TUMSAT-OACIS Repository - Tokyo University of Marine Science and Technology (東京海洋大学)

# クラウドサービスを活用した遠隔操船システムの開発



| メタデータ | 言語: jpn<br>出版者:<br>公開日: 2022-08-12<br>キーワード (Ja):<br>キーワード (En):<br>作成者: 土田, 侑<br>メールアドレス:<br>所属: |
| --- | --- |
| URL | https://oacis.repo.nii.ac.jp/records/2504 |

修士学位論文

# クラウドサービスを活用した
# 遠隔操船システムの開発

2021年度

（2022年　3月）

東京海洋大学大学院

海洋科学技術研究科

海洋システム工学専攻

土田　侑

修士学位論文

# クラウドサービスを活用した
# 遠隔操船システムの開発

2021年度

（2022年　3月）

東京海洋大学大学院

海洋科学技術研究科

海洋システム工学専攻

土田　侑

# 目次

第1章 緒言 1
1.1 研究背景 1
1.2 研究目的 4
1.3 本論文の構成 4
第2章 らいちょうⅠにおける遠隔監視・操船システム 6
2.1 らいちょうⅠ 6
2.2 らいちょうⅠにおける遠隔監視・操船システム 11
第3章 クラウドサービスを利用した遠隔監視・操船システム 15
3.1 概要 15
3.2 Node-REDを利用した遠隔監視・操船システム 16
3.2.1 システム構成 16
3.2.2 データベース 17
3.2.3 セキュリティ 19
3.2.4 実験 19
3.2.5 結果，考察 22
3.3 Webページを利用した遠隔監視・操船システム 30
3.3.1 システム構成 30
3.3.2 hostingサービス 32
3.3.3 セキュリティ 32
3.3.4 実験 34
3.3.5 結果，考察 34
3.4 安全装置 39
3.4.1 船側での安全装置 39
3.4.2 遠隔地側での安全装置 39
第4章 オートパイロット機能の開発 43
4.1 概要 43
4.2 制御方法 43
4.3 実験 46
4.4 結果，考察 47
5. 結言 50
謝辞 51

# 第1章 緒言

## 1.1 研究背景

現在世界中のほとんどの物流が，貿易量においては国際貿易の大半が船舶の海上輸送により運搬されている[1]．資源に乏しい島国の日本においても，原材料のほとんどを船によって輸入しており，輸出入のほぼ100%を船舶による輸送で担っている[2].そのため海上輸送が日本の貨物の輸出入において重要である．最近でも新型コロナウイルスによる海上輸送費の上昇によって，原材料の値上げや商品の休売などの事態が起こっていることは記憶に新しい．一方で海難の事故原因割合は図1.1に示すように，見張不十分が778隻，操船不適切が525隻，船体機器整備不良が389隻であり，人為的要因に起因するものが全体の約7割を占めている[3]．また内航船では船員不足が続いており，図1.2に示すように50歳以上の割合が5割を超え高齢化が進んでいる．この高齢化のため若年層の船員確保に取り組んでいる[4]．これらの理由や船員の労働条件改善，運航の効率化などを目的に，近年自動運航船の開発が世界的に行われている．自動運航船に関連する技術としてAIによる自動操船システムの研究[5]や，入出港，離着岸の航行の自動化の研究[6]が行われている．また自動運航船普及にともない，自動運航船関連技術に対する安全性評価の研究[7]も行われている．

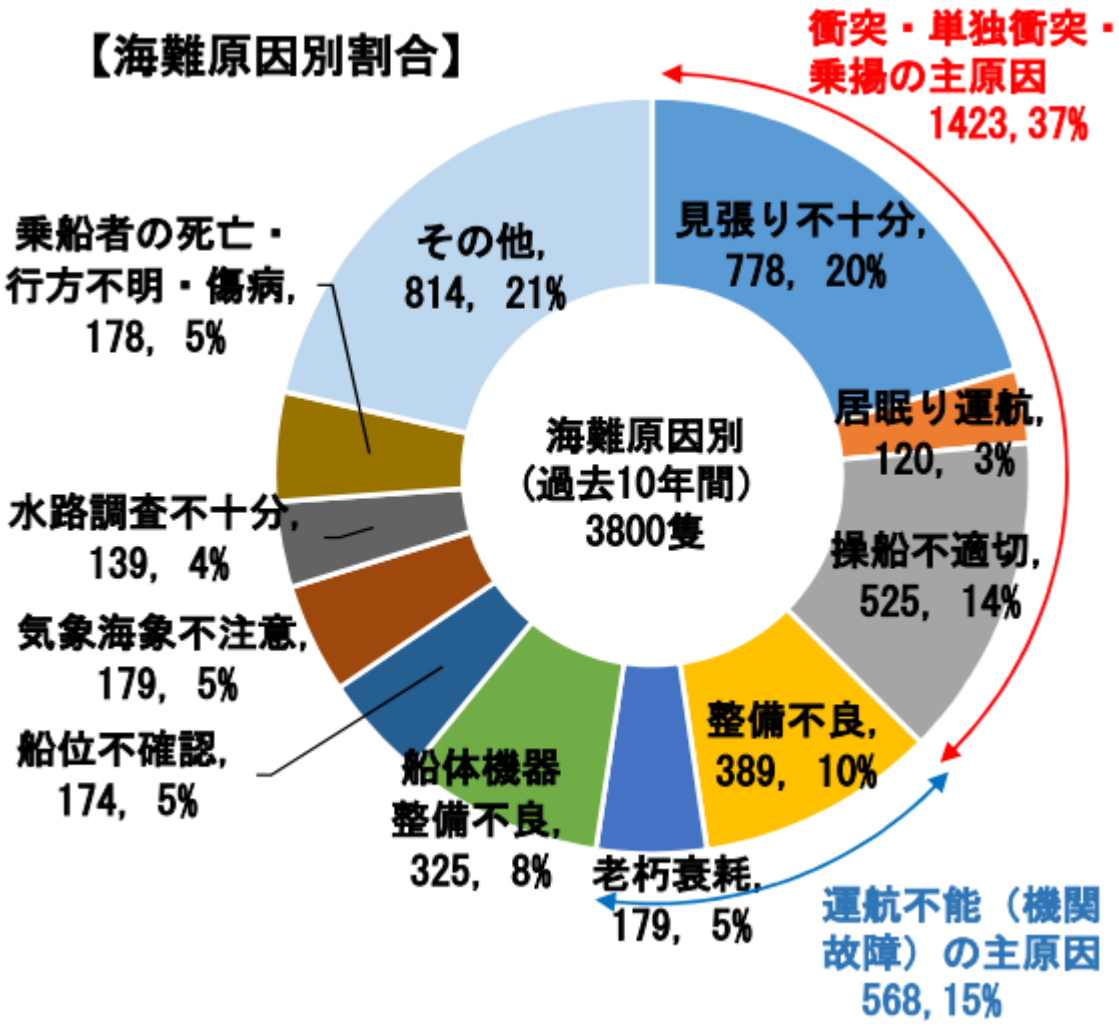
【海難原因別割合】
衝突・単独衝突・乗揚の主原因
1423, 37%
見張り不十分, 778, 20%
居眠り運航, 120, 3%
操船不適切, 525, 14%
整備不良, 389, 10%
老朽衰耗, 179, 5%
運航不能（機関故障）の主原因
568, 15%
船体機器整備不良, 325, 8%
船位不確認, 174, 5%
気象海象不注意, 179, 5%
水路調査不十分, 139, 4%
乗船者の死亡・行方不明・傷病, 178, 5%
その他, 814, 21%
海難原因別（過去10年間）3800隻

図1.1　海難原因別割合

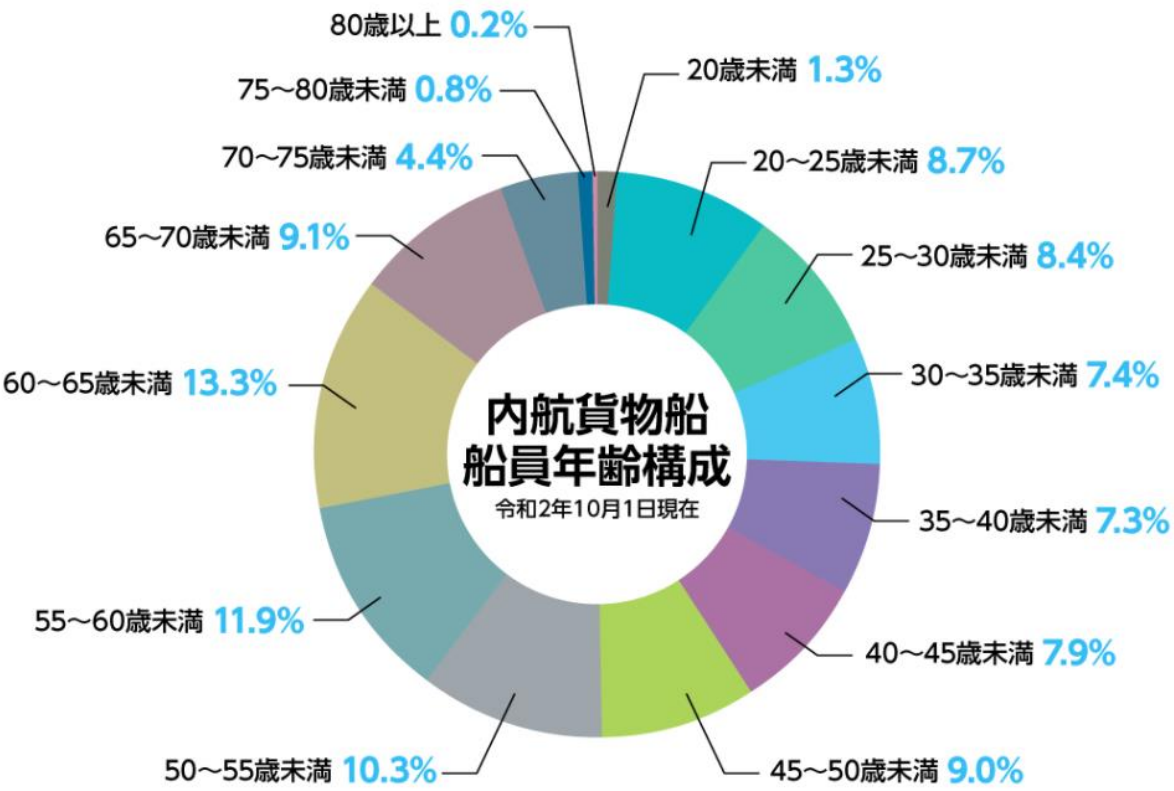


図1.2　内航貨物船船員年齢構成

図1.3は自動運航船システムの一例である．自動運航船は主に自律運航船，コントロールセンター，環境監視システム，通信システムの4つで構成されている．自律運航船は自律で障害物などを認識し，状況に応じてコントロールセンターからの指令を受け船舶を運航する．コントロールセンターは自律運航船を監視するとともに，状況に応じて自律運航船に対して操船指令を送信する．通信システムは自律運航船とコントロールセンターの間で情報のやり取りを行う．環境監視システムは航路上の水上や桟橋の上などに設置され，気象や海象状況を取得するセンサや航路全体を俯瞰できるようなカメラを装備する．収集された情報はコントロールセンターや自律運航船に送信される．本研究室では遠隔監視・操船システムの開発を行ってきた[8].

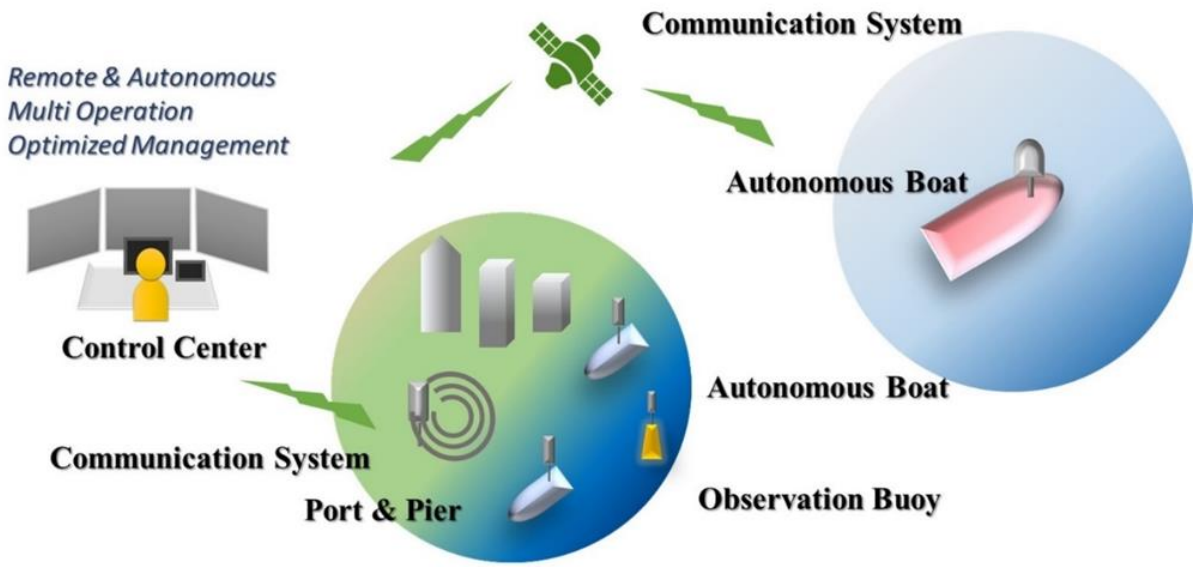


図1.3　自動運航船のシステム構成

## 1.2 研究目的

既存の遠隔監視・操船システムは船舶内に設置されたローカルPCに対して，Chrome Remote Desktop(CRD)を使用することで遠隔監視，操船を実現していた．しかしCRDはウィンドウの情報をそのまま送信しているため，通信量を無駄に消費している可能性がある．船舶は陸上と異なり通信環境が整っていないため，通信量の多さは通信回線を圧迫してしまい他の機器に影響を及ぼす．また航行中に得た航海データはローカルPCにしか保存されず，水がかかることやその他の破損によってデータが消失する恐れもある．そのため通信量も削減し，かつ安全で比較的容易に導入することが可能なクラウドサービスを利用した遠隔監視・操船システムの開発を行う．

## 1.3 本論文の構成

第2章では研究で使用するらいちょうⅠについて述べる．またそのらいちょうⅠで使用している遠隔監視・操船システムについて説明する．

第3章ではクラウドサービスを利用した遠隔監視・操船システムについて述べる．遠隔監視・操船システムをクラウドサービスの”Firebase”を使用し開発を行った．Node-REDを使用したシステムとWebブラウザを使用したシステムの各構成や通信についてそれぞれ説明する．そのうえで既存の遠隔監視・操船システムとの比較実験を行った．それぞれの通信量や各データの表示速度の遅れ，舵，推進器への指令の遅れを計測した．

第4章ではオートパイロットについて述べる．PID制御式を使用し，船舶の船首方位を維持するような制御方法を説明する．そしてらいちょうＩを使用し制御を行った結果を示す．

第5章では本論文のまとめを述べる．

# 第2章 らいちょうⅠにおける遠隔監視・操船システム

## 2.1 らいちょうⅠ

本研究では，遠隔操作機能を有している東京海洋大学の実験船「らいちょうⅠ」を使用して実験を行った．らいちょうⅠは，モータを動力，リチウムイオン電池を動力源とし，航行中に排気ガスを排出しない電池推進船である．図2.1にらいちょうⅠの外観を示す．主要目を表2.1に示す．らいちょうⅠは電動機や電池を交換して竣工後も改修されている．この主要目は2021年1月現在のものである．図2.2にらいちょうⅠの主要機器の系統図を示す．この図2.2の青線は信号線，赤線は強電線を示している．

図2.1　らいちょうⅠ

表2.1　らいちょうⅠ　主要目

| | |
|---|---|
| 全長 | 10[m] |
| 全幅 | 2.3[m] |
| 全深さ | 1.2[m] |
| 電動機推進器定格出力 | 45[kW] |
| 電動機定格トルク | 88[N・m] |
| インバータ動作保証電圧 | DC 230-430[V] |
| 最大速力 | 15[knot] |
| 電池容量 | 40[kWh] |

