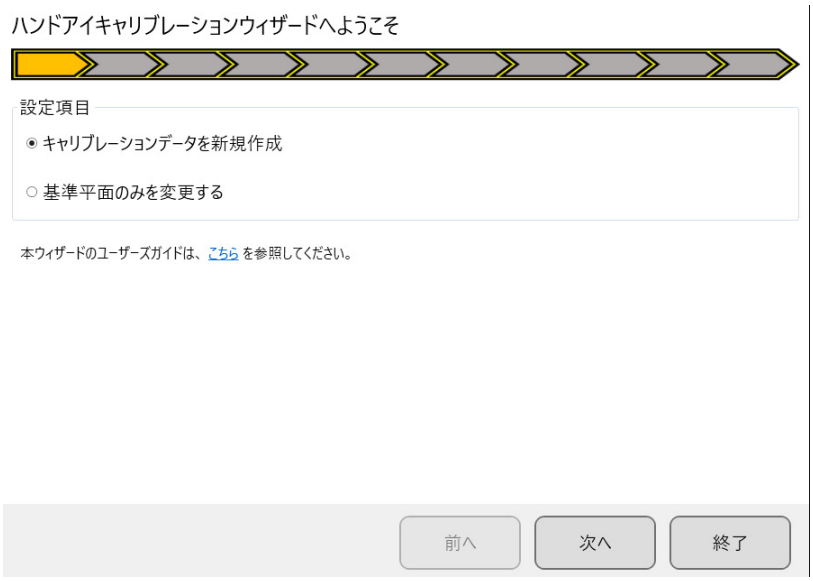
QRCodeCalibration プロバイダは，カメラキャリブレーションデータ，および QR コード位置決めマーカを利用してロボットの位置補正を行う CAO プロバイダです．



本プロバイダを実行するには，以下の条件を満たす必要があります．

1. ロボットコントローラに「QR コード位置補正」のライセンスが登録されている事．
   コントローラのライセンスの登録については，ユーザーマニュアルの”オプション＞オプション機能とライセンス＞ライセンス登録”を参照してください．
2. ロボットの 6 軸ツール部分にカメラを取り付けている事．
3. ハンドアイキャリブレーションウィザードでキャリブレーションデータを送信している事．
   ハンドアイキャリブレーションウィザードは，WINCAPSIII に付属しているアプリケーションです．
   キャリブレーションデータの送信方法については，ハンドアイキャリブレーションウィザードのユーザーズガイドを参照してください．

   ユーザーズガイドは以下の手順で確認できます．
   i. WINCAPSIII をインストールする．
   ii. ハンドアイキャリブレーションウィザードを起動する．
   iii. 文中のリンク「こちら」を押下する．


ハンドアイキャリブレーションウィザードへようこそ
設定項目
キャリブレーションデータを新規作成
基準平面のみを変更する
本ウィザードのユーザーズガイドは、こちら を参照してください。
前へ
次へ
終了

2.2. QR コード位置決めマーカ

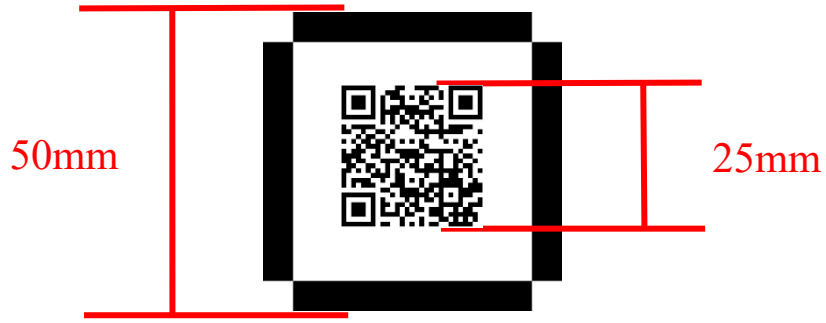
本プロバイダでは，QR コード位置決めマーカ（以下，マーカ）を使用します．

マーカとは，下図のような白黒の図形の中央に QR コードを配置したものを指します．マーカのテンプレートは，ユーザーマニュアルから取得できます．QR コードモデル 2 のみが使用できます．白黒反転，表裏反転している QR コードは使用できません．

