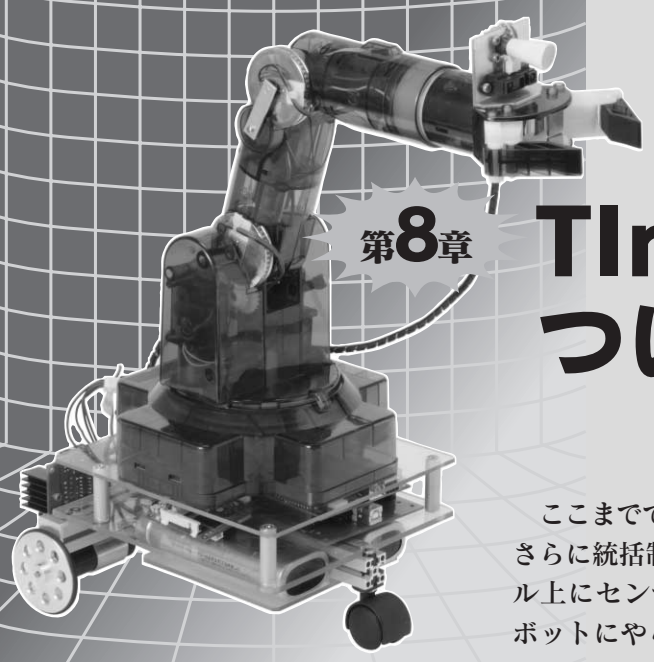

台車とアームをドッキング

第8章 TIrobo01-CQ, ついに完成!!

吉田 智章

ここまでで，台車，台車の走行制御モジュール，センサ・モジュール，アーム，さらに統括制御モジュールのすべてがそろった．残る作業は，統括制御モジュール上にセンサやモータなど，全体の動きを統括制御し，目標とするタスク(ロボットにやらせること)を実現するプログラムを作成することである．　(筆者)

ここでは，ここまでで製作した台車とアームをドッキングして，一つのシステム――アーム付き自律走行ロボットとして動かす方法について述べていきます．いよいよ，TIrobo01-CQ の完成です．

1. 目標とするタスクを細かなところまで定義

● 目標とするタスク――ロボットに何をさせたいか

まず，目標とするタスクを振り返ってみましょう，大ざっぱには，次のとおりでした．

(1) 人間を探す：走行制御モジュール，PSD 距離センサ，焦電型赤外線センサ
(2) 見つけた人間に接近し，物体を受け取る：走行制御モジュール，センサ・モジュール，アーム
(3) 決められたエリアに移動する：走行制御モジュール
(4) 掴んでいる物を置く：アーム
(5) スタート地点に帰還し，人間を探すところから繰り返す

これらについて，細かい部分をより明確に定義しておきましょう．

▶ 人を探す

まず最初は，人間を探します．単に探すといっても，どこをどうやって探すのかを決めておかないと実装しにくくなるので，あらかじめ決めてしまいました．探すのはロボットの前方，距離 60cm 以内左右 45 °の扇型の範囲(**図1**)としました．この範囲で焦電型赤外線センサと PSD 距離センサを使って人間を探します．どちらのセンサも一度の計測ではごく狭い角度しか計測できないので，左右 45 °の範囲を台車を旋回させることで順次調べていくことにします．左右 45 °の範囲を調べても人がいない場合でも，繰り返し同じ範囲を調べ続けることにします．

▶ 見つけた人間に接近し，物体を受け取る

次に見つけた人間に接近します(**図2**)．ロボットはセンサを正面に向けて台車を旋回することで人間を探すので，人間を見つけた時点では，ロボットの正面に人間がいることになります．そこで，そのまま距離センサで人間までの距離を測定しつつ前進して接近します．

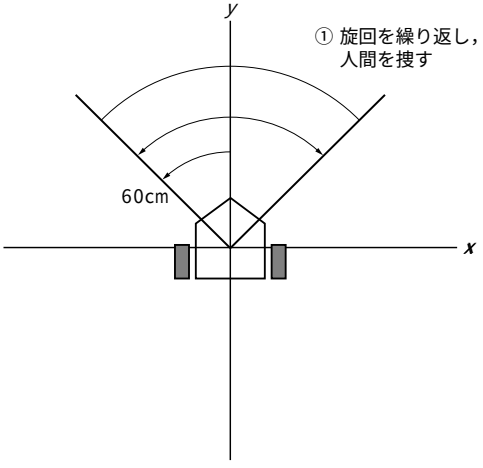

図1
人間を探す領域

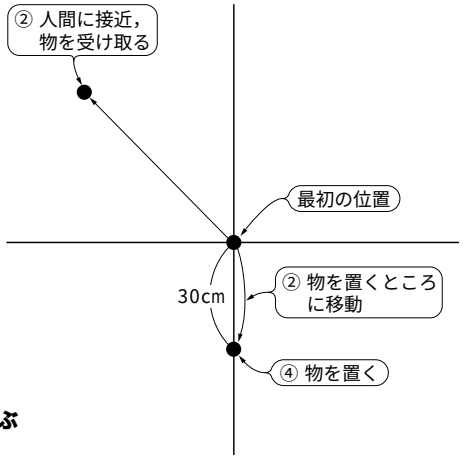

図2
物を受け取って運ぶ
(①は図1を参照)