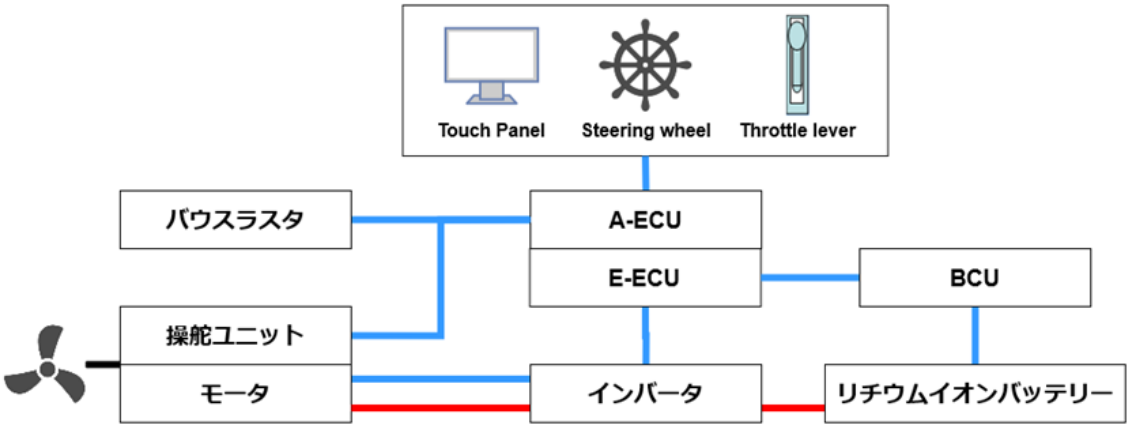


図2.2　らいちょうⅠ主要機器系統図

らいちょうⅠはE-ECU（Engine Electric Control Unit）とA-ECU（Actuator Electric Control Unit）の2つのコントローラに制御されている．E-ECUは主にリチウムイオン電池やモータ，インバータなどと接続されている．E-ECUを図2.3に示す．A-ECUは操舵装置に接続されており，また様々なモード切り替えの機能を持っている．E-ECU，A-ECUそれぞれの構成モジュールは表2.2，表2.3のようになっている．A-ECUには2つのCPUモジュールがある．高速数値演算や通信処理を行うRTOS-CPUモジュールと高速，大容量シーケンス処理を行なうシーケンスCPUモジュールの2つがある．上述の各機器は主にシーケンスCPUモジュールによって管理されている．シーケンスCPUモジュールの詳細を表2.4に示す．らいちょうⅠのECUは横河電機製のものを使用している．データの取得は，Ethernetケーブルを使

用したLANまたは，RS-232Cでネットワークを構成しデータの送受信を行う．らいちょうＩにおいて，各ECUはEthernetケーブル経由で接続して船内のネットワークを構築する．ECUと接続されたモータ，インバータ，リチウムイオン電池といった各機器の1秒毎のデータを取得し，パソコン内に保存することができる．らいちょうＩに搭載したサテライトコンパスや水深計測用の魚群探知機から得られる緯度経度データや水深データはECUを介さずに直接パソコンにUSBケーブルを接続し，受信データを変換するプログラムを介してパソコン内に保存する．らいちょうＩの全体のシステム概要図を図2.4に示す．ここでは，ECUを一つにまとめて表している．この図2.4の青線はEthernet，赤線はCAN通信，緑線はRS232Cでの通信を示している．

図2.3　ECU

表2.2　E-ECU構成モジュールの機能

| 構成モジュール | 機能 |
|---|---|
| 電源モジュール | 外部より電源を供給 |
| シーケンスCPUモジュール | シーケンス制御を行う |
| CAN通信モジュール | CAN通信を行う |
| アナログ入出力モジュール | 4-20mA出力を読み込む |
| 温度入力モジュール | 温度センサと接続 |
| デジタル接点入出力モジュール | 各種接点入出力の書き込み，読み込み |

表2.3　A-ECU構成モジュールの機能

| 構成モジュール | 機能 |
|---|---|
| 電源モジュール | 外部より電源を供給 |
| RTOS-CPUモジュール | リアルタイムOS |
| シーケンスCPUモジュール | シーケンス制御を行う |
| CAN通信モジュール | CAN通信を行う |
| アナログ入出力モジュール | 4-20mA出力を読み込む |
| デジタル接点入出力モジュール | 各種接点入出力の書き込み，読み込み |

表2.4　シーケンスCPUモジュール仕様

| | | |
|---|---|---|
| 製造 | | 横河電機（株） |
| 演算方式 | | ストアードプログラム，折り返し演算 |
| 入出力方式 | | リフレッシュ方式/ダイレクト入出力命令 |
| プログラム言語 | | オブジェクトラダー言語 |
| 命令数 | 基本命令 | 40種 |
| | 応用命令 | 432種 |
| 処理速度 | 基本命令 | 0.00375μs/命令～ |
| | 応用命令 | 0.0075μs/命令～ |
| プログラム容量 | | 最大260Kステップ |
| プロジェクト容量 | | 最大520Kステップ |
| 最大入出力点数 | | 8192点 |
| デバイス容量 | 内部リレー | 65535点 |
| | データレジスタ | 65535点 |
| | ファイルレジスタ | 262144点 |
| | キャッシュレジスタ | 524288点 |
| 通信ポート | | USB2.0，Ethernet |
| メモリーカードスロット | | SDメモリカード |
| 消費電流 | | 800 [mA]（5 [V] DC） |
| 外形寸法 | | W 28.9，H 100，D83.2 [mm] |
| 質量 | | 150 [g] |

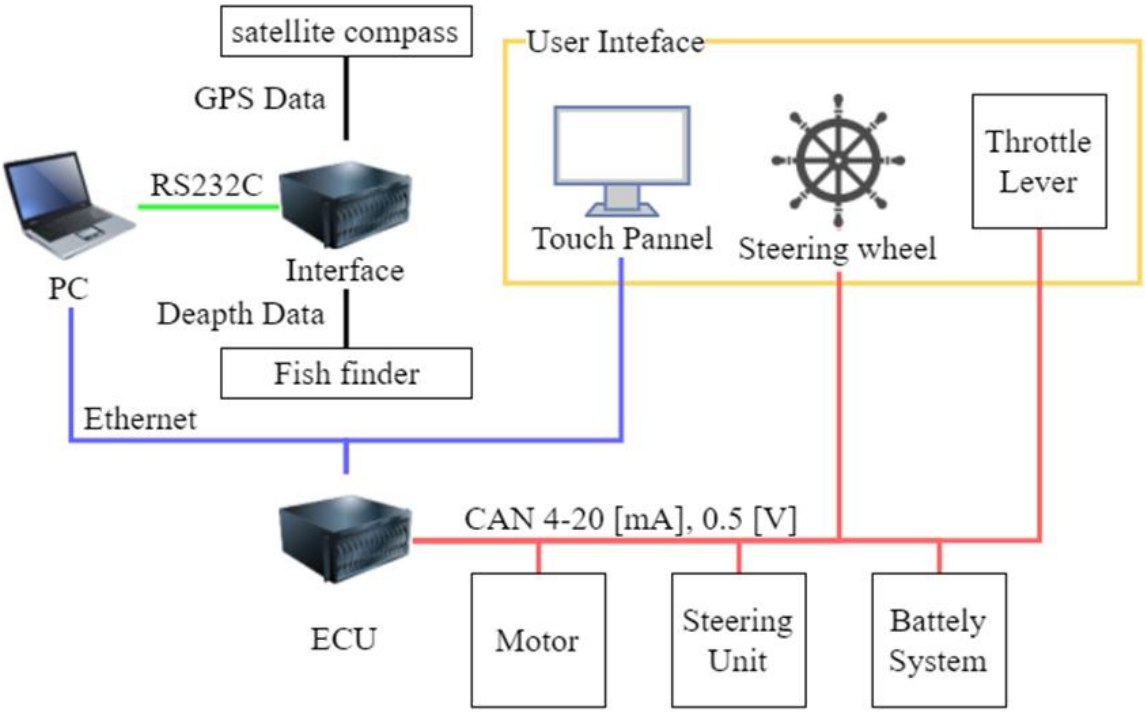


図2.4　らいちょうＩのシステム構成図

## 2.2 らいちょうＩにおける遠隔監視・操船システム

図2.5にらいちょうＩにおける遠隔操船のシステム構成を示す．実線部はEthernetケーブル等を使用した有線でのネットワークを示し，破線部はWi-Fi通信や公衆回線の無線でのネットワークを示す．

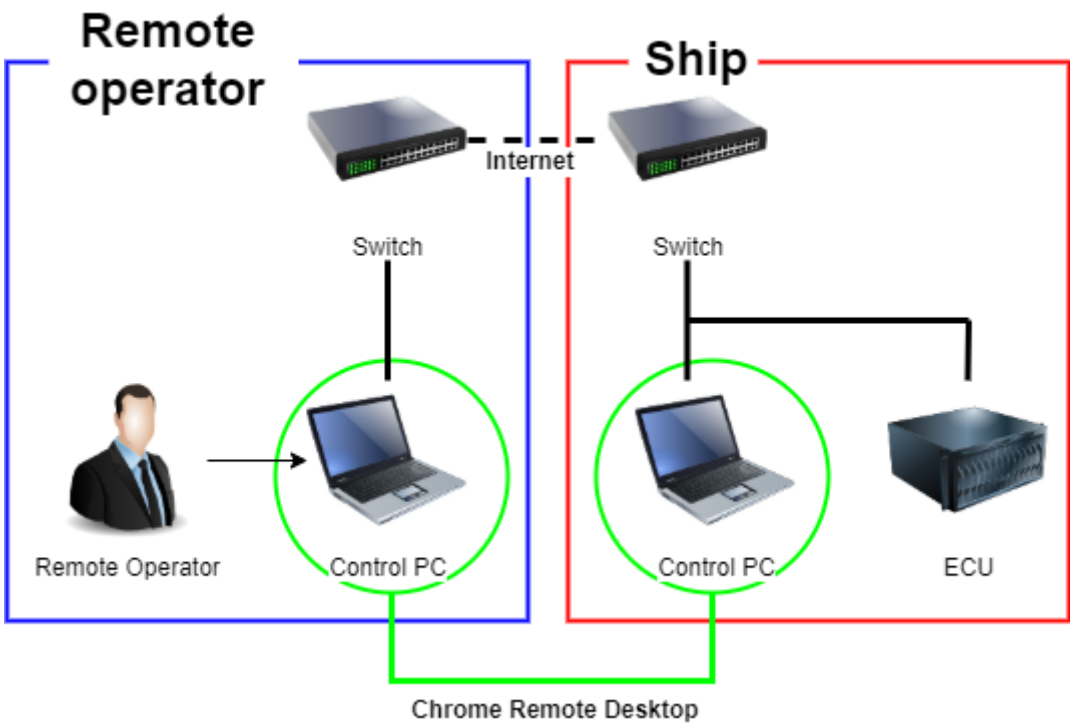


図2.5　らいちょうⅠの遠隔監視・操船システム構成

船側と遠隔地側に設置されたPCの２つによって，遠隔監視・操船システムを実現している．船内のPCでは主に2つのECUに対して通信を行うことや演算処理を担うPythonと，ヒューマンインタフェースを制御するためのNode-REDの２つが動作している．

PythonはECUとTCP/IP通信を行うことで，ECUが各機器から収集し内部レジスタに保存している操船指令や機関，航海情報を習得している．そしてNode-REDに船舶内のデータを受け渡している．一方でNode-REDのユーザーインターフェースから渡された指令を受けECUに送信することで，推進器や操舵装置に出力調整や操舵指令を指令し，操船を行っている．

Node-REDはヒューマンインタフェースとして操作画面を制御している．図2.6に操作画面を示す．この操作画面は，マウス操作によって遠隔操船を行う．操作画面には航行データが表示され，1秒毎に更新される．またには舵や出力の操作が行えるボタンが設置されており，このボタンに1つ1つに対応する操作指令値をPythonに受け渡している．

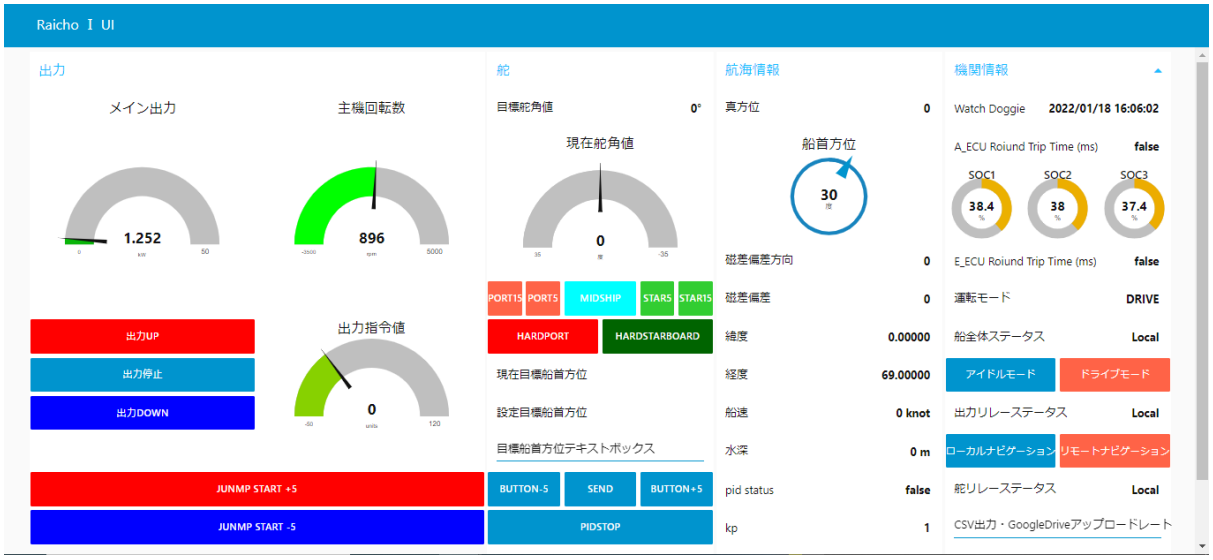


図2.6　らいちょうⅠにおける遠隔監視・操船システムユーザーインターフェース

Node-REDはフローベースドプログラミングツールである．これは図2.7のようなノードと呼ばれるものをつなぐことでプログラミングを行う．そのため視覚的な表現との相性が非常に良く，従来のプログラミング言語と違い，より幅広い範囲のユーザーにとって使いやすいものとなっている．したがって誰でも簡単に操作画面を変更することが可能となる．PythonだけでECUとの通信やユーザーインターフェースの表示を全て行うのでなく，Node-RedとPythonの2つを使用することで，利用者が通信や演算処理などの重要な部分を変更せず，ユーザーインターフェースの変更を簡単にしている．

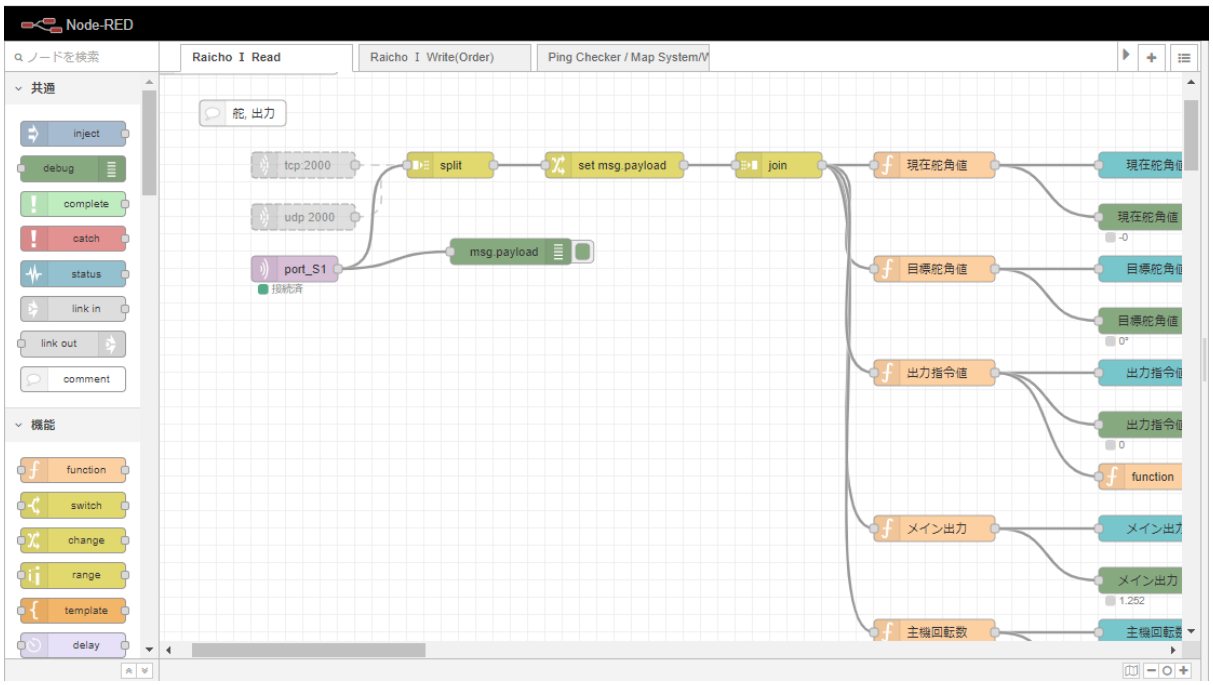


図2.7　Node-REDプログラミング例

一方遠隔地側のPCではChrome Remote Desktopを使用することで，らいちょうⅠに設置されたPCに対して操作を行っている． Chrome Remote Desktopはリモートデスクトップ機能を有しており，離れた場所からPCにアクセスすることが出来る．この機能で船側に設置されたPCにアクセスすることによって，図2.8のように船側に設置されたPCに表示されている操作画面を遠隔地側から操作を行う．遠隔地側のPCと船に設置されたPCには同じ監視・操作画面が映っているため，遠隔操船と通常操船の切り替えはらいちょうⅠ側，遠隔地側のどちらでも行うことが可能な構成となっている．