QR コードの作成には，「https://m.qrqrq.com/」の利用を推奨します．

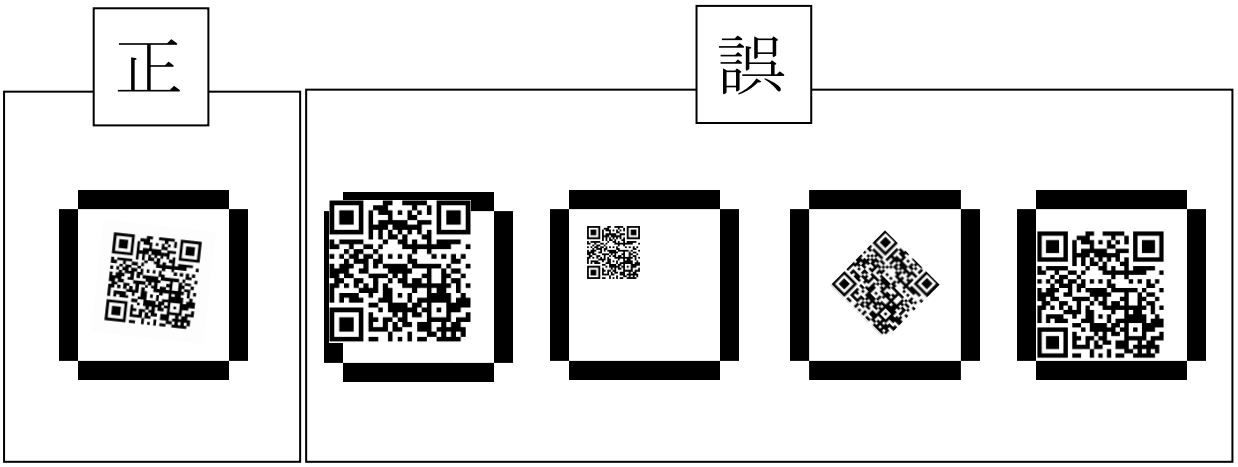
マーカのテンプレートの一部では，QR コードが配置されていないため，マーカの中央部分に，QR コードを歪みなく貼り付けてください．貼り付ける QR コードの一辺の長さは，マーカの一辺に対して 40～60%となるようにしてください．

例：一辺が 50mm のマーカのテンプレートに対して，25mm の QR コードを貼り付ける．



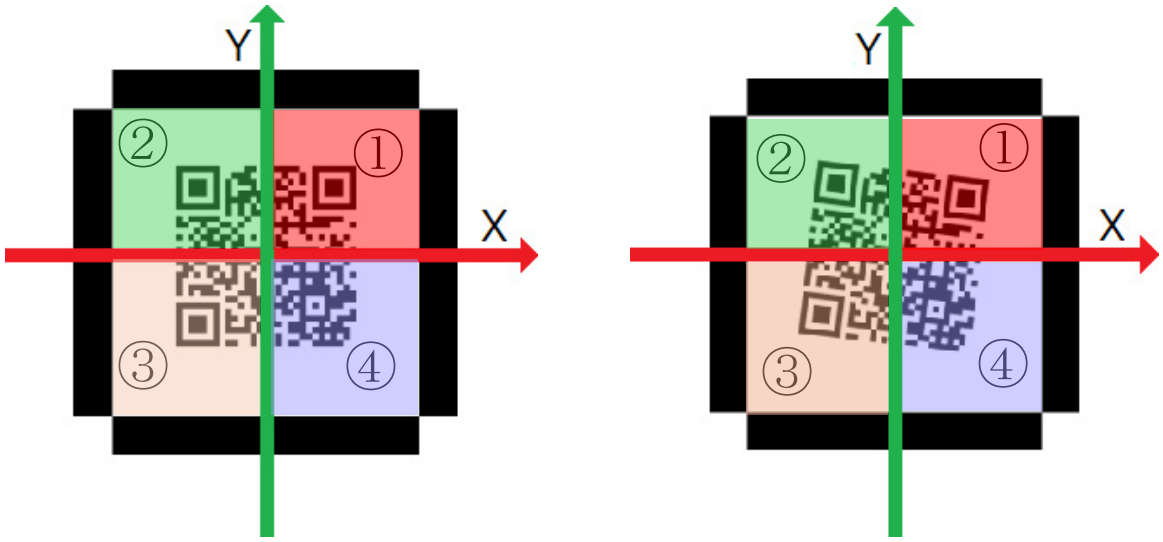
QR コードを貼り付ける際は，以下の図を参考にしてください．また，次のような貼り付け方は避けてください．

- マーカの枠をはみ出すように QR コードを貼り付ける．
- 極端に大きい，または小さい QR コードを貼り付ける．
- 大きく回転した QR コードを貼り付ける．
- QR コードの周囲の余白が狭くなるように貼り付ける．



マーカの座標系の原点は，マーカの中央部分です．原点中心からマーカの枠に垂直に交わるように 4 つ

に区切り，QR コードの左上のファインダパターンがある区切られた場所を第 2 象限，右上のファインダパターンがある区切られた場所を第 1 象限，左下のファインダパターンがある区切られた場所を第 3 象限，どれにも該当しない場所を第 4 象限とします．



