ETロボコン公式トレーニング
データロギング

データログを開発に役立てよう！

この教材について

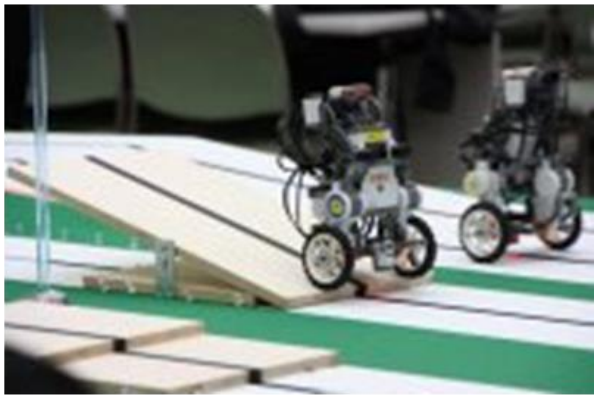
- 目的
  - この教材は、ETロボコンに参加されるみなさんに、開発に必要となる知識やスキル取得の機会を提供することを目的に作成されております
- 著作
  - この教材はETロボコン実行委員会が作成したものです
  - この教材の著作権は、 ETロボコン実行委員会に帰属します
- 使用について
  - ETロボコンの参加資格（企業/大学/個人）の範囲内に限り、ご自由に活用していただいてかまいません

必要な開発・実行環境

- ハードウェア
  - PC
    - Bluetooth通信可能であること
    - PCに通信デバイスがない場合は、USB接続タイプのBluetoothレシーバなどを利用してください
  - 走行体
    - 組立図にしたがって組み立ててください
- ソフトウェア
  - NXTGamepad
    - 以下のURLからダウンロードしてください
      http://lejos-osek.sourceforge.net/download.htm?group_id=196690
  - サンプルプログラム datalogging
    開発環境に含まれています