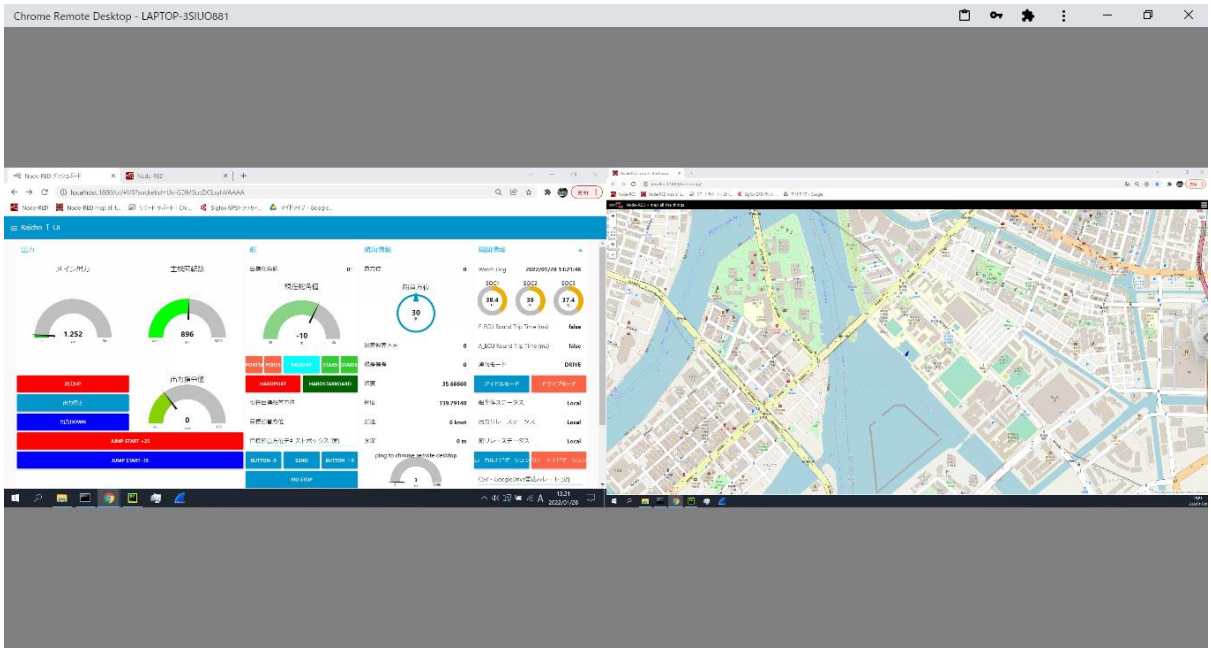


図2.8　Chrome Remote Desktopを使用した遠隔監視・操船システム

遠隔地と船との間の通信には携帯電話回線網を使用している．

# 第3章 クラウドサービスを利用した遠隔監視・操船システム

## 3.1 概要

既存の遠隔監視・操船システムは2台のPCを利用している．船舶にNode-REDとPythonを使用した操作画面が表示されたPCを設置している．遠隔地ではChrome Remote Desktopを，遠隔地に設置されたPCで使用している．これを使うことで遠隔地から船舶のPC画面見ることができ，遠隔監視・操船を可能としている．Chrome Remote Desktopの通信量はクライアント側のウィンドウサイズに依存しているため，ウィンドウサイズが例えば1920*1080ピクセルのいわゆるフルHD場合と3840*2160の4Kのものとを比較した場合，単純計算ではあるが4倍程4Kサイズの方が通信量は大きくなると考えられる．またChrome remote desktopではウィンドウの画面全体をクライアントとやり取りしているので，現在の遠隔監視・操船システムの場合，航海データや機関データなどが必要であり，その他のヒューマンインタフェースの部分などはホストとクライアントの間で必要なく通信量が増えている可能性がある．衛星回線はSpaceX社のStarlinkなど現在様々なサービスが提供され計画されているが，陸上と比べ未だ整ってはいないため，通信量が低い方が良いと考えられる．また自動運航船は現在も開発が進められており，今後様々な機器が搭載され陸上やコントロールセンターなどと通信を行う可能性もある．そのため今後のことも考慮した場合，より少ない通信量で遠隔地側と船舶側で通信を行った方が良いと考えられる．

また舵や位置情報などの航海データのログについての問題である．既存のシステムでは定期的に航海データや機関データを収集しログとしてパソコン内に保存していた．しかし船舶は海に浮かび航行しているため沈没などの問題が起こり，PCが水にぬれたりする場合が考えられる．その時にPC内部に保存されたデータが消失してしまう恐れがある．海難が発生した場合，その事故原因を究明するために航行データの保存がどの船舶であっても必要であると考えられる．さらに従来のような航海データ記録装置を使用している場合，装置の回収が必要であるが，クラウドサービスに航行データを保存する場合は，直ちにデータを取得することが可能である．

本論文ではこれらの問題を抱えているChrome Remote Desktopを使用した既存の遠隔監視・操船システムを改善するために，クラウドサービスを利用した遠隔監視・操船システムの開発を行った．本研究で構築したシステムは，船舶と遠隔地での情報の共有にクラウドサ

ービスの１つであるGoogle　Firebase内のデータベースを使用した．Firebase内にはFirestore DatabaseとRealtime Databaseの２つがありサービス内容が異なるため，それぞれについて遠隔監視・操船システムの開発を行なった．またUIについても，使用者がUIを変更しやすいNode-REDと，様々なデバイスで使うことができるWebページを使用したシステムをそれぞれ構築し，様々な要求に応えられるようにした．

## 3.2 Node-REDを利用した遠隔監視・操船システム

### 3.2.1 システム構成

Node-REDを利用し構築した遠隔監視・操船システムのシステム構成図を図3.1に示す．

今回構築したシステムでは自前でサーバを用意するのではなく，クラウドサービスである”Firebase”を利用している．クラウドサービスとは従来は利用者が手元のコンピュータで利用していたデータやソフトウェアを，ネットワーク経由でサービスとして利用者に提供するものである[9]．クラウドサービスにはGmailなどのWebメールサービスやAmazonが提供するAmazon Web Service(AWS)など様々なものがある．クラウドサービスは導入や管理が容易であることや，初期投資が安価であることがメリットとして挙げられる．

図3.1がNode-REDを使用した遠隔監視・操船システムのデバイス構成である．船舶と遠隔地にそれぞれにPCをそれぞれ設置している．図3.2のように船舶に設置されたPCはECUとLANを形成している．また各PCはインターネットに接続され，データベースとの通信を行っている．船舶に設置されたPCはECU内に保存されている航行データをTCP/IPによって取得し，データベースへ取得した航行データを追加している．遠隔地のPCではデータベースに登録されているデータをPythonによって取得し，そのデータをNode-REDに受け渡すことによってUI表示を行っている．遠隔地から操船指令を送信する場合，まずUIから得た操船指令をPythonによってデータベースへ登録される．そして船舶に設置されたPCがデータベースから操船指令を取得し，ECUにその操船指令を送信することで遠隔操船を行っている．

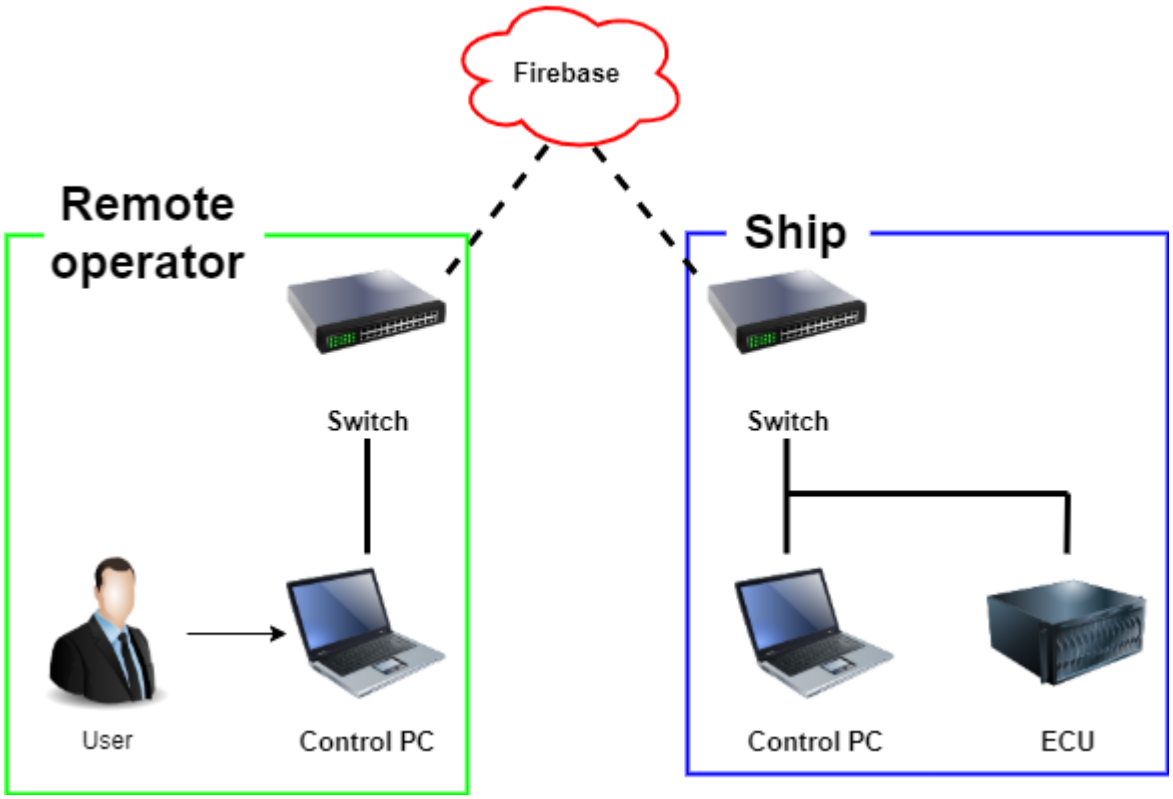


図3.1　Node-REDを利用し構築したシステム構成図

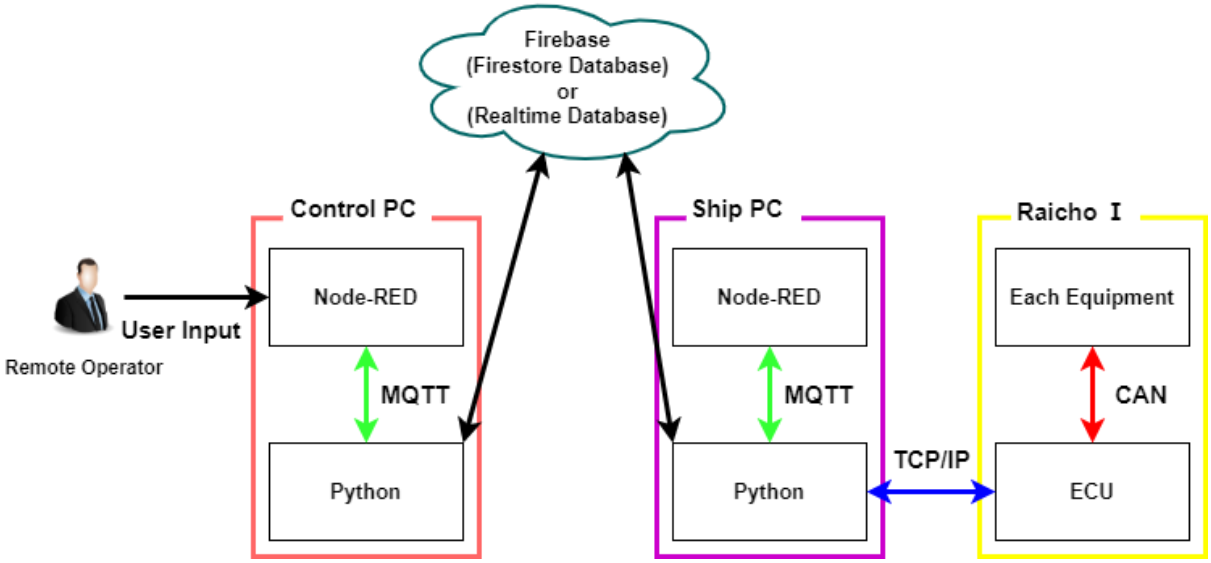


図3.2　Node-REDを利用し構築したシステムの通信

### 3.2.2 データベース

本論文で使用したデータベースは”Firebase”内の，”Firestore Database”と”Realtime Database”いずれかを使用し開発を行った．FirebaseとはGoogle社が提供しているアプリケーション開発プラットフォームである．FirebaseはBackend as a Service(BaaS)の1つである．BaaS

はバックエンド機能をアプリケーションサーバが代行するクラウドサービスである．Firebaseにはデータベースサービスの他にWebホスティングサービス，ストレージサービスなど様々なサービスが提供されている．Firebaseのサービス内のデータベースには上述したFirestore DatabaseとRealtime Databaseの２つがサービスとして提供されており，同じデータベースサービスではあるが，様々な点で違いが見られる．

Firestore Databaseはモバイル，Webサーバ開発に対応した柔軟でスケーラブルなデータベースである．NoSQLデータベースであり，Androidアプリやウェブアプリの他にPythonやC++などで使用可能である．データはドキュメントのコレクションとして保存される．Realtime Databaseと比較し，多彩で高速なクエリと高性能なスケーリングが特徴である．

Realtime DatabaseはNoSQLデータベースであり，データの保管と同期を行うことができる．データはJSONとして保存され，接続されているクライアントとリアルタイムにデータを同期する事が出来る．

２つのデータベースはそれぞれ異なる点もあり，またクラウドサービスのためかかる費用も微妙に異なっている．表3.1，3.2はFirestore DatabaseとRealtime Databaseの2022年1月時点での利用料金である．

表3.1　Firestore　料金[10]

| | 無料プラン | 従量課金制 |
|---|---|---|
| 保存済みデータ | 合計1GiB | 合計1GiBまで無料<br>その後は1GiBごとに$0.108 |
| 下り(外向き)ネットワーク | 10GiB/月 | 10GiB/月まで無料<br>その後は1GiBごとに$0.01 |
| ドキュメントの書き込み | 2万件/日 | 2万件/日まで無料<br>その後はドキュメント100000点あたり$0.115 |
| ドキュメントの読み取り | ５万件/日 | ５万件/日まで無料<br>その後はドキュメント100000点あたり$0.038 |
| ドキュメントの削除 | 2万件/日 | 削除2万件/日まで無料<br>その後はドキュメント100000点あたり$0.013 |

表3.2　Realtime Database　料金[10]

| | 無料プラン | 従量課金制 |
|---|---|---|
| 同時接続 | 100 | 20万/データベース |
| GB　保存済み | 1GB | $5/GB |
| GB　ダウンロード済み | 10GB/月 | $1/GB |
| プロジェクトごとに複数のデータベース | 無し | 有り |

各データベースには船舶の機関情報や位置情報などの航行データと，遠隔地からの操舵や出力操作などの操船指令データを保存している．航行データや指令データは船舶や遠隔地から新しいデータを取得するたびに，データベースへ現在時刻とともに追記している．