人間がいる位置までたどり着いたら，グリッパ(2本の指)を閉じて，渡された物を掴みます．残念ながら，このロボットに搭載されているセンサだけでは受け取る物体がどこにあるかまでは認識できないので，人間が物をちゃんと差し出してくれることを期待する――ということにしました．

▶ 決められたエリアに移動する

ロボットは，物を掴んだら，物をもったまま決められたエリアまで移動します．これは台車の走行制御モジュールのおかげで比較的簡単に実装できます．今回は，このエリアをロボットが最初にいた位置の後方30cmということにしました．

▶ 掴んでいる物体を置く

ロボットは最後に，掴んだ物を置きます．グリッパを開くだけでも良いのですが，それでは置くというよりも落とすといった感じになってしまうので，手先を20cmほど下げてからグリッパを開くことにしました．物を置いた時点で物を運ぶというタスクは成功となりますが，今回はここからさらにスタート地点に帰還して，次のタスクに備えることにしました．

● ソフトウェアの構成を決める

この目標タスクを実現するには，台車，アーム，センサのすべてを扱う必要があります．**図3**にこのソフトウェアの構成を示します．

▶ 統括制御プログラム

まず，最上位にあるのが統括制御モジュール用プログラム(統括制御プログラム)です．このプログラムが，センサからの値により，台車やアームに指示を送ります．目標タスクの手順はここに実装します．

▶ 走行制御プログラム

台車を制御するのは統括制御モジュールのシリアル・ポートに接続された走行制御モジュールで，モジュール上には制御プログラムであるtRunCtrlが動いています．統括制御プログラムが走行制御モジュールに指示を送ったり，状態を問い合わせるには走行制御ライブラリrunCtrl_cを利用します．runCtrl_cは，シリアル・ポートで通信する部分を中継プログラムsertcpに任せています．

▶ モータ制御モジュール

アームを制御するのは統括制御モジュールのRS-485インターフェースに接続されたモータ制御モジュールです．

統括制御プログラムがモータ制御モジュールと通信するには，アーム・インターフェース・ライブラリcqarmlibを利用します．cqarmlibは同じくRS-485インターフェースに接続されたセンサ・モジュールと通信する機能も提供します．

cqarmlibはモータ制御モジュール，センサ・モジュールと通信する機能のみが実装されているので，指示に使われる値やセンサ値として報告される値が，モジュール上のA-Dコンバータの値そのものとなります．

そこでcqarmlibの上位に，A-Dコンバータの値を解釈して物理量として扱うための関数群phy.cを用意します．実際に統括制御プログラムがマニピュレータとセンサを扱うにはphy.cを使用します．

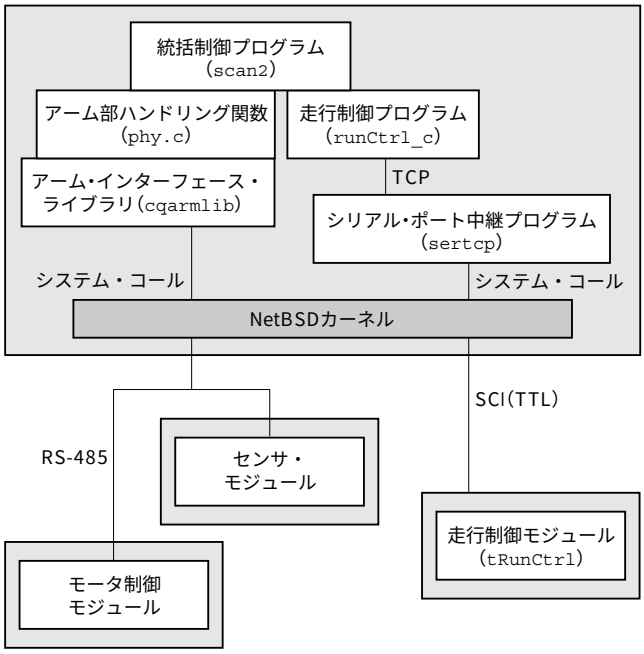

図3　目標タスクを実現するためのソフトウェア構成

結局，統括制御プログラムで台車とアームを扱うには，runCtrl_cとphy.cのインターフェースを使うことになり，シリアル・ポートを介したモジュールとの通信に関する部分は初期化時以外はほとんど意識する必要がありません．runCtrl_c，phy.c，cqarmlibの各ライブラリの関数のうち，このデモ・プログラムで使用するおもな関数を**表1**にまとめます．

2. 目標となるタスクの実装

次に，統括制御プログラムで目標タスクの一つ一つの手順をそれぞれどのように実装するかについて説明します．

● 初期化を行う

最初に，初期化のためのプログラムを**リスト1**に示します．

▶ RS-485の初期化とモータ制御モジュールの制御パラメータの設定

まず，アームのモータ制御モジュール，センサ・モジュールとの通信を行うRS-485の初期化と，モータ制御モジュールの制御パラメータの設定を行います．同様に走行制御モジュールとの通信路の初期化と走行速度，旋回角速度の設定を行います．

走行制御モジュールと走行制御ライブラリは連携してロボットの位置の推定と座標系の管理を行います．プログラムを開始して初期化を実行すると，これらの推定位置およびその座標系は，その時点のロボットの位置と向きが原点にあるとして設定されます．

このとき，runCtrl_cの関数を使えば，たとえばセンサで