試走

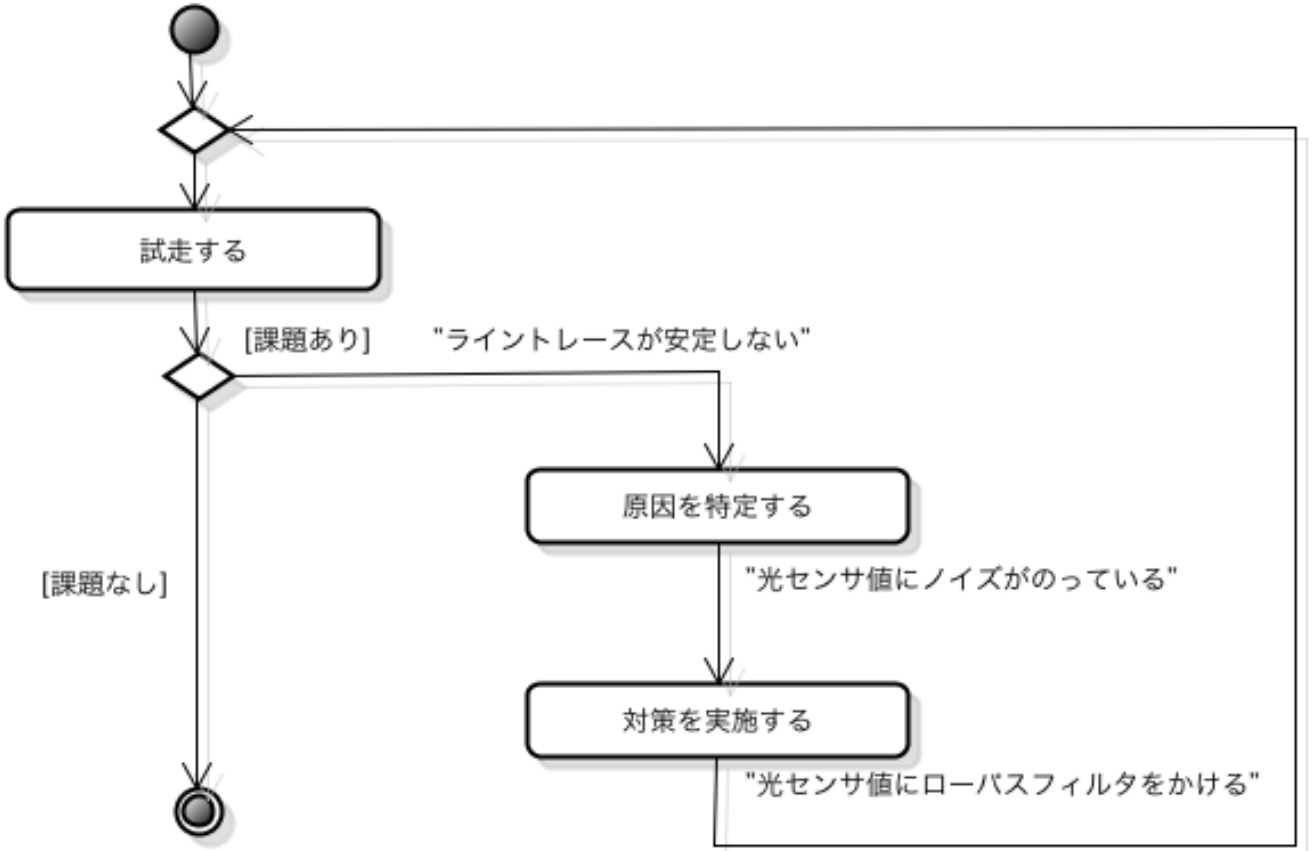
実際にロボットを走行させて、その性能を評価します



試走会の様子

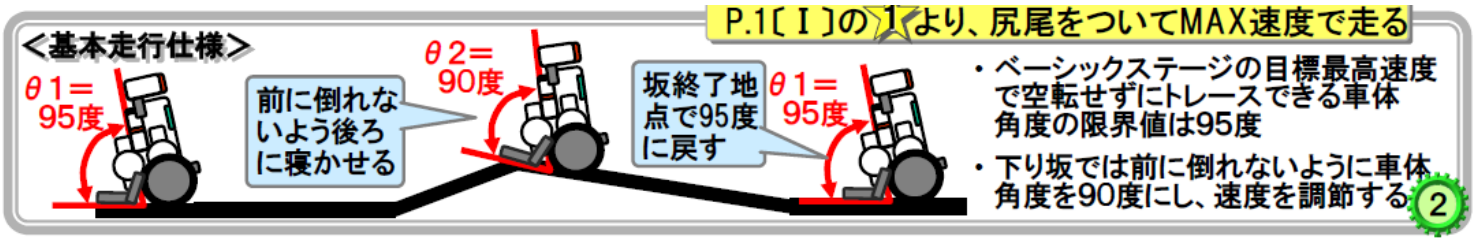
課題の発見と解決

試走では様々な課題が発見されます。
その原因を特定し、解決することが要求されます。



データロギング

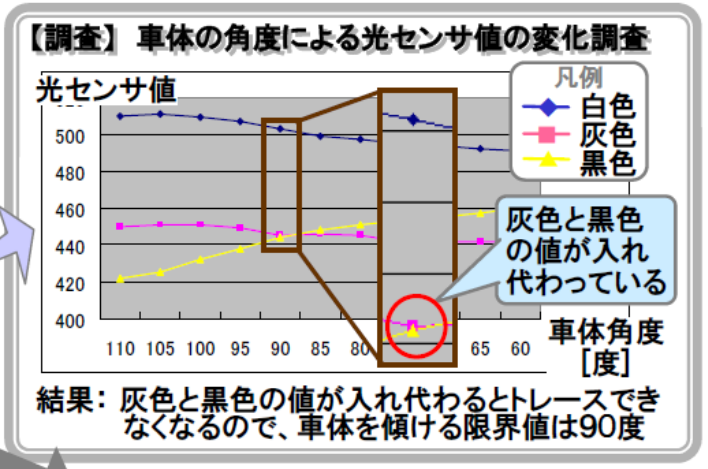
各センサーからの入力値、モーターの動きを数値として連続的に記録し、そのデータを分析することは走行体の動きを把握するための効果的な方法です。



<冗長設計項目>　※検討したミスケース
(1) 光の外乱を受けてトレースできなくなる
(2) 下り坂で倒立制御しないので前のめりになって倒れる
(3) 車体が後ろ加重になるので空転して走行距離がくるう
(4) 速度が速すぎてカーブでコースを逸脱する

対応

(1)の結果：トレースできる車体角度を調査し、安定して走れる角度を決める
(2)の結果：下り坂では、車体を前のめりにならない角度までさらに倒す
(3)の結果：空転しない車体角度と速度のバランスをとる
(4)の結果：カーブで内外輪差をつけて強制的に曲がれるようにする