2.3. エラーコード

QRCodeCalibration プロバイダでは，以下の固有エラーコードが定義されています．ORiN2 共通エラーについては，「ORiN2 プログラミングガイド」のエラーコードの章を参照してください．

表 2-1 独自エラーコード一覧

エラー名	エラー番号	説明
E_NO_QR_CODE_CALIBRATION_LICENSE	0x835810A0	QR コード位置補正のライセンスがありません．
E_NO_HAND_EYE_CALIBRATION_DATA	0x835810A1	キャリブレーションデータの読み込みに失敗しました．

E_NO_QR_CODE_CALIBRATION_LICENSE：QR コード位置補正のライセンスが正しく登録されていません．コントローラに正しいライセンスキーを入力してコントローラの再起動を行い，ライセンスを有効化させてください．

E_NO_HAND_EYE_CALIBRATION_DATA： 正しいキャリブレーションデータを選択しているか，確認してください．ハンドアイキャリブレーションウィザードでキャリブレーションデータを送信していない場合は，送信してから本プロバイダのコマンドを実行してください．

3. コマンドリファレンス

本章では CaoController::Execute メソッドの各コマンドについて解説します.

3.1. コントローラクラス

表 3-1 CaoController::Execute コマンド一覧

3.1.1. CaoController::Execute("GetMarkerInfo") コマンド

マーカの位置・姿勢を表すポジション型データを取得します. 本コマンドは GetImageAcquisitionPos コマンドを使用せず, 単独で使用できます.

本コマンドから取得したポジション型データをワーク座標系として設定することで, マーカを基準としたロボット動作が可能になります. マーカ基準で動作させる場合には, ポジション型データや同次変換型データを使用してください. ワーク座標系やポジション型データの詳細については, お使いのコントローラのユーザーマニュアルの該当ページをご参照ください. 該当コンテンツ ID は以下の通りです.

表 3-2 ワーク座標系と位置データのコンテンツ ID

項目	ユーザーマニュアルのコンテンツ ID		
	RC8	RC9	COBOTTA PRO
ベース座標系・ワーク座標系	1938	10086	16623
位置データ	1939	10094	16631

書式 qrCodeCalibrationCtrl.Execute("GetMarkerInfo", Array(image, calibrationId, imageAcquisitionPos, markerLength))

image : (VT_UI1|VT_ARRAY)
カメラで撮影したマーカが写った画像.

calibrationId : (VT_I4)
コントローラに送信したキャリブレーションデータのキャリブレーション ID.

imageAcquisitionPos : (VT_R4| VT_ARRAY)

マーカ撮影時のロボットの位置・姿勢を表すポジション型データ.

markerLength : (VT_R4)
マーカ一辺の長さ[mm]. 0< markerLength<= 1000 を指定可能.
省略時は 50.

戻り値 : (VT_VARIANT|VT_ARRAY)
マーカの位置・姿勢を表すポジション型データとマーカ内の QR コードの文字列の VARIANT 配列. マーカの認識に失敗した場合, {0,0,0,0,0,0,0}と""(空白)を返す.

[注意事項]

本コマンドの引数, image に設定する画像については, 以下の点にご注意ください.

- ピントが合っていること.
  - ピントが合っていない画像は認識精度が低下し, 認識に失敗する原因となります.
- マーカ面全体が均一な照明下で撮影されていること.
  - 過度な暗さ・明るさ, 影, 反射がある画像は, マーカの認識が不安定になります.

使用例

```
markerInfo = qrCodeCalibrationCtrl.Execute("GetMarkerInfo", Array(image, calibrationId, imageAcquisitionPos, markerLength))
```

3.1.2. CaoController::Execute("GetImageAcquisitionPos") コマンド

マーカを正面から撮影できるロボットの位置・姿勢のポジション型データを取得します. 本コマンドは GetMarkerInfo コマンドと併用して使用します.

本コマンドで取得した位置・姿勢に移動してから GetMarkerInfo コマンドを実行することで, ばらつきの少ないマーカの位置・姿勢を取得できます. よりばらつきを少なくするためには, 引数 markerRatio を大きな値に設定してください.

本コマンドによって取得したポジション型データの Fig は, デフォルトで Auto Fig(-3)に設定されています. 必要に応じて, LetF コマンドを用いて Fig の値を変更してください.

書式 qrCodeCalibrationCtrl.Execute("GetImageAcquisitionPos ", Array(image, calibrationId, markerPos, markerAngle, markerRatio, markerLength))

image : (VT_UI1|VT_ARRAY)
カメラから取得した画像.

calibrationId : (VT_I4)

コントローラに送信したキャリブレーションデータのキャリブレーション ID.

markerPos : (VT_R4|VT_ARRAY)
マーカの位置・姿勢を表すポジション型データ. GetMarkerInfo コマンドで取得した値を指定する.