### 3.2.3 セキュリティ

PCから各データベースへの通信は暗号化されている．Pythonからのデータベースへの認証は秘密鍵を使用しており，データベースに対してすべての書き込み，読み込みなどのアクセスが可能となる．そのため秘密鍵の扱いに注意する必要がある．他人による端末の操作を防ぐために，パスワードや指紋認証，顔認証による認証を用い，PCにロックをかけることが必要であると考えられる．またPC自体が自由に持ち運べないように，ワイヤーを繋げることで盗難防止を施すことも場合によっては必要である．

### 3.2.4 実験

らいちょうⅠを使用し，今回構築したクラウドサービスを使用した遠隔監視・操船システムと，既存のリモートデスクトップを使用した遠隔操船システムとの操船指令の遅延，通信量，UI表示の遅延を2つのデータベース使用し，それぞれ計測する．

図3.3に操船指令の遅延計測を行う時のシステム構成を示す．赤い経路が計測を行う経路である．らいちょうⅠに設置されたPCは1秒間隔でECUに対して通信を行い，ECUが各機器から収集した航海データをクラウドサービス上のデータベースに保存している．遠隔地側から指令があった場合，まず遠隔地に設置されたPCからクラウド上のデータベースに指令

が保存される．らいちょうⅠに設置されたPCがそのデータベースに保存されている指令値を読み込みその指令をECUに対し送信する．

遠隔地側から指令信号を出した時間から，実際にらいちょうⅠの操舵機や推進器が操作開始されるまでの時間を計測し比較する．操舵機は指令舵角値と実際の舵角値を計測する．推進器は出力指令値と出力を計測する．2台のPCで時刻を合わせるために，操船指令や航海データがデータベースへと保存される時刻を使用している．

図3.3のシステム構成で通信量の計測を行う．船舶が航行している際に，3分間通信量の計測を行う．通信量の計測はネットワークプロトコルアナライザのWireshark (Version 3.4.6 (v3.4.6-0-g6357ac1405b8))を使用する．

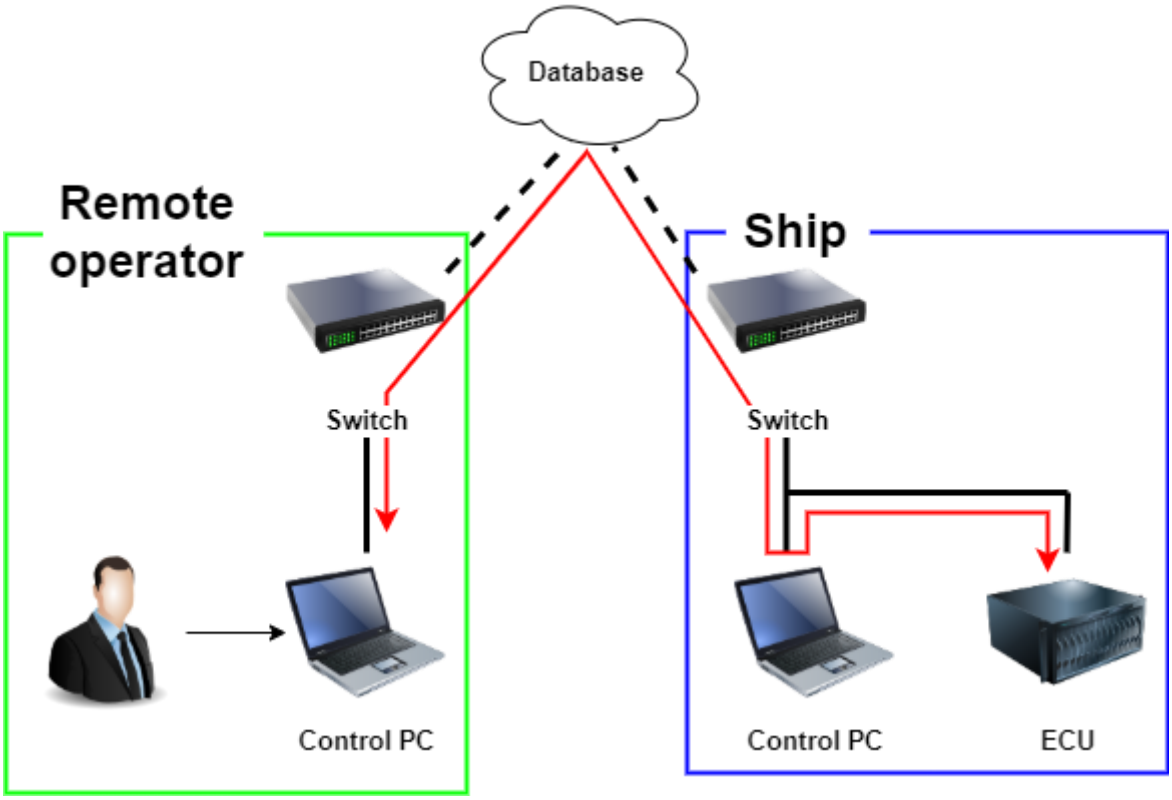


図3.3　操船指令，通信量の計測を行うシステム構成図

図3.4がCRDを使用した既存の遠隔監視・操船システムと，3.2.1で構築したNode-REDを使用したシステムとの表示の遅延計測を行う時のシステム構成である．3.2.1で構築したNode-REDを使用したシステムと，CRDを使用したシステムのそれぞれのUIを同時に1台のカメラで撮影する．そして，それぞれのUIに同じ航海データが表示されるまでのフレーム数を計測することで，UI表示遅延の計測を行う．

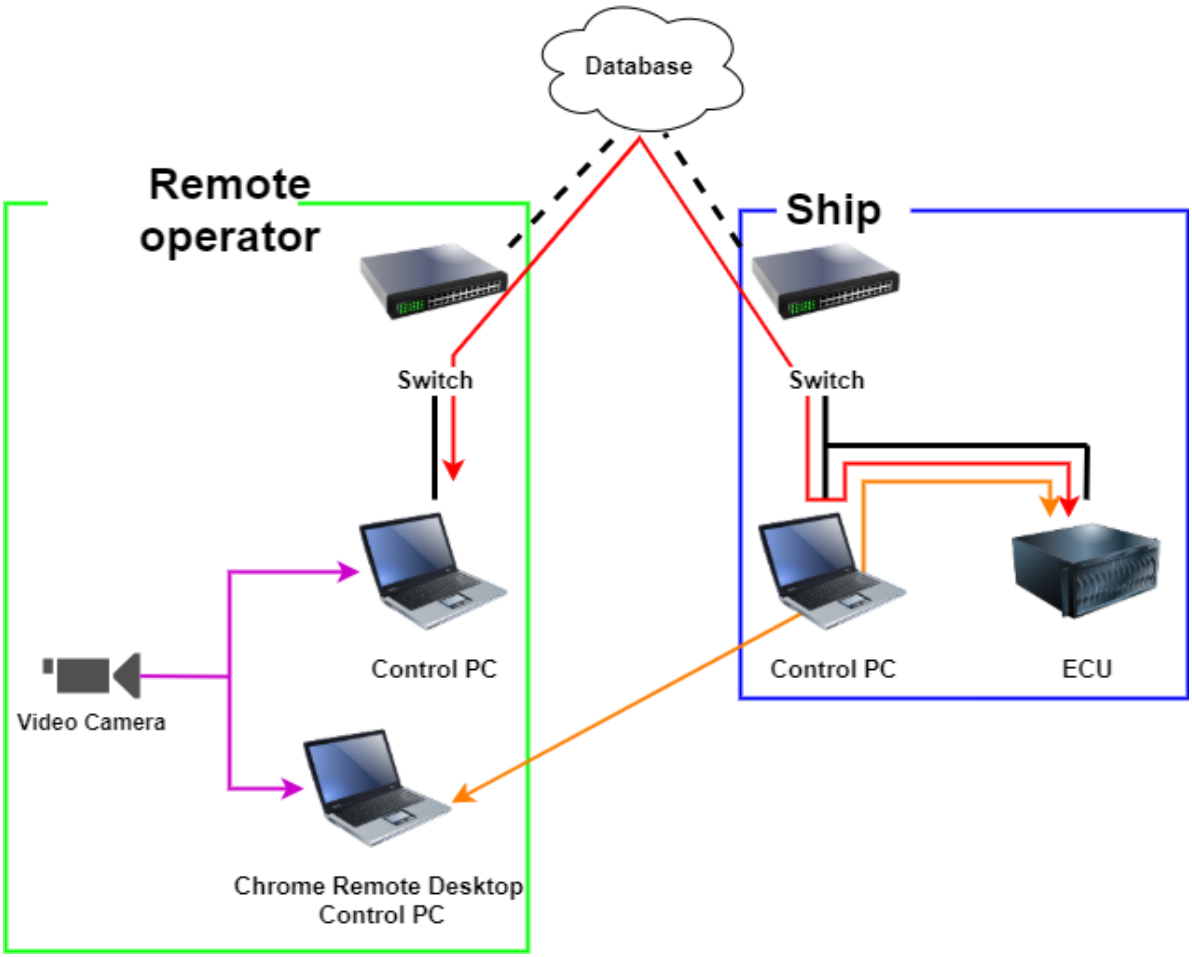


図3.4　UI表示の遅延の計測を行うシステム構成図

実験で使用したPCとカメラの仕様を表3.3，3.4，3.5に示す．

表3.3　Node-REDを利用したシステムで使用する船舶と遠隔地に設置したPCの仕様

| | ASUS ExpertBook P2451F |
|---|---|
| CPU | Intel® Core™ i5-10210U CPU @ 1.60GHz 2.11GHz |
| RAM | 8.00GB |
| ネットワークアダプター | Intel® Wireless AC 9560 160MHz<br>Intel® Ethernet Connection (10) I219-V |

表3.4　遠隔地からCRDを利用しアクセスするPCの仕様

| | HP ProBook 430 G3 |
|---|---|
| CPU | Intel® Core™ i5-6200U CPU @ 2.30GHz 2.40GHz |
| RAM | 8.00GB |
| ネットワークアダプター | Intel® Dual Band Wireless-AC 3165 |

表3.5　使用したカメラの仕様

| | |
|---|---|
| イメージセンサー | Sony Exmor RS IMX315 |
| F値 | f2.2 |
| 画素数 | 1200万画素 |
| フレームレート | 60fps |

船舶と遠隔地からデータベースへの通信は携帯電話通信網を使用し実験を行う．

### 3.2.5 結果，考察

以下図3.5から図3.8に，遠隔からの操船指令に対する操舵機，推進器の遅延計測結果を示す．ECUは1秒間隔でしか各機器のデータを計測していないため，1秒未満の計測はできない．操舵機では舵角指令値と舵角値のグラフを示す．舵角指令の場合，指令を送信した時間がx.5秒以上の場合だと次の1秒ごとの更新には変更されていないようにみえるが，その1秒後の更新時間には指令通りの舵角に変更されていることが確認できた．この遅れはECUの更新タイミングや操舵機の油圧装置，操船指令がデータベースを通じ船舶側に到達するまでの遅れによるものだと考えられる．

推進器では操船指令とモータの出力を比較している．出力指令値のマイナスはモータを逆回転させ更新の動作を行っている．モータの出力はモータの回転方向に関わらず正の値を持っている．推進器は慣性があり時定数が大きいのでスロットルレバーで操作してもこのような時間遅れは発生する．これはそのような時間遅れと同程度であるため問題ないと考える．