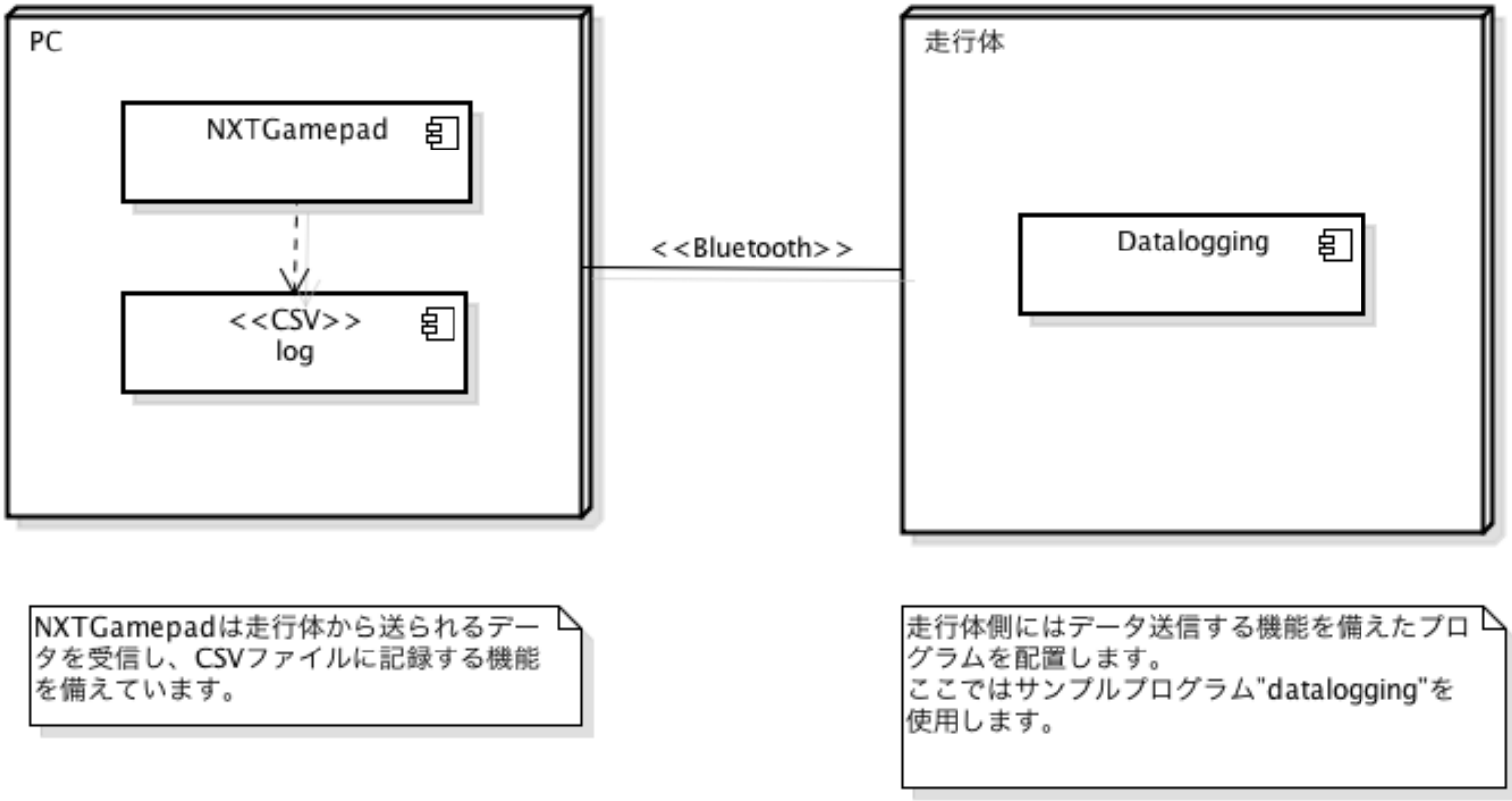


〔Ⅲ〕にて制御仕様を記述

尻尾走行の検討　HELIOSさん

最も簡単なデータロギング方法

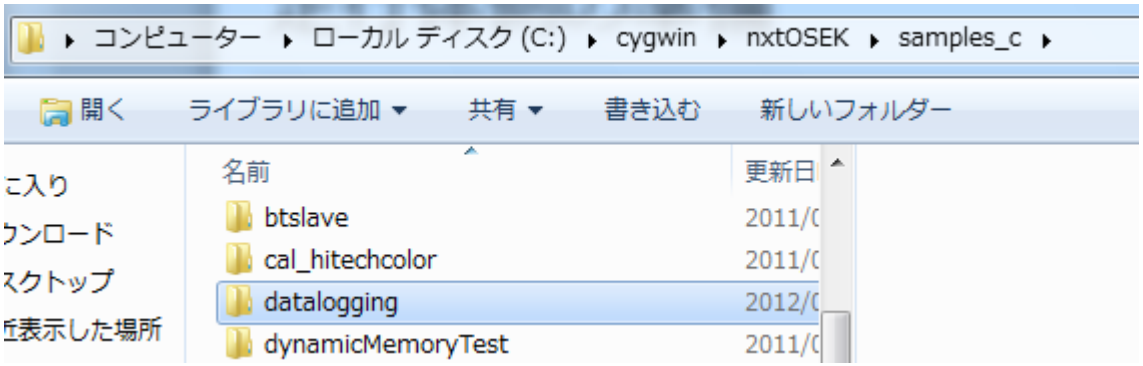
NXTGamepadを使用します。



走行体側の準備

走行体にサンプルプログラムを配置します。

サンプルプログラム“datalogging”は/nextOSEK/samples_Cにあります。



Bluetoothデバイス名が重複しないように、“ET＋チームID”としてください。

datalogging.c に次のコードを追加します。

```
/* LEJOS OSEK hooks */
void ecrobot_device_initialize()
{
    ecrobot_init_sonar_sensor(NXT_PORT_S2);

    // デバイス名を設定する////////////////////////////
    // デバイス名は重複しないように ET + チームIDとする
    if(ecrobot_get_bt_status()==BT_NO_INIT){
        ecrobot_set_bt_device_name("ET500");
    }
    ////////////////////////////////////////////////////

    ecrobot_init_bt_slave("LEJOS-OSEK");
}
```

