4G-05

水中ドローンの実験環境の開発

島袋　真成†　　早川　栄一†
拓殖大学工学部情報工学科†

1. 研究の背景と目的

水中ドローン OpenROV[1]では、ソースコードが公開されており、独自に機能を開発することができる。現在、OpenROV には自律制御の機能がない。そこで自律制御の機能を開発したい要求がある。しかし、開発した機能を検証する際、水中ドローン本体を水中で動作させることは、基板などの浸水による機器故障のリスク、検証場所が限られているという問題点がある。

本研究の目的は、水中ドローンの動作をシミュレートすることにより、機能開発及び検証を効率的に実施できる実験環境を提供することである。

2. 研究概要

OpenROV のシステムは BeagleBoneBlack(以下 BBB)と Arduino mega(以下 MEGA)で構成する。BBB は Web サーバとして機能する。ユーザは PC のブラウザから BBB にアクセスし、水中ドローンの操作やカメラ映像のモニタリング、取得したセンシングデータの閲覧を行う。BBB と MEGA が互いに通信し、ユーザの操作コマンドは BBB から MEGA へ送信される。MEGA は各デバイスの制御とセンサから取得した情報を BBB へ送信する。

次に OpenROV に新しい機能を追加する事例として、任意の水深まで潜降後、水面に浮上する機能を挙げる。

(1) この機能を開発する際、まず水深を検知する必要がある。デバッグの際にはユーザが指定した水深データを使用する。OpenROV のセンサは MEGA が管理するため、ここに使用する水深データを送信する。センサの代わりに別のボードを使用し、指定した値を送信する。

(2) 指定した水深になったことが確認できたら、今度は浮上させるためのコマンドを送信する。そのコマンドの送信を確認するため、ボード間の通信を監視する。

(3) 浮上のコマンドが送信できていることを確認後、モータの動作を検証する。OpenROV のデバイスの代わりに、他のボードで信号を取得することで、どのデバイスが制御されているのか確認する。

これらの検討を行い、本研究では OpenROV の機構を利用し、次の三つの機能を実装する。

(1) データの送信：OpenROV にユーザが指定したデータを送信する。

(2) 通信の監視：OpenROV に使用される二つのボードの通信状況を監視する。

(3) デバイスの制御確認：OpenROV に接続されているデバイスの制御処理を確認する。

3. 実験環境の設計

図 1 に実験環境の構成図を示す。本研究では、BBB と MEGA、Raspberry Pi(以下 RPi)を二つ、および Arduino uno(以下 UNO)を使用する。BBB と MEGA で OpenROV のシステムを構成し、RPi で通信の監視とデバイスの制御確認を行う。UNO はデータの送信の際に使用する。すべての機能を一つの RPi で管理し、GUI アプリケーションとしてユーザに提供する。

RPi は Python、UNO は C/C++ベースの Arduino 言語で開発する。

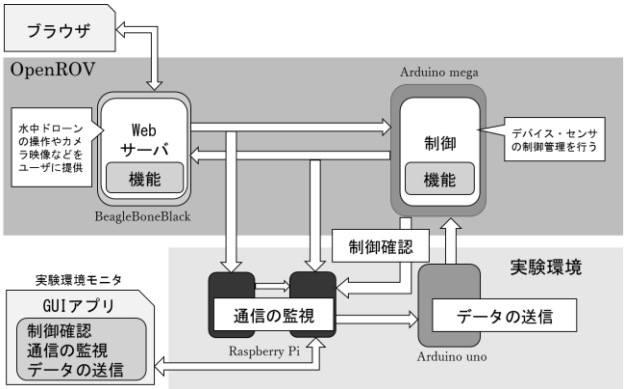

図 1 OpenROV を基にした実験環境の構成

3.1 データの送信

OpenROV に使用されるセンサを UNO で代用してセンシングデータを送信する。OpenROV のセンサは I²C 通信で接続されている。MEGA がマスタ、センサがスレーブとなる。センサの代わりに他のボードからデータを送信する。RPi

Development of experimental environment of underwater drone
Masanari SHIMABUURO†, Eiichi HAYAKAWA†
Department of Computer Science, Faculty of Engineering, Takushoku University†