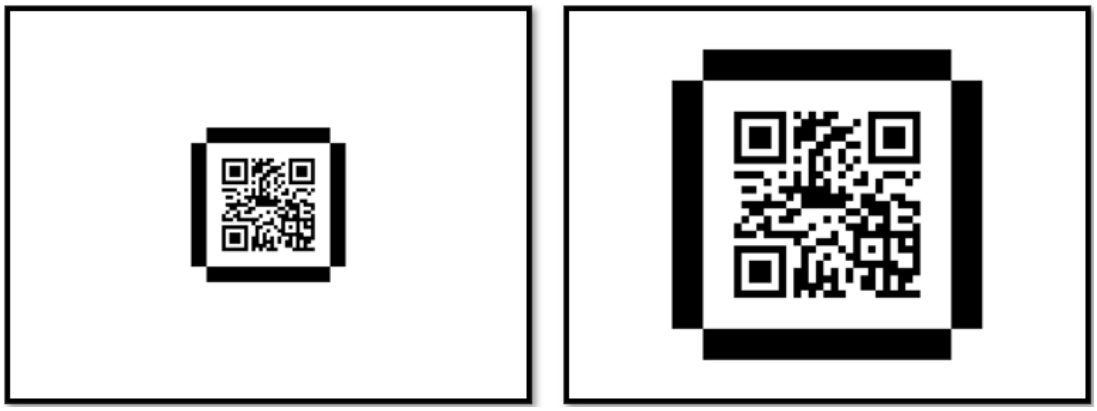
markerAngle : (VT_I4)
画像上のマーカの角度[deg]. 左回りが正. 画像に対するマーカの角度を指定する. 0 <=markerAngle <= 359 を指定可能.

markerRatio : (VT_R4)
画像の短辺を 1 として正規化したときの, 画像上におけるマーカ一辺の長さの比率. 0. 1 < `markerRatio` <= 1 を指定可能. 引数省略時は 0.6.

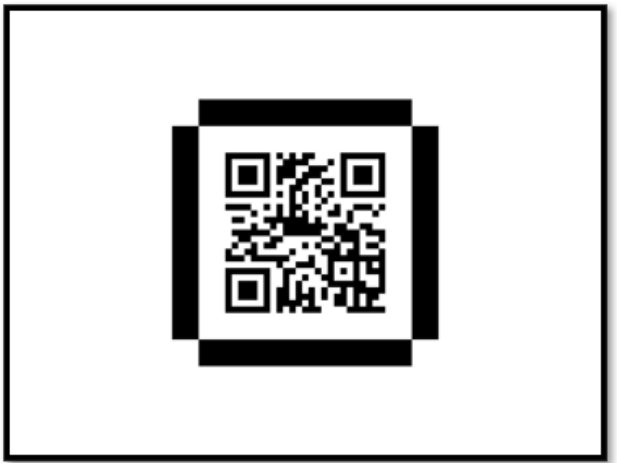
markerLength : (VT_R4)
マーカ一辺の長さ[mm]. 0< markerLength<= 1000 を指定可能. 引数省略時は 50.

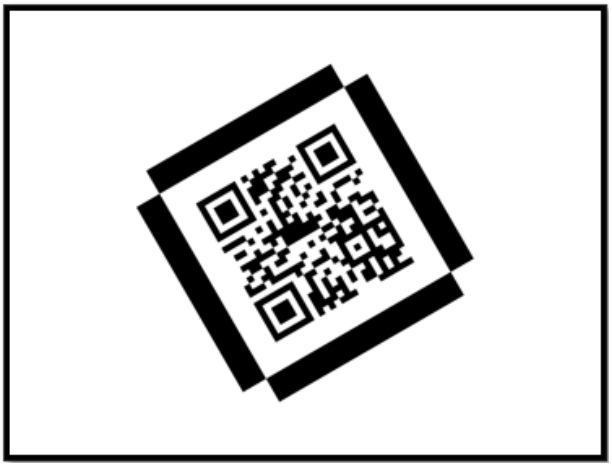
戻り値 : (VT_R4|VT_ARRAY)
マーカを撮影できるロボットの位置・姿勢を表すポジション型データ. Fig は Auto Fig(-3)

[markerRatio の違いによる画像の変化(markerAngle=0)]
(左)markerRatio = 0.4 の場合 (右)markerRatio = 0.8 の場合



[markerAngle の違いによる画像の変化(markerRatio=0.6)]
(左)markerAngle = 0 の場合 (右)markerAngle = 30 の場合