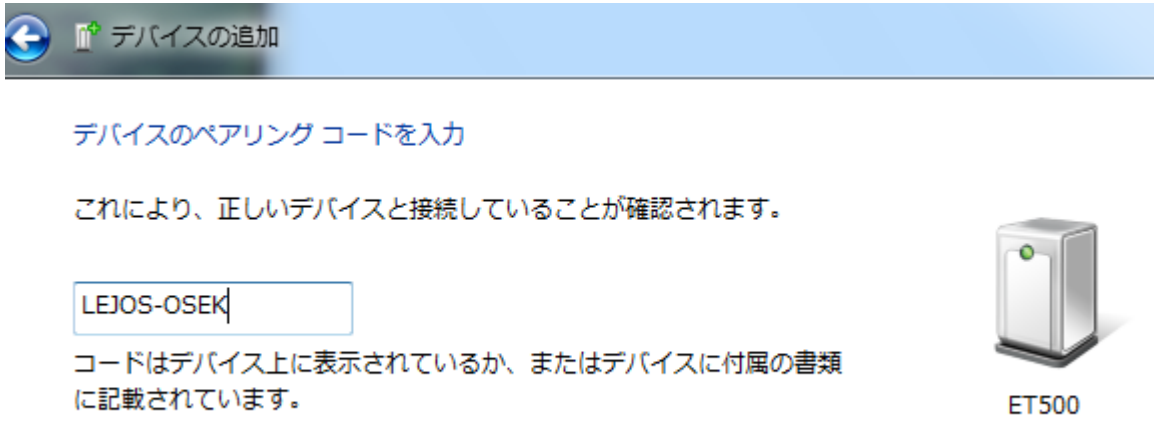
Makeして、走行体にロードします。

ペアリング

①走行体でdataloggingを実行します

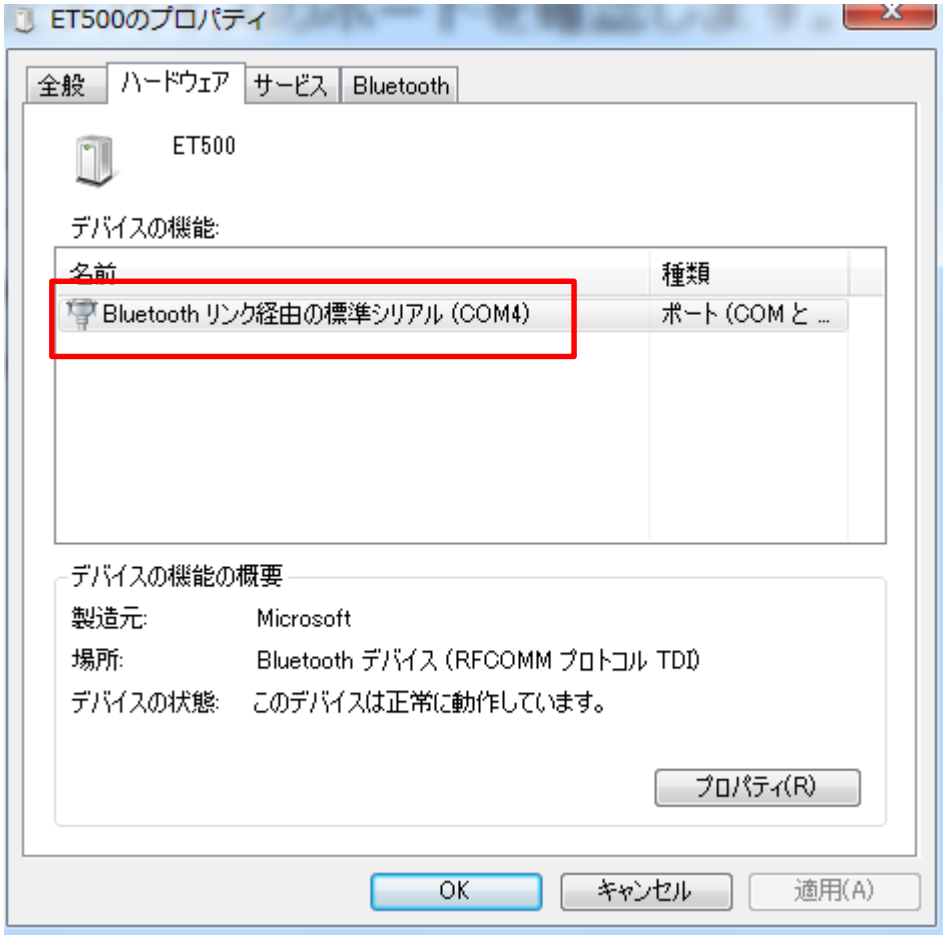
②PCでデバイスを追加します。

コントロールパネルーハードウェアとサウンドーデバイスとプリンタからデバイスの追加を実行します。



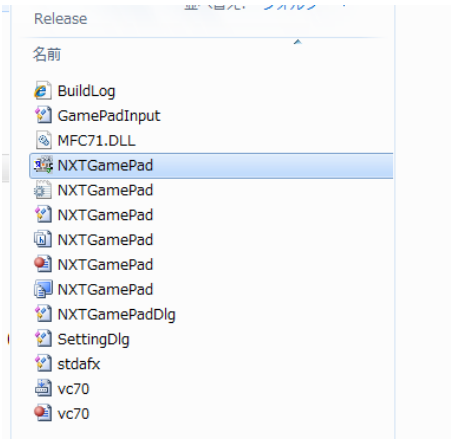
ペアリング

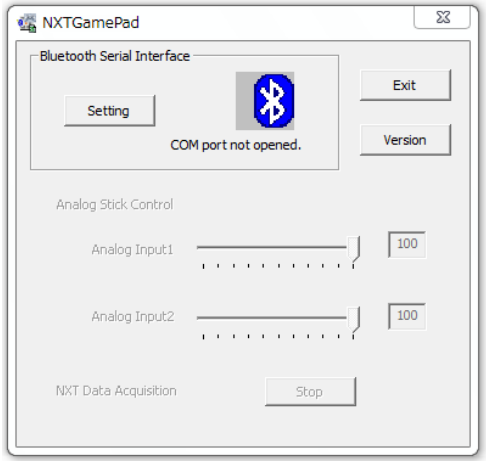
③追加されたデバイスのポートを確認します。