にはスレーブとなる機能がないため、UNO を使用する。RPi から UNO にデータを送信し、UNO が MEGA にそのデータを送信する。水深、温度、圧力、加速度の値を変更できる機能を提供する。

3.2 通信の監視

OpenROV の通信を監視するため、二つの RPi を使用する。RPi にはシリアル通信の受信ポートが一つしかないため、二つ使用する必要がある。それぞれが通信の情報を取得し、RPi 同士を通信させて、一つの RPi で情報を管理する。これにより、BBB と MEGA の通信の状況を確認する。

3.3 デバイスの制御確認

水中ドローンのモータやライトなどのデバイスはすべて MEGA で管理する機構になっている。BBB から操作コマンドが送信されると MEGA の出力ポートから信号が出力されデバイスを制御する。デバイスの代わりに RPi を使用し、信号の取得をすることでデバイスの制御処理を確認する。本システムではモータの制御を確認する。モータが浮上、前後左右、どの動作状態かを表示する。

また本システムでは潜降の動作を水深の値にフィードバックすることで、デバイスの制御処理に合わせて水深の値が変化する。

3.4 GUI アプリケーション画面

図 2 に GUI アプリケーションの画面を示す。実装した三つの機能を GUI アプリケーションとしてユーザに提供する。水深の変化を容易に把握するため、中央に OpenROV のイメージ画像を表示する。左側に BBB と MEGA から取得した情報を表示する。上部が BBB で下部が MEGA である。右上にはデバイスの制御確認を表示する。右下はデータを送信する部分である。各テキストボックスに数値を入力し、対応するボタンをクリックするとデータの値が送信される。デバッグ中に潜降動作を一時停止させるために、停止ボタンを設置する。

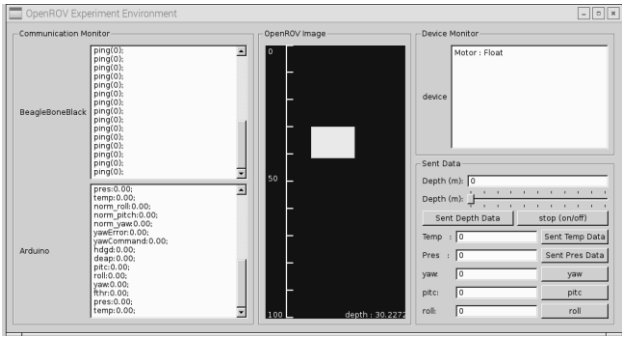

図 2GUI アプリケーション画面

4. 実験環境の検証

4.1 検証用機能の開発

実験環境を用いた検証用機能の開発として、ユーザの指定した水深まで自動で潜降後、警告を表示し動作を停止するシステムを開発する。

この検証用機能はブラウザ画面で水深を指定し、自動潜降のボタンを押したら、潜降動作が開始する。BBB 側で水深を一定時間ごとに監視し、指定した水深の値になったらユーザに警告を表示する。その後、潜降動作を停止する。

この検証用機能は Javascript で開発する。

4.2 実験環境による検証

制作した検証用機能を実験環境で検証する。

ユーザの指定した水深まで潜降することを確認する。ブラウザ画面で水深を指定し、自動潜降ボタンをクリックする。OpenROV 上で水深が変更され、実験環境の画面でも水深が沈んでいくのを確認できた。その際、BBB から MEGA 側に潜降のコマンドが送信されることも確認できた。その後、ユーザの指定した水深まで到達すると、OpenROV のブラウザ画面に警告が表示される。これで水深の値は取得でき、プログラム上で処理できているのが確認できた。

最後にモータが動作していることを確認する。浮上のコマンドが送信されると、水中ドローンの上部についているモータを作動させる処理が開始する。その信号を取得し、水深の値がフィードバックされ、水深の値が変更されていることが確認できた。

5. おわりに

今回、開発した実験環境を用いることで、水深の値を利用するシステムの開発を、水中に沈めることなく実験を行えた。また BBB からのコマンドを確認し、モータを動作させる信号を取得したので、プログラムの処理が正しいことを確認できた。

今後の課題は次のとおりである。

- ボード間の通信速度によるタイムラグを無くし、よりリアルタイムに近づける
- より高度なシミュレーションと連携させる

参考文献

[1] OpenROV JAPAN オフィシャルサイト (参照日:2016/12/13)
http://www.openrov-japan.com/