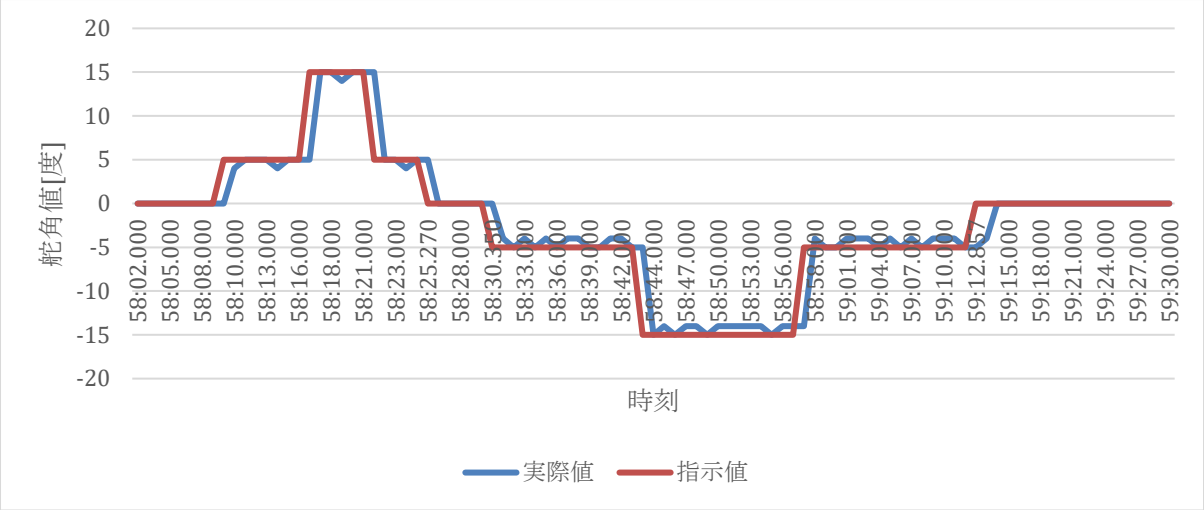


図3.5　Firestore Database使用時の舵での遠隔操船指令の遅れ

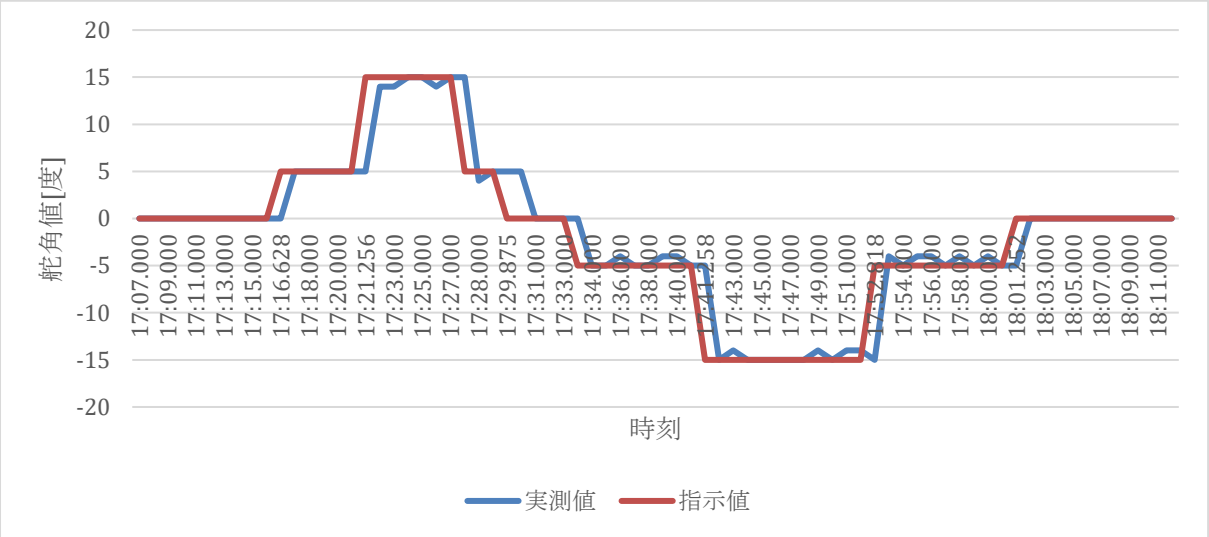


図3.6　Realtime Database使用時の舵での遠隔操船指令の遅れ

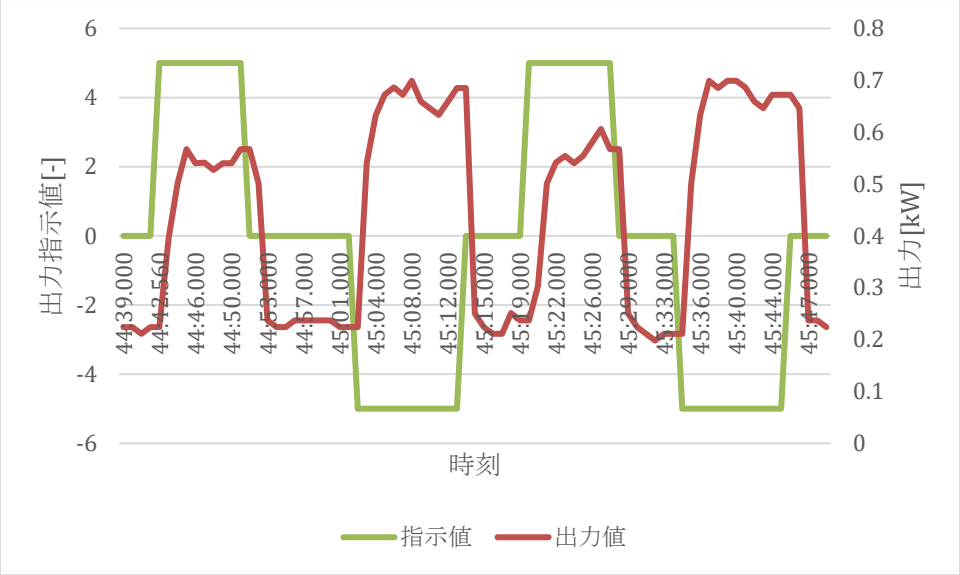


図3.7　Firestore Database使用時の推進器での遠隔操船指令の遅れ

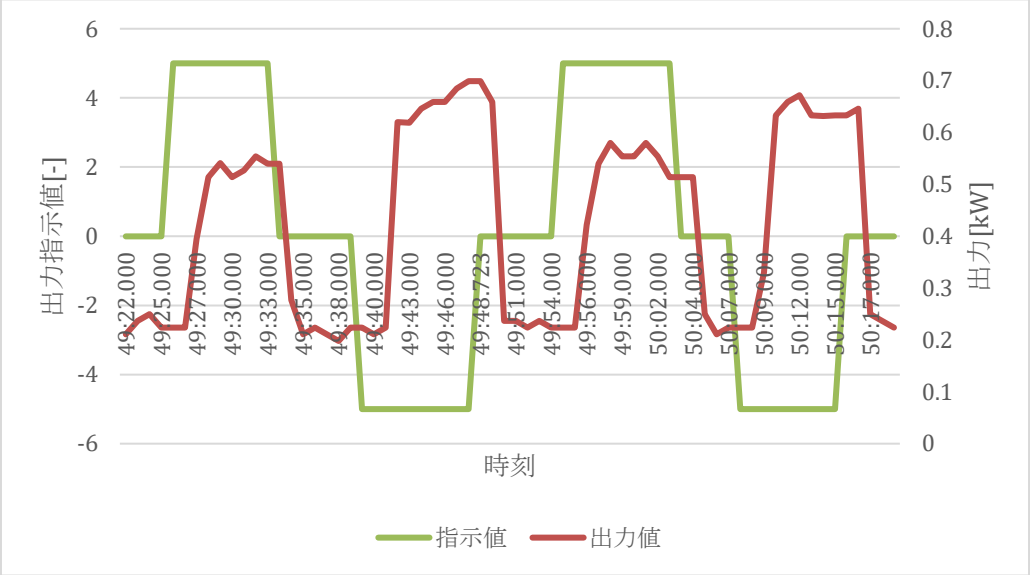


図3.8　Realtime Database使用時の推進器での遠隔操船指令の遅延

次に表示の遅延計測結果について示す．それぞれのデータベースを使用し，30回計測を行った．Firestore Database使用時では平均6.8フレーム，Realtime Database使用時では平均4.63フレームであった．実験では60fpsで動画を撮影したため，1フレームは1/60≒0.17秒である．したがって各データベースを使用した遠隔監視・操船システムとChrome Remote Desktopを使用したシステムとの遅延時間はFirestore Databaseは6.8*1/60≒0.113秒，Realtime Databaseは4.63*1/60≒0.077秒であった．ECUでの航海データ更新周期が1秒ごとに設定されており，これまでもこの1秒周期での情報で舵輪を使用し航海をおこなってきた．またChrome Remote Desktopを使用したシステムでも遠隔監視，操船を行うことが出来ていた.し

たがって本実験で計測した表示の遅れであれば遠隔監視，操船ともに問題はないと考えられる．

最後に通信量計測の結果について述べる．図3.9から図3.11までは船舶に設置されたPCで通信量を計測した結果であり，それぞれ図3.9がFirestore Databaseを使用したときの結果，図3.10がRealtime Databaseを使用したときの結果，図3.11がChrome Remote Desktopを使用したときの結果である．また表3.6が遠隔地での各通信の比較である．Chrome Remote Desktopを使用したときは平均0.864Mbpsの通信を行っており，最高約5Mbpsの通信を行っていた．それに対し今回構築したクラウドサービスを利用した遠隔監視・操船システムでは，Firestore Databaseの場合平均0.041Mbpsの通信を行っており，最高約0.18Mpbsの通信を行っていた．同じくReatime Databaseを使用した場合平均0.041Mbpsの通信を行っており，最高約0.1Mpbsの通信を行っていた．Chrome Remote Desktopを使用したものと比べ，Firestore Database，Realtime Databaseのどちらにおいても通信量を削減することが出来ている．これはChrome Remote Desktopを使用したシステムは，データだけでなくウィンドウに表示されたユーザーインターフェースを送信しており，それに対して今回構築したシステムはデータベースを通じ航海データだけ送信しているため，このような通信量の差が発生していると考えられる．

図3.12から図3.14までは遠隔地に設置されたPCで通信量を計測した結果であり，それぞれ図3.12がFirestore Databaseを使用したときの結果，図3.13がRealtime Databaseを使用したときの結果，図3.14がChrome Remote Desktopを使用したときの結果である．また表3.7が遠隔地での各通信の比較である．Chrome Remote Desktopを使用した場合は平均約4.11Mbps，Firestore Database使用時は平均約0.024Mbps，Realtime Database使用時は平均約0.022Mbpsであった．遠隔地においてもどちらのデータベースを使用した場合でもChrome Remote Desktopを使用した場合と比べ通信量が小さくなっていることが確認できた．

使用する海域や通信の混雑状態によっても変動すると考えられるが，今回実験を行った範囲ではクラウドサービスを利用し構築したシステムは，Chrome Remote Desktopを使用したシステムと比べ通信量が大幅に抑えられたことが分かった．