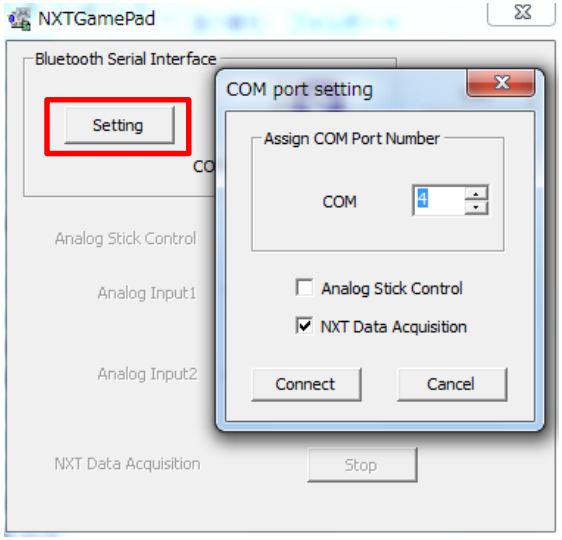
NXTGamepadの実行

Releaseフォルダに解凍された、NXTGamePad.exeを実行します。



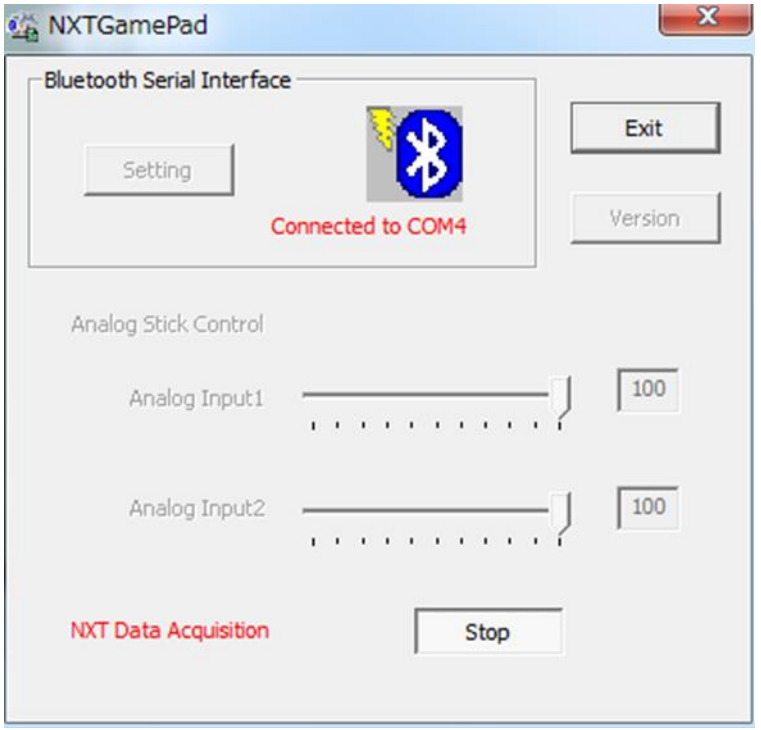


ポートを設定し接続します。ペアリング時に割り当てられたポートを指定します。



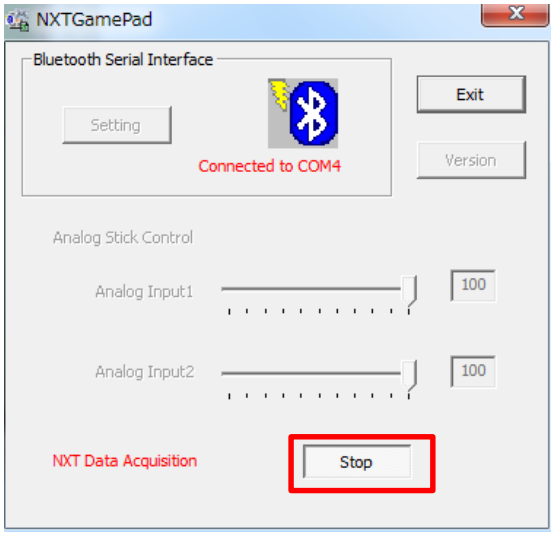
データロギング開始

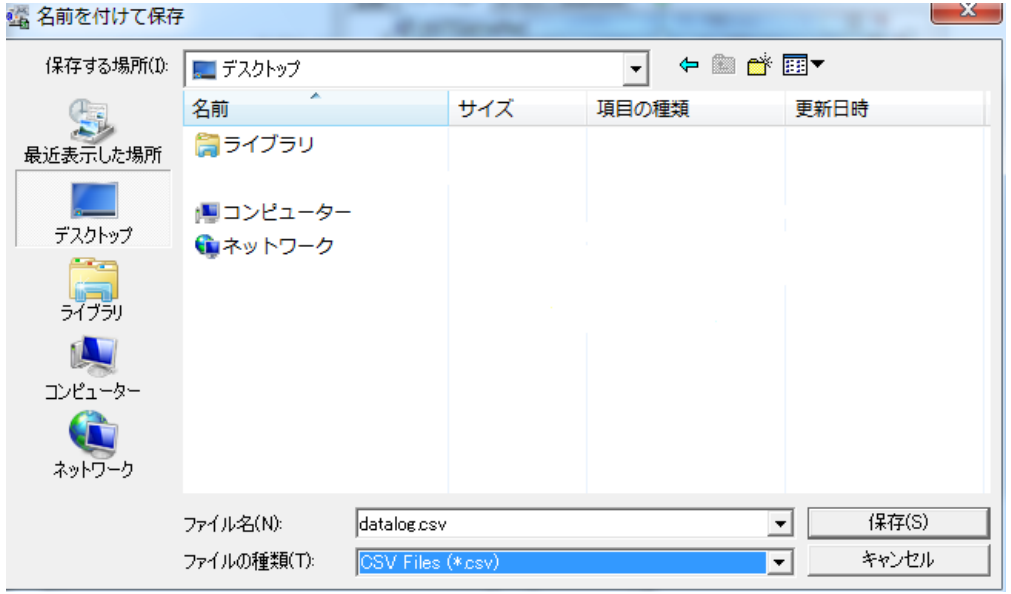
走行体のRunボタンを押すと、PCへのデータ送信が開始されます。



データログの保存

STOPボタンを押すとファイルダイアログが開きます。
ログ出力先のファイル名（CSV形式）を指定し、ログを書き出します。





ログの内容

CSVファイルには次のデータが保存されます。
Time: データ取得タイムスタンプ [msec]
Data1/Data2: ユーザー選択データ
Battery: バッテリ電圧 [mV]
Motor Rev A/B/C: ポートA/B/Cの各モータ回転角度 [度]
ADC S1/S2/S3/S4: ポートS1/S2/S3/S4の各A/Dセンサ生データ
I2C: 超音波センサ距離データ [cm]