使用例

```
imageAcquisitionPos = qrCodeCalibrationCtrl.Execute("GetImageAcquisitionPos", Array(image, calibrationId, markerPos, 0))
```

4. サンプルプログラム

4.1. GetMarkerInfo

```
!TITLE "GetMarkerInfoSample.pcs"
' 例: マーカの位置・姿勢をワーク座標系番号 1 に設定する.
Sub Main
    ' プロバイダを準備する.
    Dim rilCtrl as Object
    Dim qrCodeCalibrationCtrl as Object
    qrCodeCalibrationCtrl = Cao.AddController("QrCodeCalibration", "CaoProv.DENSO.QrCodeCalibration", "", "")
    rilCtrl = Cao.AddController("RIL", "CaoProv.DENSO.RIL", "", "")

    Dim rilFile As Object
    Dim rilFileOption As String = "ID=201" ' カメラ ID.
    rilFile = rilCtrl.AddFile("Camera", rilFileOption)
    Dim image As Variant
Dim imageAcquisitionPos As Position
' マーカを撮影し, そのときのロボットの位置・姿勢を保存する.
    Call rilFile.Execute("ExtExecSoftTrigger")
    Call rilFile.Execute("ExtRefreshImage")
    image = rilFile.Value
imageAcquisitionPos = CurPos

    Dim calibrationId As Integer = 12  ' ハンドアイキャリブレーションウィザードで設定したキャリブレーション ID.
    Dim markerLength As Integer = 50  ' マーカの一辺の長さ [mm].

    ' マーカの位置・姿勢を求める.
    Dim markerInfo As Variant
    Dim markerPos As Position
Dim markerString As String
    markerInfo = qrCodeCalibrationCtrl.Execute("GetMarkerInfo", Array(image, calibrationId, imageAcquisitionPos, markerLength))
    markerPos = markerInfo(0)
markerString = markerInfo(1)
' markerString を確認し, マーカが認識されたかを判定する.
```

```
    If markerString <> "" Then
    ' ワーク座標系番号 1 にマーカの位置・姿勢を設定する.
    TakeArm Keep = 1
    Dim targetWork As Integer = 1
    ' Work で設定した値は, 電源 OFF 後は保持されない. 電源 OFF 後も保持する場合は, ティーチングペンダントで, 値を設定する.
    Work targetWork, markerPos
    GiveArm
    Else
      MsgBox "マーカを認識できませんでした. ", 2
    End If

End Sub
```

4.2. GetImageAcquisitionPos

```
'!TITLE " GetImageAcquisitionPosSample.pcs"
' 例: マーカを画像中心で撮影できる位置・姿勢に移動させる.
Sub Main
        ' プロバイダを準備する.
        Dim rilCtrl As Object
        Dim qrCodeCalibrationCtrl As Object
        qrCodeCalibrationCtrl = Cao.AddController( "QrCodeCalibration", "CaoProv.DENSO.QrCodeCalibration", "", "" )
        rilCtrl = Cao.AddController( "RIL", "CaoProv.DENSO.RIL", "", "" )
        Dim rilFile As Object
        Dim rilFileOption As String = "ID=201" ' カメラ ID.
        rilFile = rilCtrl.AddFile( "Camera", rilFileOption )

        Dim image As Variant
        Dim imageAcquisitionPos As Position
        ' マーカを撮影する.
        Call rilFile.Execute( "ExtExecSoftTrigger" )
        Call rilFile.Execute( "ExtRefreshImage" )
        image = rilFile.Value

        Dim calibrationId As Integer = 12 ' ハンドアイキャリブレーションウィザードで設定したキャリブレーション ID.
```

```
    ' マーカの位置・姿勢を取得する.
    ' 例:GetMarkerInfoSample.pcs でワーク座標系番号 1 に保存している場合.
    Dim markerPos As Position
    Dim workNumber As Integer = 1
    markerPos = WorkPos( workNumber )

    Dim markerAngle As Integer = 0
    Dim markerRatio As Single = 0.6
    Dim markerLength As Integer = 50

    ' カメラの視野にマーカを収めることができる, ロボットの位置・姿勢を求める.
    imageAcquisitionPos = qrCodeCalibrationCtrl.Execute( "GetImageAcquisitionPos", Array( image, calibrationId, markerPos, markerAngle, markerRatio, markerLength ) )

    TakeArm Keep = 1
    ' カメラの視野にマーカを収めることができる, ロボットの位置・姿勢を求める.
    ' 必要に応じて Fig を変更する.
    ' LetF imageAcquisitionPos = CurFig
    Move P, imageAcquisitionPos
    GiveArm

End Sub
```