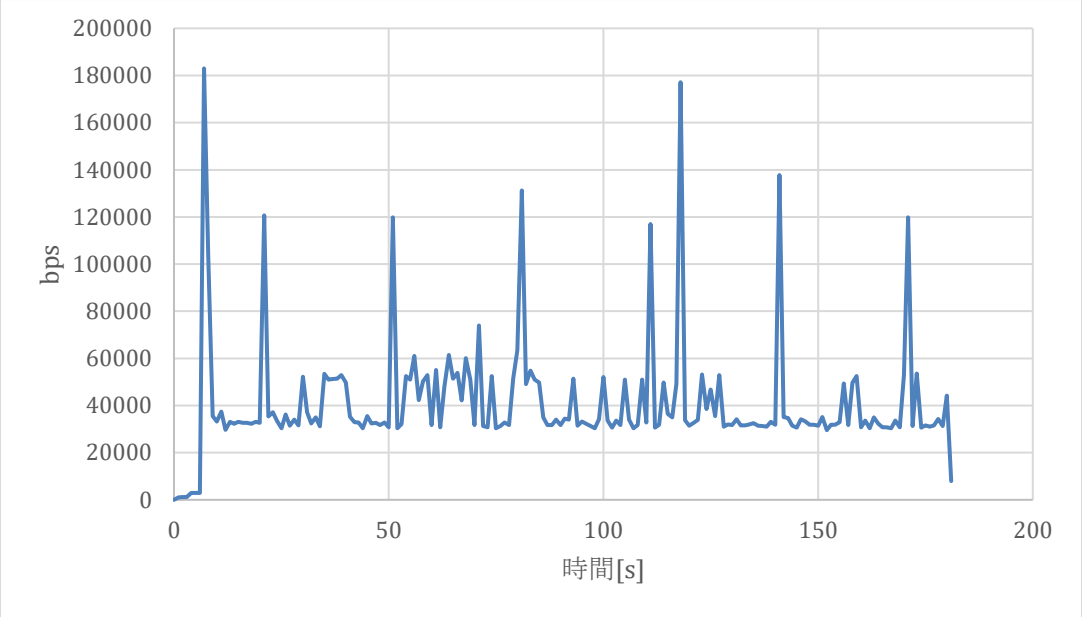


図3.9　Firestore Database使用時の船舶側での通信量

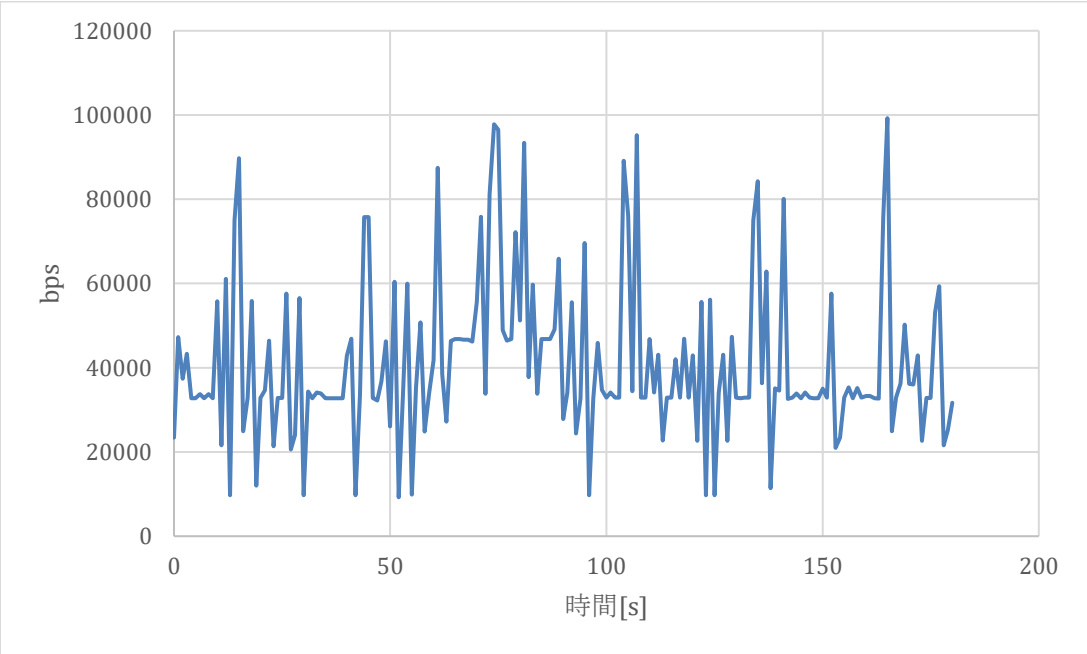


図3.10　Realtime Database使用時の船舶側での通信量

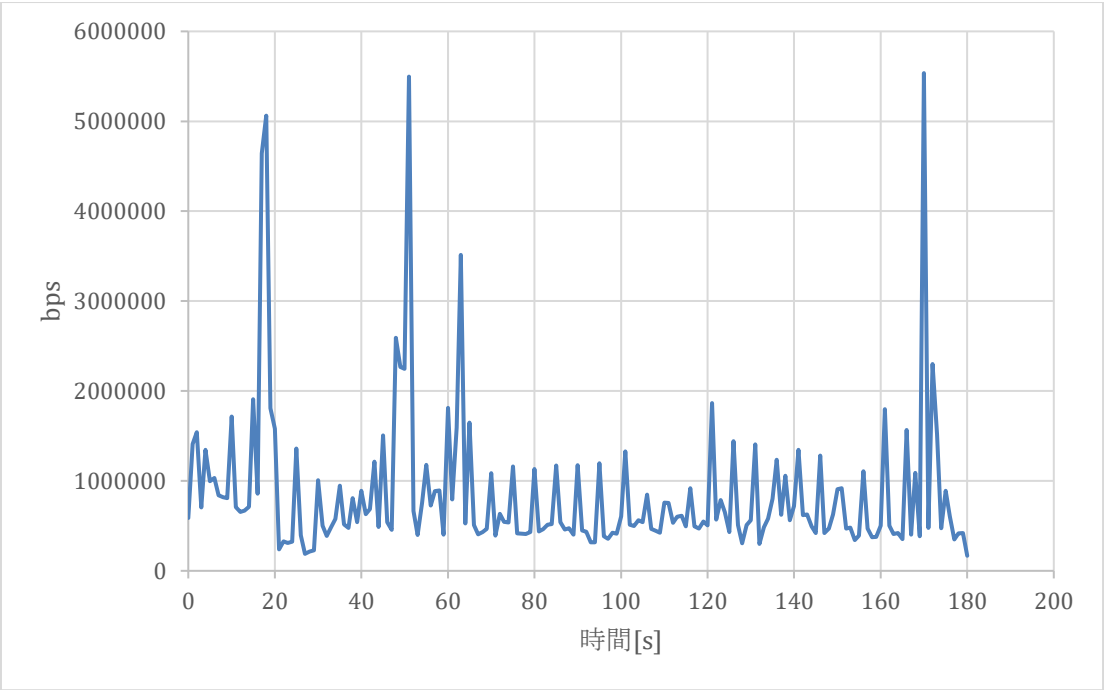


図3.11　Chrome Remote Desktop使用時の船舶側での通信量

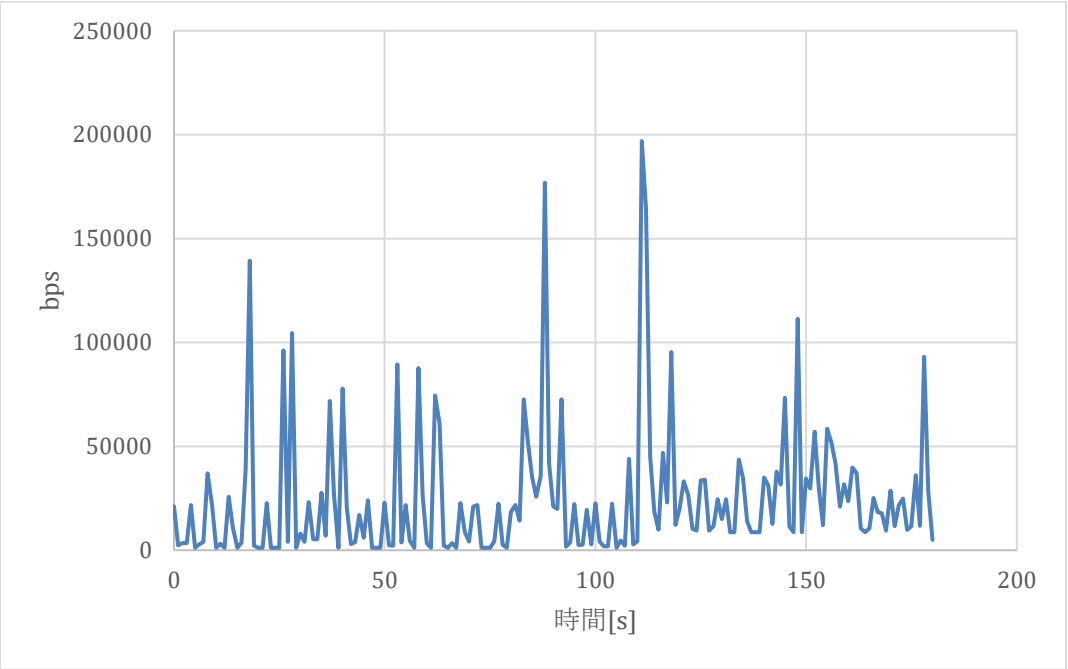


図3.12　Firestore Database使用時の遠隔地側での通信量

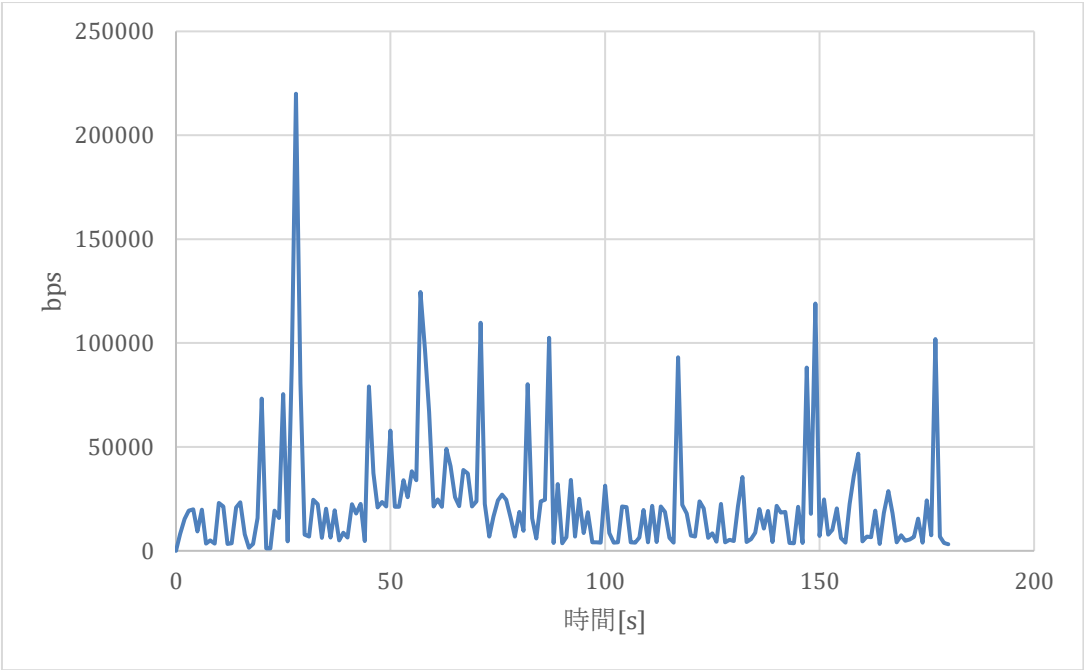


図3.13　Realtime Database使用時の遠隔地側での通信量

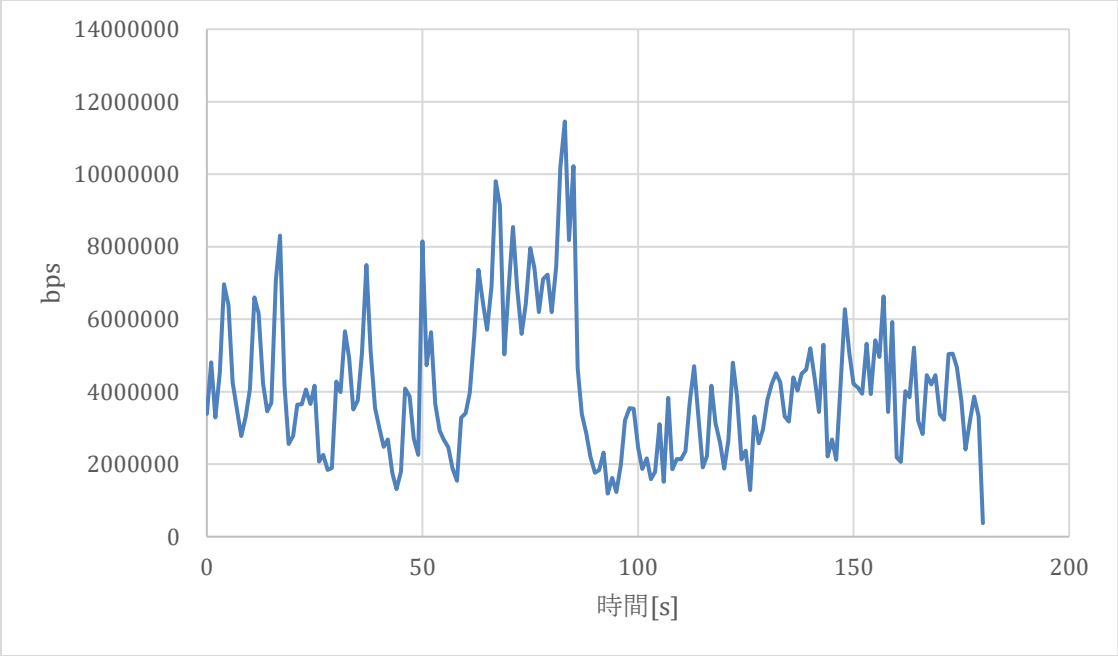


図3.14　Chrome Remote Desktop使用時の遠隔地側での通信量

表3.6　船舶側での通信量

| | Chrome Remote Desktop | Firestore Database | Realtime Database |
|---|---|---|---|
| パケット数 | 30034 | 4185 | 3390 |
| 平均パケット毎秒 | 166.9 | 23.3 | 18.9 |
| 平均パケットサイズ，バイト | 647 | 223 | 275 |
| バイト数 | 19443983 | 934116 | 933440 |
| 平均バイト数毎秒 | 108022 | 5199 | 5194 |
| 平均ビット数毎秒 | 864176 | 41592 | 41552 |

表3.7　遠隔地側での通信量

| | Chrome Remote Desktop | Firestore Database | Realtime Database |
|---|---|---|---|
| パケット数 | 85472 | 2150 | 2216 |
| 平均パケット毎秒 | 474.8 | 12 | 12.3 |
| 平均パケットサイズ，バイト | 1084 | 258 | 232 |
| バイト数 | 92635587 | 553722 | 513950 |
| 平均バイト数毎秒 | 514642 | 3081 | 2863 |
| 平均ビット数毎秒 | 4117136 | 24648 | 22904 |

## 3.3 Webページを利用した遠隔監視・操船システム

### 3.3.1 システム構成

3.2節では，リモートオペレーターが自由にユーザーインターフェースを変更可能なようにPythonとNode-REDを使用したシステムを構築した．しかしNode-REDを動作させるためにはPCを使う必要があり使用できるデバイスに限りがある．Chrome Remote Desktopなどリモートデスクトップを使用したアプリケーションを使用する事で，様々なデバイスで遠隔監視，操船を行うことができるが煩雑である．そのためWebブラウザを使用した遠隔監視・操船システムの開発を行った．

図3.15が今回構築したシステム構成図である．船舶にはPCを設置し，ECUと通信することで航行データを取得し，データベースに航行データの保存を行っている．データベースに保存を行う航行データの取得種類は3.2のシステムと変更せず，また航海データの更新頻度は1秒である．また遠隔地から送信された操船指令を受け，ECUに送信することで制御を行っている．ECUとEthernetを使用した通信を行うため，船舶内にPCを設置している．遠隔地からは，図3.16のWebページにアクセスすることで遠隔監視，操船を行う．このWebページはデータベースと通信を行っており，航行データを表示している．またボタンを押すことで，操船指令を船舶に送信することが可能である． このWebページはHTMLとjavascriptを使用しており，リロードせずにUIの航行データの変更などの動作を実現している．Webページのアクセスに Webブラウザを使用するため，デバイスやOSなどに依存せず遠隔監視，操船を行うことが可能である．

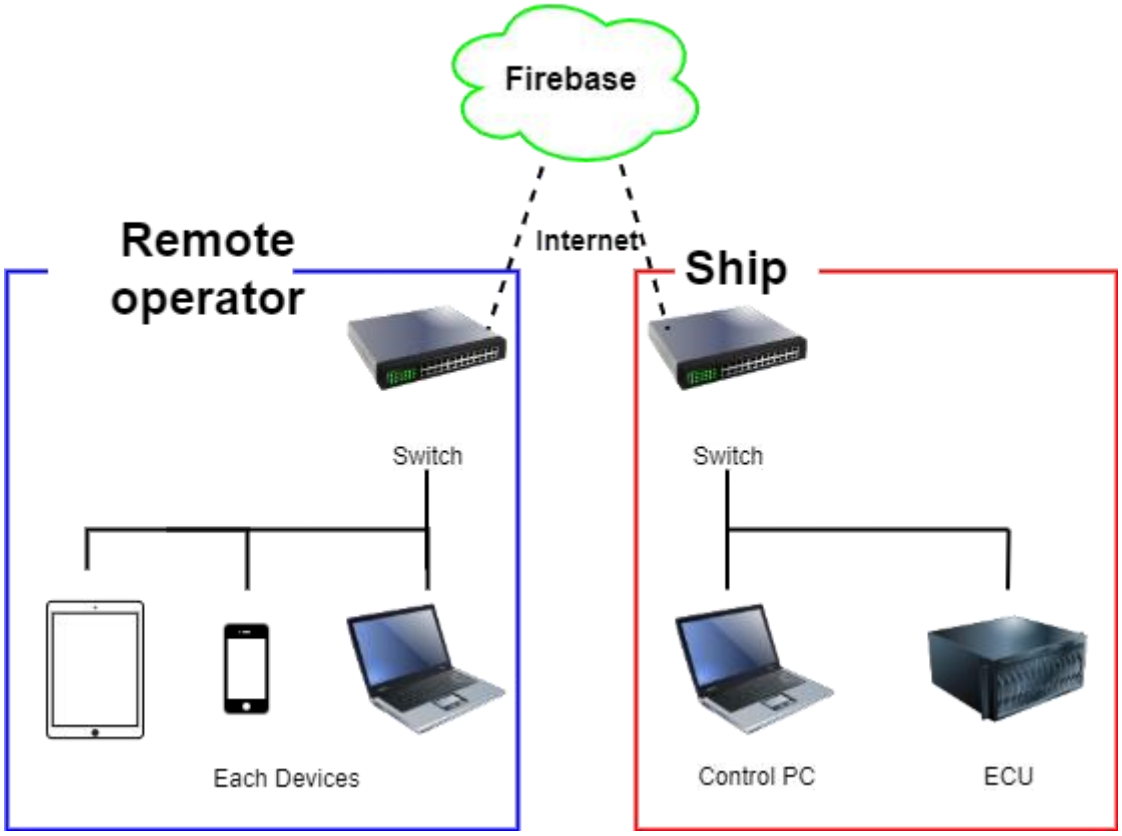


図3.15　システム構成図

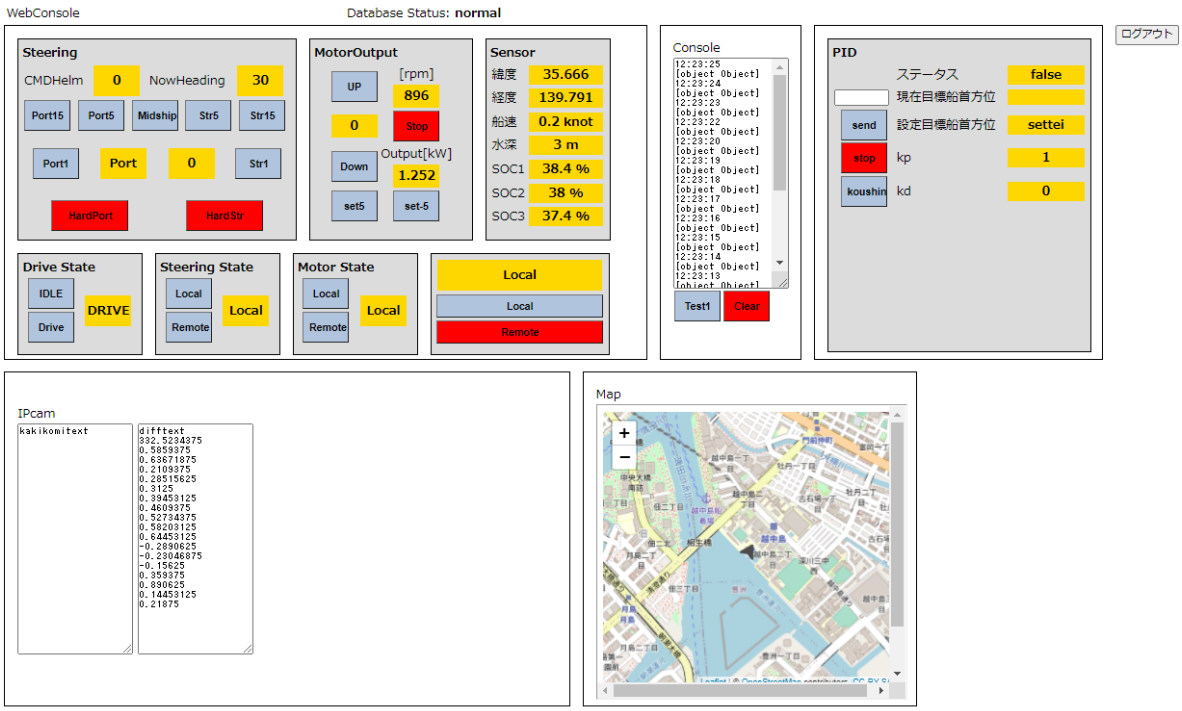


図3.16　ユーザーインターフェース

### 3.3.2 hostingサービス

Webページを作成しただけでは様々なところで見ることが出来ない．いろいろなデバイスで自由に見るためにはWebページを置き，アクセスすることができるサーバが必要である．そのため3.2節でも使用したクラウドサービスであるFirebaseを使用し，その中のhostingサービスである図17の”Firebase Hosting”を利用しWebページを設置した．このhostingサービスを使用することで，デバイスがインターネットに接続されている場合，図3.16のWebページにアクセスすることが可能である．

図3.17　Firebase Hosting

### 3.3.3 セキュリティ

上述したhostingサービスを利用することで，図18のようにHTTPSでの通信を行い，Webページにアクセスすることが可能となる．HTTPSとはHyper Text Transfer Protocolの略でありホームページを見るときのプロトコルの1つである．Webページのプロトコルには HTTPとHTTPSの 2 つがあるが，HTTPSの場合は暗号化されて封書のように中身が受け手以外には分からないようになり，通信が見知らぬ誰かに盗聴されず，安全である．

また図3.19のようにメールアドレスとパスワードによるログイン画面を実装している．これはFirebase内のFirebase Authenticationを使用し図3.20のようなログイン画面を構築している．ログインにはメールアドレスだけでなく外部のGoogleやFacebookなどのidプロバイダーによる認証が可能となっている．この機能を施すことでよりセキュリティを高めている．

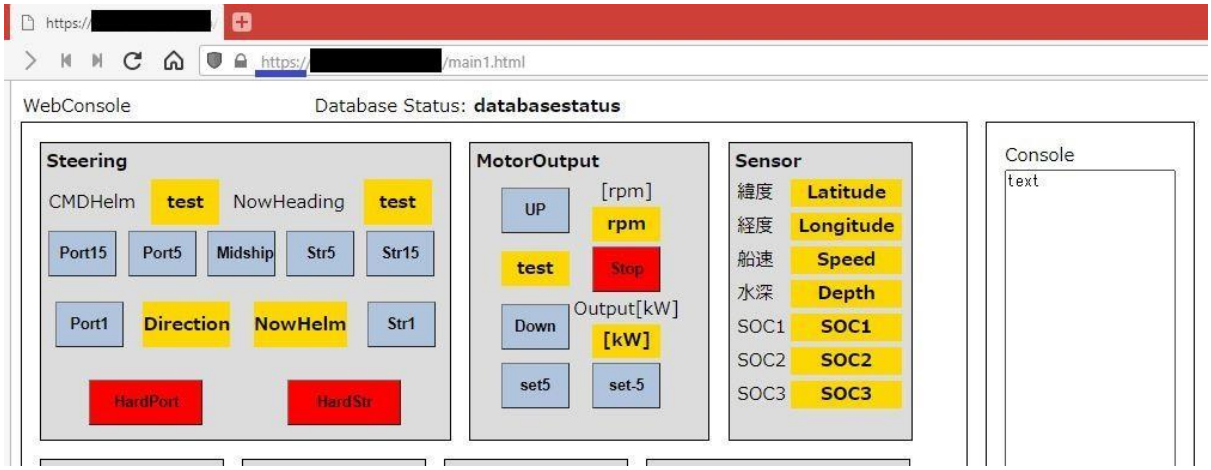


図3.18　Webページのプロトコル

図3.19　Firebase Authentication

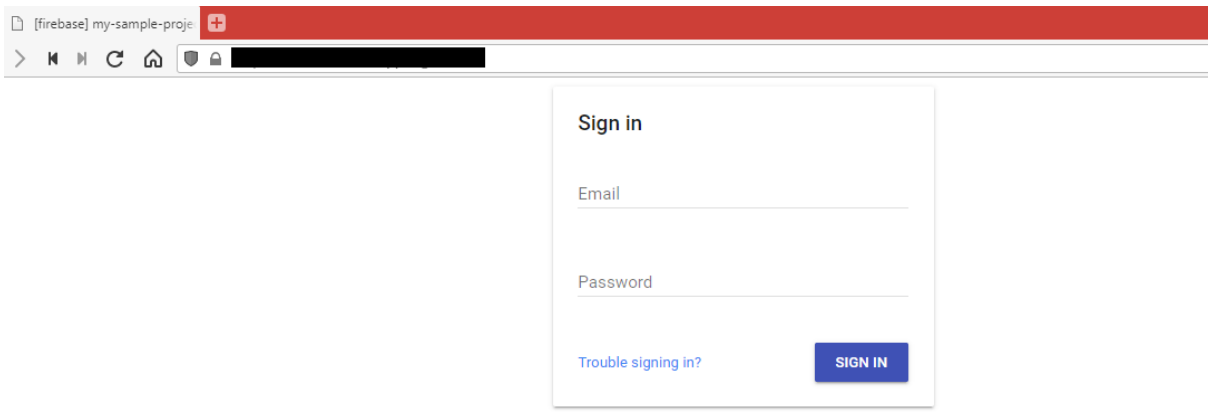


図3.20　ログイン画面

またデータベース側にもセキュリティをかけている．データベースに書き込まれるデータに対しても配列や文字列などのデータの型やそのデータ自身の大きさに制限をかけることで