CSVファイルに保存されたデータをMicrosoft ExcelやMATLABなどのソフトウェアを用いて解析することも可能です。

	A	B	C	D	E	F	G	H	I	J	K	L
1	Time	Data1	Data2	Battery	Motor Rev A	Motor Rev B	Motor Rev C	ADC S1	ADC S2	ADC S3	ADC S4	I2C
2	0	0	0	8807	0	0	0	607	1023	1023	1023	0
3	50	1	-1	8793	0	0	0	604	1023	1023	1023	255
4	100	2	-2	8793	0	0	0	609	1023	1023	1023	255
5	150	3	-3	8807	0	0	0	607	1023	1023	1023	255
6	200	4	-4	8779	0	0	0	605	1023	1023	1023	255
7	250	5	-5	8765	0	0	0	607	1023	1023	1023	255
8	300	6	-6	8793	0	0	0	605	1023	1023	1023	255
9	350	7	-7	8779	0	0	0	606	1023	1023	1023	255
10	400	8	-8	8779	0	0	0	607	1023	1023	1023	255
11	450	9	-9	8779	0	0	0	607	1023	1023	1023	255
12	500	10	-10	8807	0	0	0	608	1023	1023	1023	255
13	550	11	-11	8793	0	0	0	607	1023	1023	1023	255
14	600	12	-12	8779	0	0	0	609	1023	1023	1023	255
15	650	13	-13	8793	0	0	0	607	1023	1023	1023	255
16	700	14	-14	8793	0	0	0	608	1023	1023	1023	255
17	750	15	-15	8807	0	0	0	608	1023	1023	1023	255
18	800	16	-16	8793	0	0	0	607	1023	1023	1023	255
19	850	17	-17	8793	0	0	0	606	1023	1023	1023	255
20	900	18	-18	8765	0	0	0	605	1023	1023	1023	255
21	950	19	-19	8779	0	0	0	605	1023	1023	1023	255
22	1000	20	-20	8751	0	0	0	607	1023	1023	1023	255

datalogging で使用する関数

関数	説明
U8 ecrobot_set_bt_device_name(const CHAR*bd_name)	Bluetoothデバイス名の設定 引数: 　bd_name: デバイス名(最大16文字) 戻り値: 　1(succeeded)/0(failure)
SINT ecrobot_get_bt_status(void)	Bluetooth接続状態の取得 引数: 　無し 戻り値: 　Bluetooth接続状態 　　BT_NO_INIT (未初期化状態) 　　BT_INITIALIZED (初期化状態) 　　BT_CONNECTED (接続確立状態) 　　BT_STREAM (データ送受信可能状態)
void ecrobot_init_bt_slave(const CHAR *pin)	NXT をBluetooth通信のスレーブデバイスとして初期化し、マスターデバイス(PC, NXTマスターデバイス)との接続を確立します。 引数: 　pin: パスキー交換用ピンコード(最大16文字) 戻り値: 　無し

datalogging で使用する関数

関数	説明
void ecrobot_bt_data_logger(S8 data1,S8 data2)	データロギング用送信API。NXTの全ポートに接続されたセンサおよびモータ(回転角度)のデータおよび内部状態データ(システムタイマー、バッテリ電圧) 等を送信します。 データパケット　0-3バイト: システムタイマー値[msec], データタイプ U32 データパケット　4バイト: ユーザーデータ1, データタイプ S8 データパケット　5バイト: ユーザーデータ2, データタイプ S8 データパケット　6-7バイト: バッテリ電圧値[mV], データタイプ U16 データパケット 8-11バイト: ポートA サーボモータ回転角度[度], データタイプ S32 データパケット12-15バイト: ポートB サーボモータ回転角度[度], データタイプ S32 データパケット16-19バイト: ポートC サーボモータ回転角度[度], データタイプ S32 データパケット20-21バイト: ポートS1 A/Dセンサデータ, データタイプ S16 データパケット22-23バイト: ポートS2 A/Dセンサデータ, データタイプ S16 データパケット24-25バイト: ポートS3 A/Dセンサデータ, データタイプ S16 データパケット26-27バイト: ポートS4 A/Dセンサデータ, データタイプ S16 データパケット28-31バイト: 超音波センサデータ, データタイプ S32 引数: 　data1: ユーザーデータ1 (例, 左側ゲームパッド入力) 　data2: ユーザーデータ2 (例, 右側ゲームパッド入力) 戻り値: 　無し
void ecrobot_term_bt_connection(void)	Bluetooth通信終了処理用API。 引数: 　無し 戻り値: 　無し