付録 A. QR コード位置補正の実施例

次の手順に従うことで，マーカによって補正された位置・姿勢へとロボットを移動させることができます．

1 セットアップ - マーカ基準の動作教示

1-1~1-5 の手順で，マーカ基準の動作を教示します．

1-1 マーカの設置

本書記載の仕様に従い，適切なマーカ（中央に QR コードを貼付済み）を所定の位置に設置します．

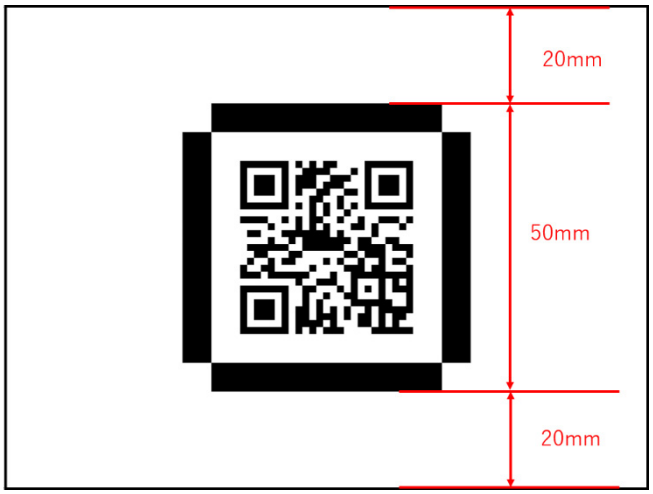
1-2 作業したい位置・姿勢の保存

マーカ基準で作業を行いたいロボットの位置・姿勢を決め，P 型のグローバル変数に保存します．

例) P[11]~P[15]に動作させたいロボットの位置・姿勢をワーク座標系番号 0 で保存します．保存したポジション型データは，後述の「1-5 ポジション型データの変換」で作成したワーク座標系番号の座標に置き換えて使用します．

1-3 マーカ撮影距離の調整とカメラの設定

マーカを撮影できる位置・姿勢にロボットを移動させます．この時，マーカによる補正が適切に行えるよう，撮影距離を考慮します．例として，以下のように撮影した場合，マーカが約±20mm ずれていても補正できます．



移動後は，カメラのピントを調整し，マーカが鮮明に写っていることを確認します．複数の撮影距離で補正を行う場合は，それぞれの位置・姿勢でピントが合っていることを確認します．その後，コントローラに「QR コード位置補正」のライセンスを登録し，ハンドアイキャリブレーションウィザードを実行します．詳細は「2.1 概

要」を参照してください.

1-4 ワーク座標系の登録

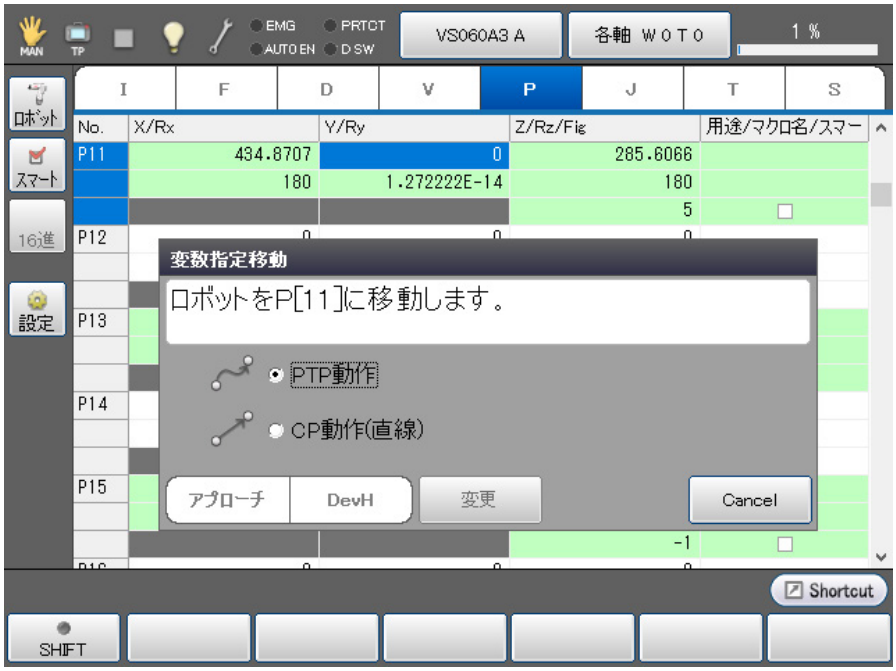
プログラムを用いて, マーカの位置・姿勢を取得し, ワーク座標系として登録します. 本ユーザーズガイド記載のサンプルコードを参考にしてください.