不適切なデータが書き込まれることを防いでいる．今後，ユーザーによってどのデータに書き込めることが可能で，読み込めることが可能なのかの制限も必要と考える．

### 3.3.4 実験

らいちょうＩを使用し3.3.1で構築したWebページを使用した遠隔監視・操船システムでの操船指令信号の遅延，通信量，UI表示の遅延をそれぞれ計測する．Webページを使用した遠隔監視・操船システムはPCに限らず様々なデバイスで使用できることが利点ではあるが，実験時では実験データを計測するためPCを使用し計測を行った．

図3.3に操船指令の遅延計測を行う時のシステム構成を示す．赤い経路が計測を行う経路である．らいちょうＩに設置されたPCは1秒間隔でECUに対して通信を行い，ECUが各機器から収集した航海データをクラウドサービス上のデータベースに保存している．遠隔地側から指令があった場合，まず遠隔地に設置されたPCからクラウド上のデータベースに指令が保存される．らいちょうＩに設置されたPCがそのデータベースに保存されている指令値を読み込みその指令をECUに対し送信する．

遠隔地側から指令信号を出した時間から，実際にらいちょうＩの操舵機や推進器が操作開始されるまでの時間を計測し比較する．操舵機は指令舵角値と実際の舵角値を計測する，推進器は出力指令値と出力を計測する．2台のPCで時刻を合わせるために，操船指令や航海データがデータベースへと保存される時刻を使用している．

図3.3のシステム構成で通信量の計測を行う．船舶が航行している時，3分間通信量の計測を行う．通信量の計測はネットワークプロトコルアナライザのWireshark (Version 3.4.6 (v3.4.6-0-g6357ac1405b8))を使用する．

図3.4がCRDを使用した既存の遠隔監視・操船システムと，3.2.1で構築したNode-REDを使用したシステムとの表示の遅延計測を行う時のシステム構成である．3.2.1で構築したNode-REDを使用したシステムと，CRDを使用したシステムのそれぞれのUIを同時に1台のカメラで撮影する．そして，それぞれのUIに同じ航海データが表示されるまでのフレーム数を計測することで，UI表示遅延の計測を行う．

実験で使用したPCとカメラの仕様は表3.3，3.4，3.5である．また船舶と遠隔地からデータベースへの通信は，携帯電話通信網を使用し実験を行う．

### 3.3.5 結果，考察

まず遠隔操船指令に対する舵，推進器動作の遅延について述べる．図3.21，図3.22は操船指令に対して実際に舵や出力の動作との表したものである．舵の指令については船舶の制御

周期の関係で1秒ごとの値にはなるが，1秒以内には舵は変わっており遅延もあまり見られない．そのためWebページを使用したシステムであっても問題がないと考えられる．出力指令の遅延についても指令値に対して出力値がずれているが，推進器は慣性があり時定数が大きいので，スロットルレバーで操作してもこのような時間遅れは発生する．これはそのような時間遅れと同程度であるため問題ない．したがって操舵機への指令や推進器への指令どちらに対しても遅れは多少あるが，実際に操船した時の感覚も合わせて問題は無いと判断する．

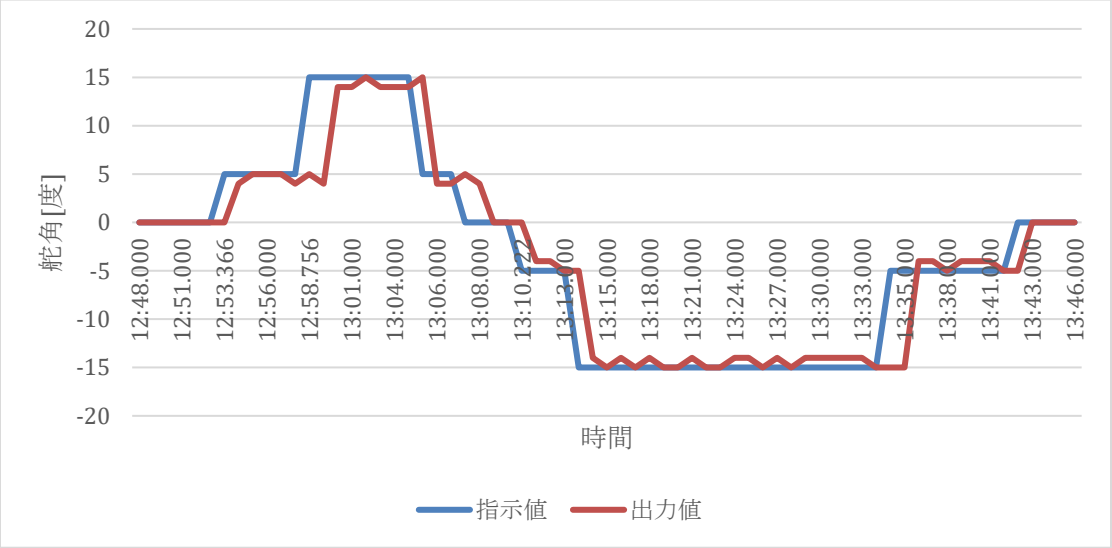


図3.21　Webページ使用時の舵での遠隔操船指令の遅れ

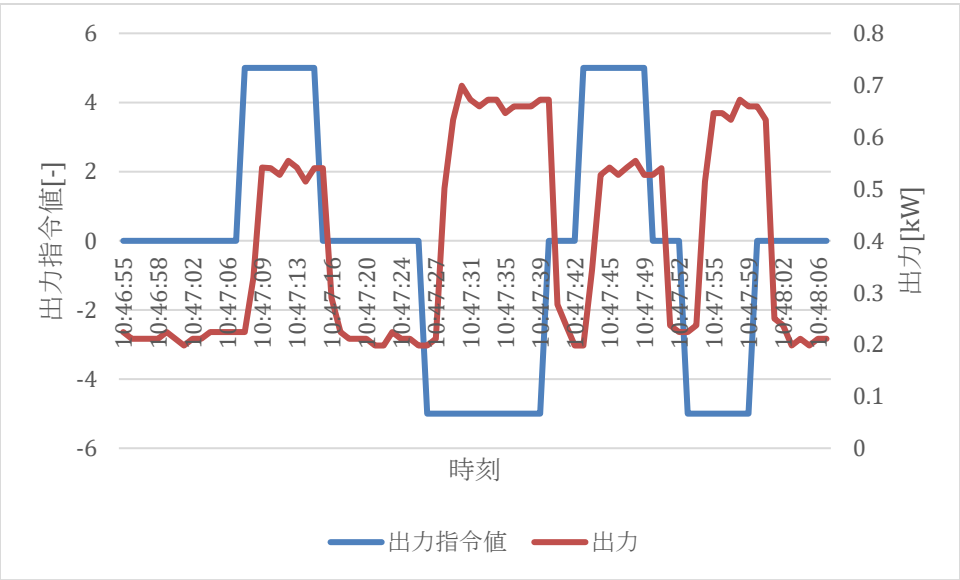


図3.22　Webページ使用時の出力での遠隔操船指令の遅れ

次にWebページを使用した遠隔操船・監視システムと，CRDを使用したシステムとの表示の遅延について述べる．30回計測を行った結果，Webページを使用したシステムはCRDを使用したシステム平均4.5フレームの遅延であった．実験では動画を60fpsで撮影したため，上述の表示の遅延は4.5*1/60≒0.075秒である．ECUでの航海データ更新周期が1秒ごとに設定されており，これまでもこの1秒周期での情報で舵輪を使用し航海をおこなってきた．またChrome Remote Desktopを使用したシステムでも遠隔監視，操船を行うことが出来ていた.したがってWebページを使用したシステムであっても問題はないと考えられる．

最後に通信量計測の結果について述べる．

図3.23がWebページを使用したシステム使用時に，船舶内に設置されたPCで通信量を3分間計測したものであり，表3.8が船舶内に設置されたPCで計測したCRDを使用したシステムとWebページを使用したシステムの比較である．CRDを使用した遠隔監視・操船システムは平均0.864Mbpsの通信であったのに対し，Webページを使用したシステムは平均0.031Mbpsの通信であった．また総通信量はCRDを使用したシステムと比較し，今回構築したWebページを使用した遠隔監視・操船システムは約97%削減された．

図3.24が遠隔地で遠隔監視，操船を行っているPCで3分間計測したものである．また表3.9が遠隔地で遠隔監視，操船を行っているPCで計測した結果の比較である．CRDを使用した遠隔監視・操船システムは平均4.117Mbpsの通信であったのに対し，Webページを使用したシステムは平均0.072Mbpsの通信であった．また総通信量はCRDを使用したシステムと比較し，今回構築したWebページを使用した遠隔監視・操船システムは約98%削減された．

したがって船舶，遠隔地の両方でCRDを使用したシステムと比較し，Webページを使用したシステムでの通信は削減された．Webページを利用したシステムはデータベースを通して航海データだけを通信しているため，UIを含めて通信を行っているCRDを使用したシステムと比べ，通信が削減されていると考える．

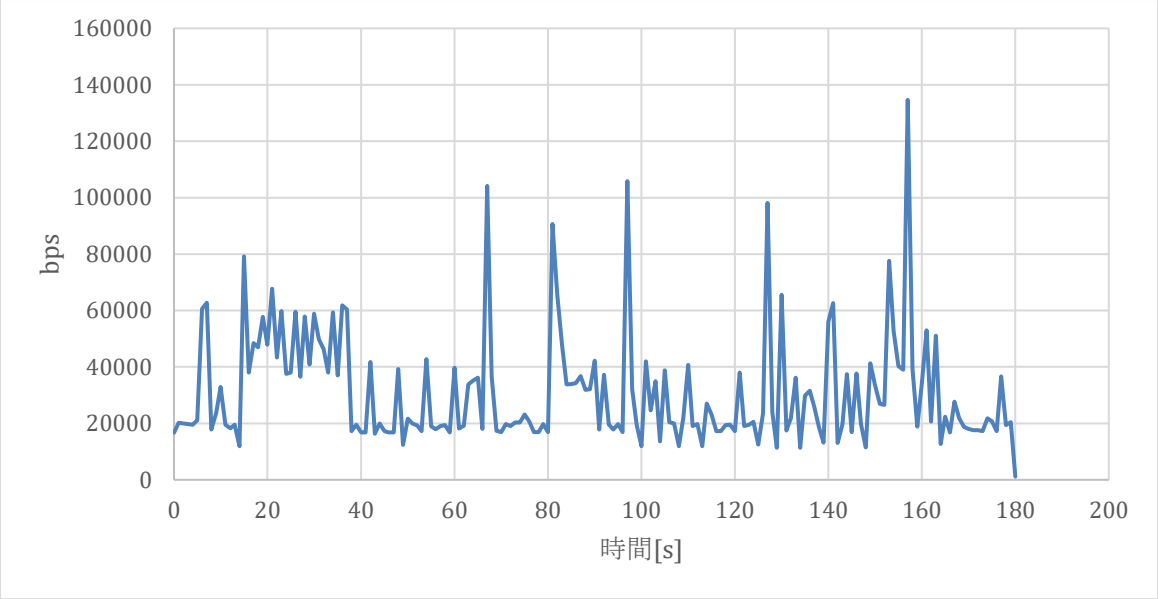


図3.23　Webページ使用時の船舶側での通信量

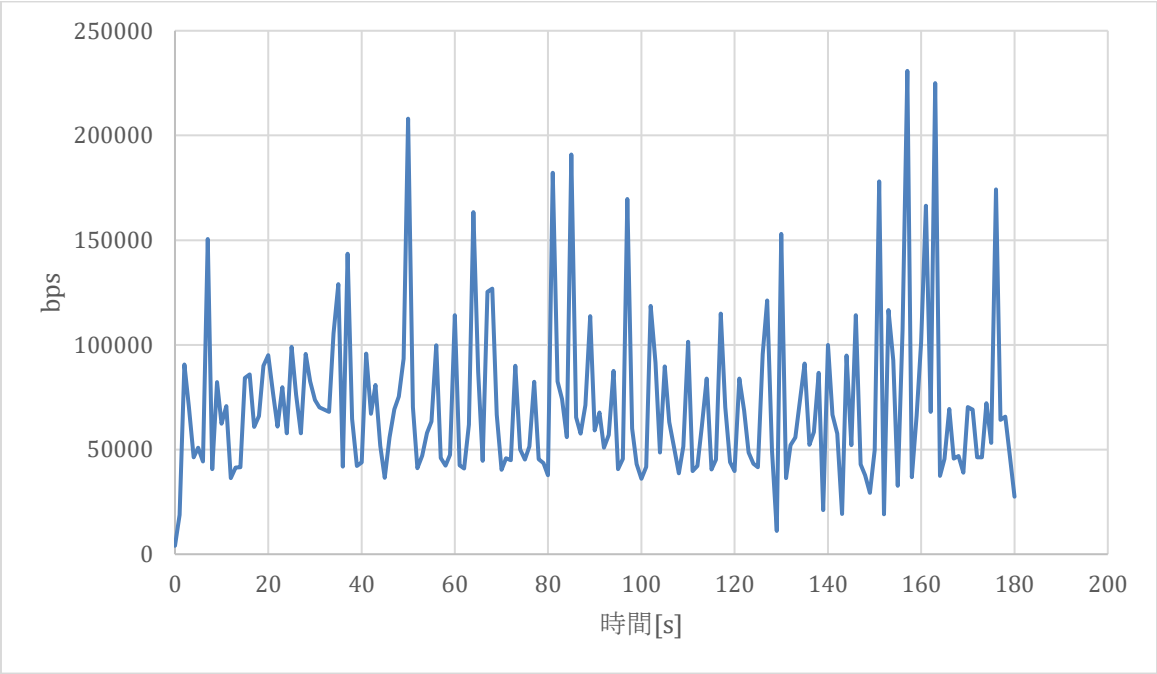


図3.24　Webページ使用時の遠隔地側での通信量

表3.8　船舶側での通信量

| | Chrome Remote Desktop | Firestore Database + Webページ |
|---|---|---|
| パケット数 | 30034 | 3435 |
| 平均パケット毎秒 | 166.9 | 19.1 |
| 平均パケットサイズ，バイト | 647 | 205 |
| バイト数 | 19443983 | 703232 |
| 平均バイト数毎秒 | 108022 | 3915 |
| 平均ビット数毎秒 | 864176 | 31320 |

表3.9　遠隔地側での通信量

| | Chrome Remote Desktop | Firestore Database + Webページ |
|---|---|---|
| パケット数 | 85472 | 8765 |
| 平均パケット毎秒 | 474.8 | 48.8 |
| 平均パケットサイズ，バイト | 1084 | 184 |
| バイト数 | 92635587 | 1615616 |
| 平均バイト数毎秒 | 514642 | 8944 |
| 平均ビット数毎秒 | 4117136 | 71552 |

## 3.4 安全装置

### 3.4.1 船側での安全装置

いかなる機械装置にも安全装置があり，不測の事態に備えている．本研究で構築したシステムにおいても安全装置を組み込んだ．船舶側に設置しているPCにはpingツールを用い，使用しているクラウドサービスのデータベースにパケットが到達しているのかどうかの監視を行っている．遠隔操船を行っている場合，電波が途切れた場合，正常に航行が出来ない．そのため，通信状況が悪い場合を考慮した動作を入れる必要がある．通信が正常に行えていることが可能かどうかの判断は，まず1秒ごとにpingを使用し，データベースへ到達可能だった場合はある変数を 0 にし，到達出来なかった場合は変数に 1 ずつ足していく．そしてその変数が5になったときに，遠隔操船から下記の動作への変更を行っている．今回構築したシステムでは変数の値を5に設定している．この 5 という値は通信環境や航行する海域によって変更する必要があると考える．

またその後の動作については様々な方法が考えられる．現在はある一定の出力と舵を保持しながら進むようにしている．これは，船舶は車と異なり停止した場合でも海流や風の影響を受け流されてしまう．そのため出力と舵をある程度与えることで他船からも進行方向が分かりやすく，船をよける操船を行えることを目標としている．これは航行する海域によって検討する必要があり，他の船舶やブイなどの状況から判断する事で電波があるところまで戻る方法などが考えられる．