技術教育の演習で使用する場合

- モデリング技術教育の演習の際にデータロギングを使用できると非常に効果的です
  - 使用するには、model_impl.c に以下のコードを追加してください

model_impl.c

※**赤太文字部分**が追加箇所です。

```
// デバイス初期化用フック関数
void ecrobot_device_initialize()
{
    // 各デバイスの初期化関数をここで実装することができます
    // ⇒　光センサ赤色LEDをONにする
    ecrobot_set_light_sensor_active(NXT_PORT_S3);

    // ⇒　Bluetooth通信開始処理を行う
    // デバイス名を設定します
    // デバイス名は重複しないように ET + チームIDとします
    if(ecrobot_get_bt_status()==BT_NO_INIT){
        ecrobot_set_bt_device_name("ET500");
    }
    // bluetooth通信のスレーブデバイスとして初期化します
    // 引数はパスキーです
    ecrobot_init_bt_slave("LEJOS-OSEK");
}
// デバイス終了用フック関数
void ecrobot_device_terminate()
{
    // 各デバイスの終了関数をここで実装することができます。
    // ⇒　光センサ赤色LEDをOFFにする
    ecrobot_set_light_sensor_inactive(NXT_PORT_S3);

    // ⇒　Bluetooth通信終了処理を行う
    ecrobot_term_bt_connection();
}
```

```
TASK(TaskMain)
{
    // オブジェクト間のリンクを構築する
    //　（省略）

    // 各オブジェクトを初期化する
    //　（省略）

    // 4ms周期で、ライントレーサにトレース走行を依頼する
    while(1)
    {
        LineTracer_trace(&lineTracer);

        // NXTの全ポートに接続されたセンサおよび
        // モータ(回転角度)のデータおよび
        // 内部状態データ(システムタイマー、バッテリ電圧)
        // 等を送信します。
        ecrobot_bt_data_logger(0, 0);

        systick_wait_ms(4);
    }
}
```

最後に

　データロギングにより走行体の動きを把握することにより、走行性能改善の検討を効率的に行うことができます。
　今回はもっとも簡単な方法を紹介しましたが、開発を進めていくと、さまざまな機能が欲しくなると思います。
　このような開発環境の整備にも力を注ぎ、開発の効率化を図ることをお勧めします。

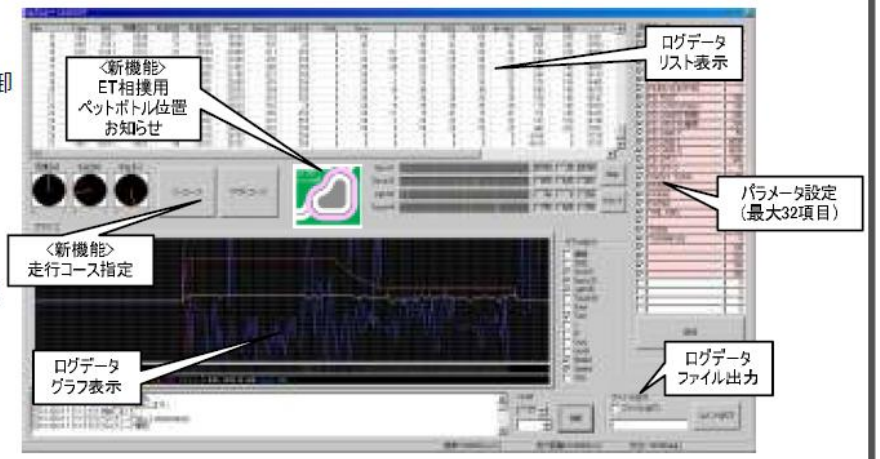
開発支援環境

＜自作デバッグ支援ツール「logDisp2.0」＞

2009年に誕生した「logDisp」は、2010年に「logDisp++」と進化し、さらに2011年、新機能を加えた「logDisp2.0」として生まれ変りました！

■新機能
・IN/OUTコースを選択できるスタート制御
・ペットボトルの場所を指定可能
・ショートカット決行スイッチ追加
→2011年ルール/走法に対応！！

■既存機能
・各種センサ値のリアルタイム表示
・ログファイル保存/復元による開発支援
・各種パラメータ変更の即時反映
・ログデータグラフ表示
・走行軌跡表示
→開発効率大幅UP！！



おやじプログラマーず　さん