例) 「4.1. GetMarkerInfo」の GetMarkerInfoSample.pcs を実行します. 必要に応じて「4.2. GetImageAcquisitionPos」の GetImageAcquisitionPosSample.pcs を実行し, マーカを撮影できる位置・姿勢を調整します. 調整した場合, GetMarkerInfoSample.pcsを再度実行します. マーカとカメラの距離が「マーカ撮影距離の調整とカメラの設定」で調整した時と同じになるように, GetImageAcquisition コマンドの引数 markerRatio を設定してください.

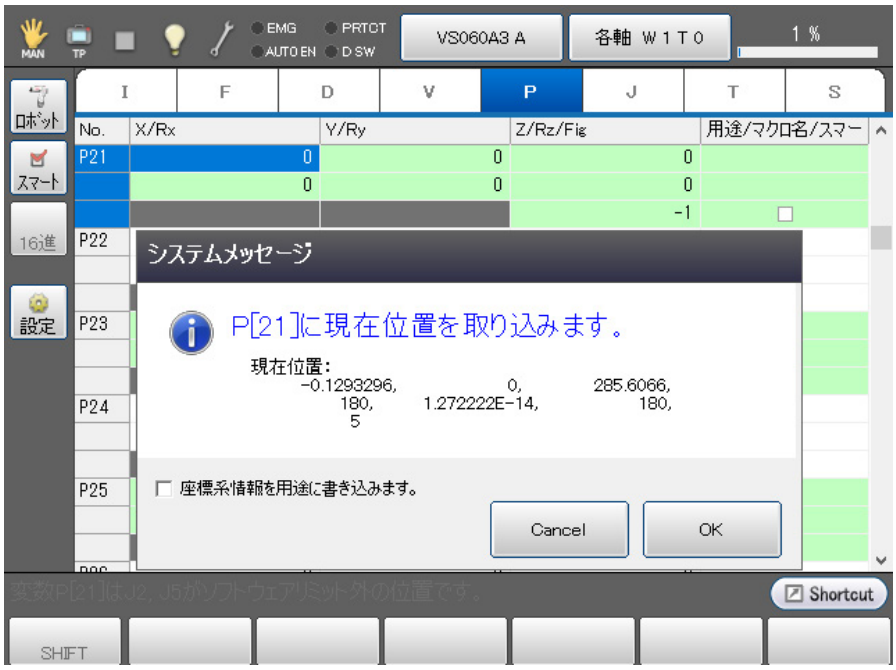
1-5 ポジション型データの変換

「1-2 作業したい位置・姿勢の保存」で保存した P 型のグローバル変数にロボットを移動させ,「1-4. ワーク座標系の登録」で登録したワーク座標系に切り替え, 位置取り込みにより, 別の P 型のグローバル変数に保存します.

以下の図は, i. ワーク座標系番号 0 で位置取り込みを行った P[11]に移動し, ii. ワーク座標系番号 1 で P[21]に位置取り込みを行う例を示しています.



i. ワーク座標系番号 0 で移動



ii. ワーク座標系番号 1 で位置取込

教示点が多い場合はプログラムを用いて座標系の変換を行います.

例) ワーク座標系番号 0 の P[11]~P[15]をワーク座標系 1 に変換して P[21]~P[25]に保存します. 以下はサンプルコードです.

```
Sub Main
    Dim count As Integer
    For count = 0 To 4
        P[21 + count] = ConvertPosWork(P[11 + count], 0, 1)
    Next
End Sub
```

2 稼働 - マーカによる補正動作

補正を行い, ロボットを動作させます. 以下は, ロボットとマーカの相対位置・姿勢が変化した場合を想定したサンプルコードです.

```
Sub Main
        ' 各種パラメータの初期設定.
        ' 最初の撮影位置・姿勢が保存された P 型グローバル変数のインデックス
        Dim firstPosIndex As Integer = 10
        ' マーカのワーク座標系番号
        Dim targetWork As Integer = 1
        ' ワーク座標系 targetWork の位置・姿勢が保存された P 型グローバル変数のインデックス
```

```
Dim markerBasedPosIndex As Integer = 21
' ワーク座標系 targetWork の位置・姿勢が保存された P 型グローバル変数の数
Dim markerBasedPosCount As Integer = 4
' RIL のファイルオプション
Dim rilFileOption As String = "ID=201"
' キャリブレーション ID
Dim calibrationId As Integer = 12
' マーカ一辺の長さ[mm]
Dim markerLength As Integer = 50
' 画像上のマーカの角度[deg]
Dim markerAngle As Integer = 0
' マーカ一辺の長さを 1 とした場合の画像上でのマーカ一辺の長さ.
Dim markerRatio As Single = 0.8

' マーカがカメラ視野に収まる位置・姿勢にロボットを移動させる.
TakeArm Keep = 0
Move P, P[firstPosIndex]
GiveArm

' マーカを撮影し，マーカの位置・姿勢を取得する.
Dim rilCtrl as Object
Dim rilFile As Object
Dim image As Variant
rilCtrl = Cao.AddController("RIL", "CaoProv.DENSO.RIL", "", "")
rilFile = rilCtrl.AddFile("Camera", rilFileOption)

Call rilFile.Execute("ExtExecSoftTrigger")
Call rilFile.Execute("ExtRefreshImage")
image = rilFile.Value

Dim qrCodeCalibrationCtrl As Object
Dim markerInfo As Variant
Dim markerPos As Position
Dim markerString As String

qrCodeCalibrationCtrl = Cao.AddController("QrCodeCalibration",
"CaoProv.DENSO.QrCodeCalibration", "", "")
```

```
	markerInfo = qrCodeCalibrationCtrl.Execute("GetMarkerInfo", Array(image, calibrationId, CurPos, markerLength))
	markerPos = markerInfo(0)
	markerString = markerInfo(1)

	If markerString = "" Then
		MsgBox "マーカを認識できませんでした. ", 16
		Exit Sub
	End If

	' マーカを画像中心で撮影できるロボットの位置・姿勢を取得し, 移動させる.
	Dim imageAcquisitionPos As Position
	Dim imageAcquisitionArg As Variant
	imageAcquisitionArg = Array(image, calibrationId, markerPos, markerAngle, markerRatio, markerLength)
	imageAcquisitionPos = qrCodeCalibrationCtrl.Execute("GetImageAcquisitionPos", imageAcquisitionArg)

	TakeArm Keep = 0
	Move P, imageAcquisitionPos
	GiveArm

	' 再度マーカを撮影し, マーカの位置・姿勢を取得する.
	Call rilFile.Execute("ExtExecSoftTrigger")
	Call rilFile.Execute("ExtRefreshImage")
	image = rilFile.Value

	markerInfo = qrCodeCalibrationCtrl.Execute("GetMarkerInfo", Array(image, calibrationId, CurPos, markerLength))
	markerPos = markerInfo(0)
	markerString = markerInfo(1)

	If markerString = "" Then
		MsgBox "マーカを近距離で認識できませんでした. ", 16
		Exit Sub
	End If
```

```
        ' マーカの位置・姿勢をワーク座標系に設定する.
        TakeArm Keep = 0
        Work targetWork, markerPos
        ChangeWork targetWork

        ' マーカ基準の位置・姿勢へロボットを移動させる.
        Dim count As Integer
        Dim targetPos As Position
        For count = 0 To markerBasedPosCount - 1
                targetPos = P[markerBasedPosIndex + count]
                ' 必要に応じて Fig を変更する.
                ' LetF targetPos = CurFig
                Move L, targetPos
        Next

        GiveArm

        MsgBox "補正動作が完了しました. ", 0
End Sub
```

[補足]

サンプルコードでは Work コマンドでワーク座標系を設定しています. Work コマンドで設定した値は, 電源 OFF 後は保持されないため, 電源 OFF 後も保持する場合は, ティーチングペンダントで, 座標要素を選択し, 「編集」ボタンを押下後, 「OK」ボタンを押下します.

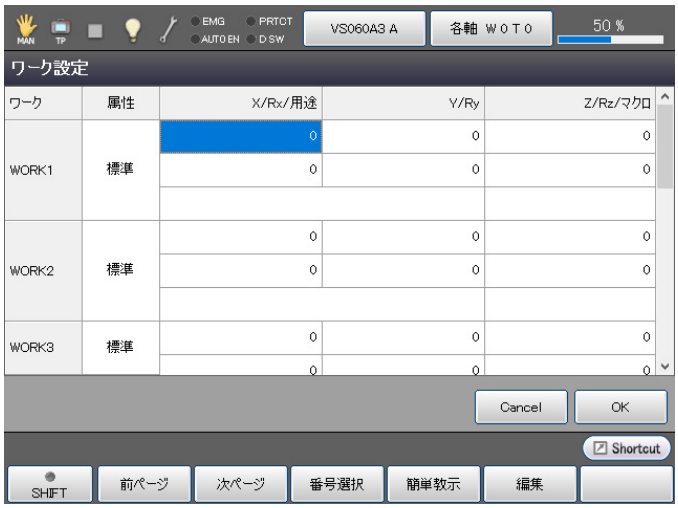


表 A-1 ワーク座標のコンテンツ ID

項目	ユーザーマニュアルのコンテンツ ID		
	RC8	**RC9**	**COBOTTA PRO**
ワーク座標の表示・設定	1743	9701	17450