### 3.4.2 遠隔地側での安全装置

本研究で構築した遠隔監視・操船システムでは，船舶側から各データベースに書き込んだ際の時間を航海データや機関データなどと合わせて保存している．構築したシステムでは現在1秒ごとにデータベースへ航海データの保存を行っている．船舶では陸上と違い電波状況が悪い場合や，プログラムが動作しているPCが何らかの形でエラーが発生することでデータベースに航海データが書き込むことが出来ない可能性が考えられる．遠隔地側ではデータベースから航海データを受け取る時，Firebaseの機能の1つのリアルタイムアップデートを使用している．この機能はデータベースの値が変更するたびにスナップショットが作成され，データベースが変更されるたびそのスナップショットが更新されるというものである．この機能を使用することで，余計な通信を行わずに通信量を削減することができ，また瞬時に遠隔地で最新の航海データを取得することが可能となる．しかしこの機能を使用している時，上述のように船舶側で何か異常が起こった場合にはデータベースが更新されないため，

古い航海データがユーザーインターフェースに表示され続けてしまう．そのため船舶側から各データベースに書き込んだ時の時間を航海データや機関データなどと合わせて保存し，遠隔地側で現在の時刻との差を計測している．今回構築したシステムではこの差が3秒以上になると，図3.25から図3.28のようにヒューマンインタフェースにユーザーインターフェースに表示されている舵角や出力値などが，異常であることを表示している．異常があった場合には，他の船舶からの情報や航路に設置されたカメラなどにアクセスすることで，遠隔監視を続けることが想定される．

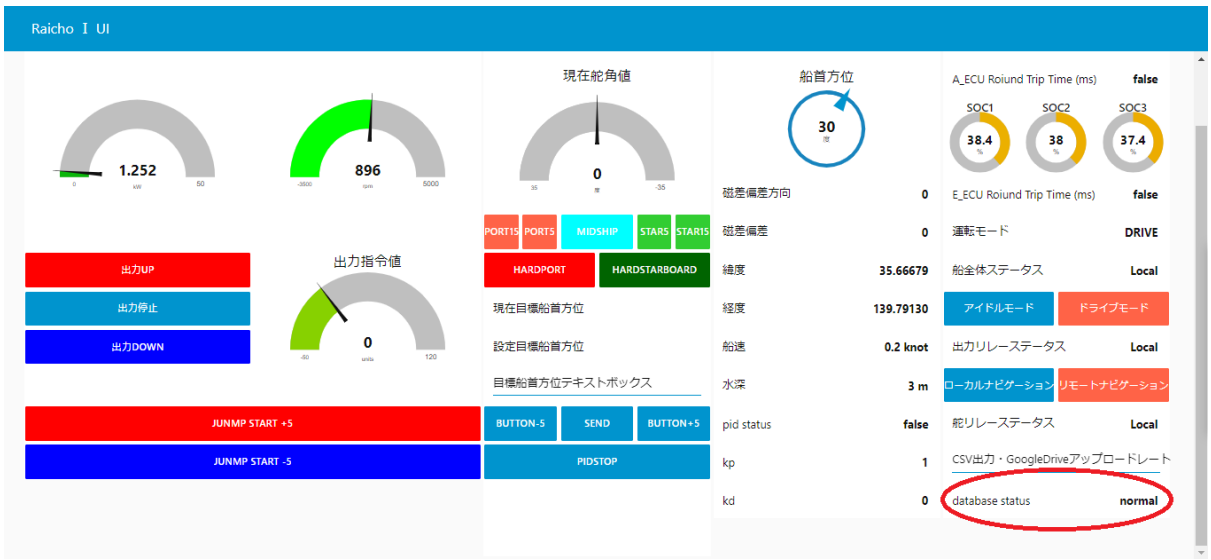


図3.25　データベースの値が正しいと判断した場合

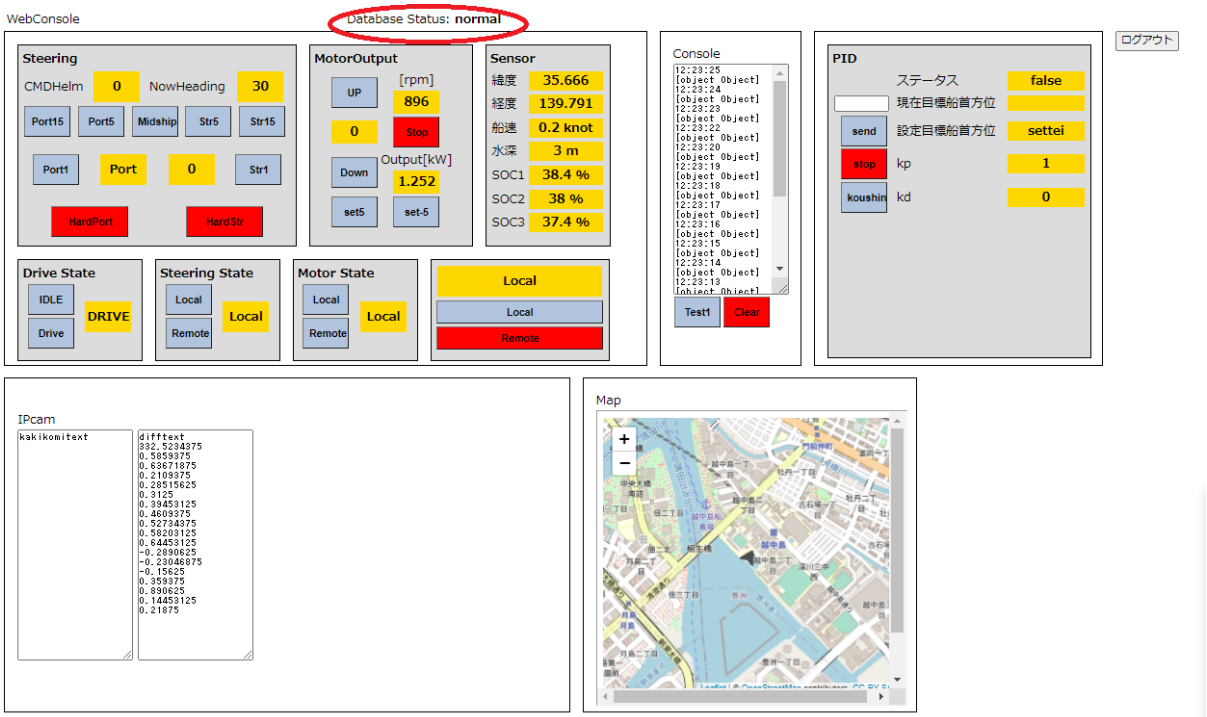


図3.26　データベースの値が正しいと判断した場合

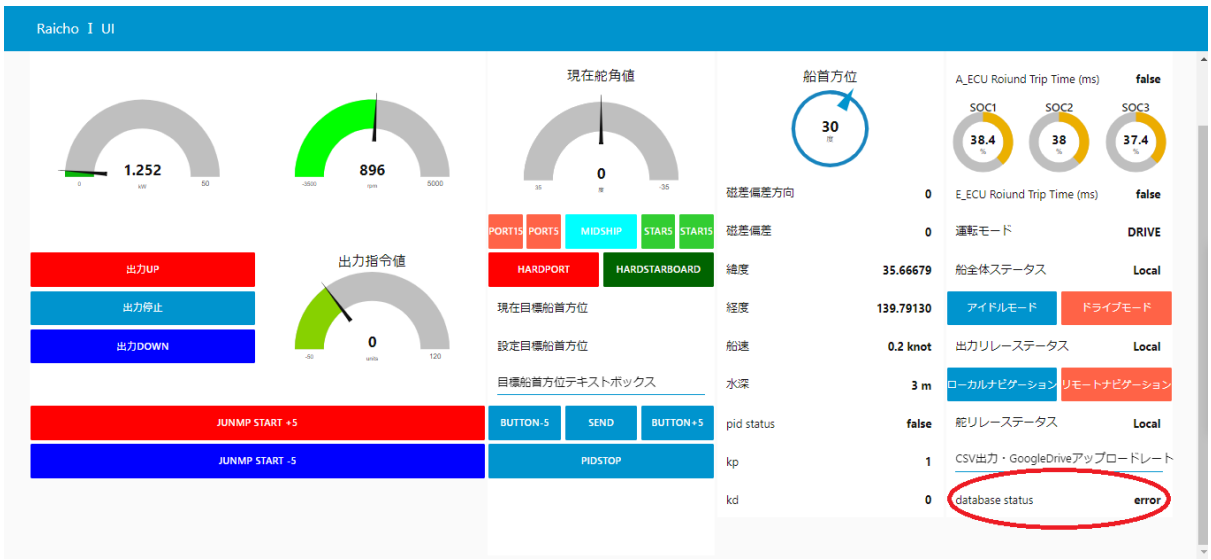


図3.27　データベースの値が正しくないと判断した場合

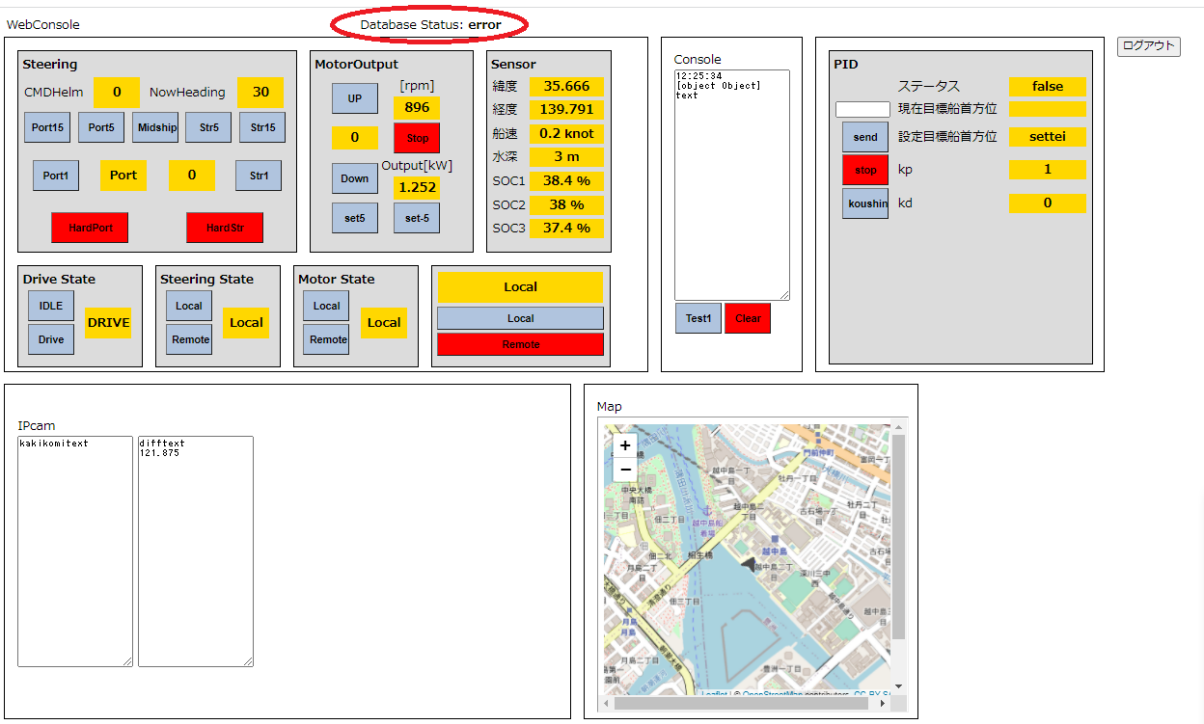


図3.28　データベースの値が正しくないと判断した場合

# 第4章 オートパイロット機能の開発

## 4.1 概要

3章では自動運航船のシステム構成の1つである遠隔監視，遠隔操船についての研究を行った．遠隔操船を行う場合，通信が途切れた場合や通信環境が悪く通信が途切れた場合，航行不能に陥ってしまう．またカメラを使用する場合や通信環境が整ってない場合，通信帯域を圧迫し通信が不安定になり，同じく航行不能に陥る可能性が出てくる．そのため本章では，本研究で構築したクラウドサービスを利用したシステムに対応したオートパイロット機能の開発を行った．オートパイロットとはジャイロコンパスなどの方位データをもとに，目標の針路を保持するように操舵を制御する機能である．開発を行ったオートパイロット機能は，遠隔地から舵角を逐次船舶に与え制御を行うのではなく，船舶に対して目標船首方位だけを与え，船舶内のローカルPCで制御を行う．エッジコンピュータである船舶に設置されているPCで制御を行うことで，遠隔地と船舶間での通信不可の低減を行えるほか，逐次の通信によるデータ漏洩の懸念，通信の遅延による制御の遅れを防ぐことが可能である．

## 4.2 制御方法

本研究で使用した制御はPID制御を用いた．PID制御はフィードバック制御の1つであり長い歴史を持っているが扱いやすさから産業界では現在でも使用されている．PIDのパラメーターKp，Kdを調節しそれぞれ5，0.5とした．舵が油圧制御装置であるため舵を切りすぎ戻る頃には制御が遅れてしまっていた．また使用する船舶で人間が直進を行う操船する場合，左右約5度の操舵を行うことで針路を保持できていた．これらのことを踏まえ，左右5度以上に操舵を行わないような制限を設けた．

図4.1に本研究で構築したオートパイロットで使用した制御のブロック線図を示す．舵角を与える式は舵角を$\theta(t)$，目標船首方位と現在船首方位の偏差を$e(t)$とおくと

$$\theta(t) = Kp * e(t) + Kd * \frac{de(t)}{dt} + \theta(t-1)$$

となる．しかし式は連続系であるためコンピュータで扱う時に時間がかかってしまう．そのため上述の式を差分方程式に直す離散化を行った．

$$\theta(t) = Kp * e(t) + Kd * (e(t) - e(t-1)) + \theta(t-1)$$

離散化を行った結果が式である．微分の離散化には後退差分を用いた．この式を用い制御を行う．

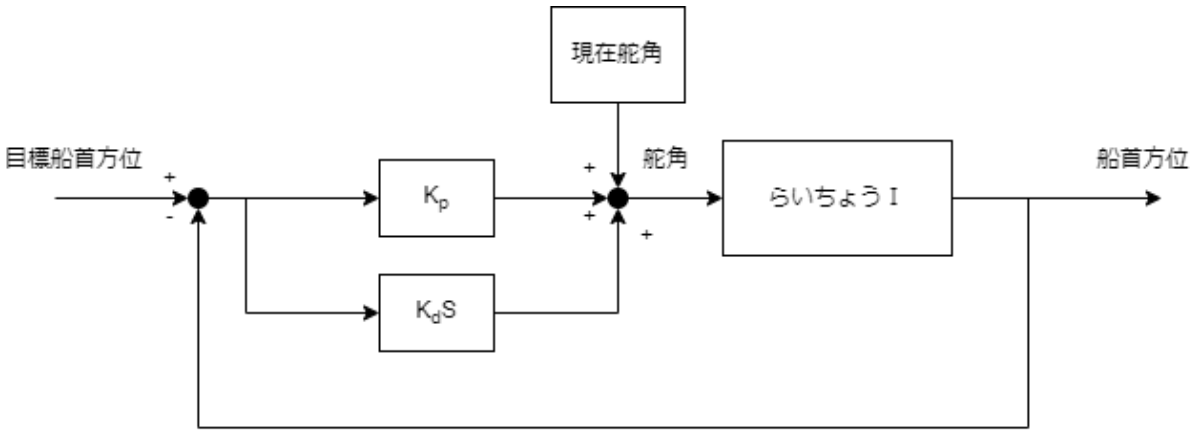


図4.1　ブロック線図

また図4.2，図4.3にオートパイロット機能のユーザーインターフェースを示す．ユーザーインターフェースには目標船首方位を送信及びオートパイロットの開始するボタンと，オートパイロットを終了するボタンをそれぞれ配置している．また船首方位を設定するにはキーボードで数字を打つ他に，現在設定船首方位に５足したり，引いたりすることができるボタンも合わせて設置し，操作性を高めている．ラベルには現在の目標船首方位の設定値やパラメーターの値を表示している．この部分は使用環境によって変更する必要があると考える．

遠隔地からの目標設定値やオートパイロットの開始，停止などの指令は3章で説明したクラウドサービスを利用し船舶へと送信している．

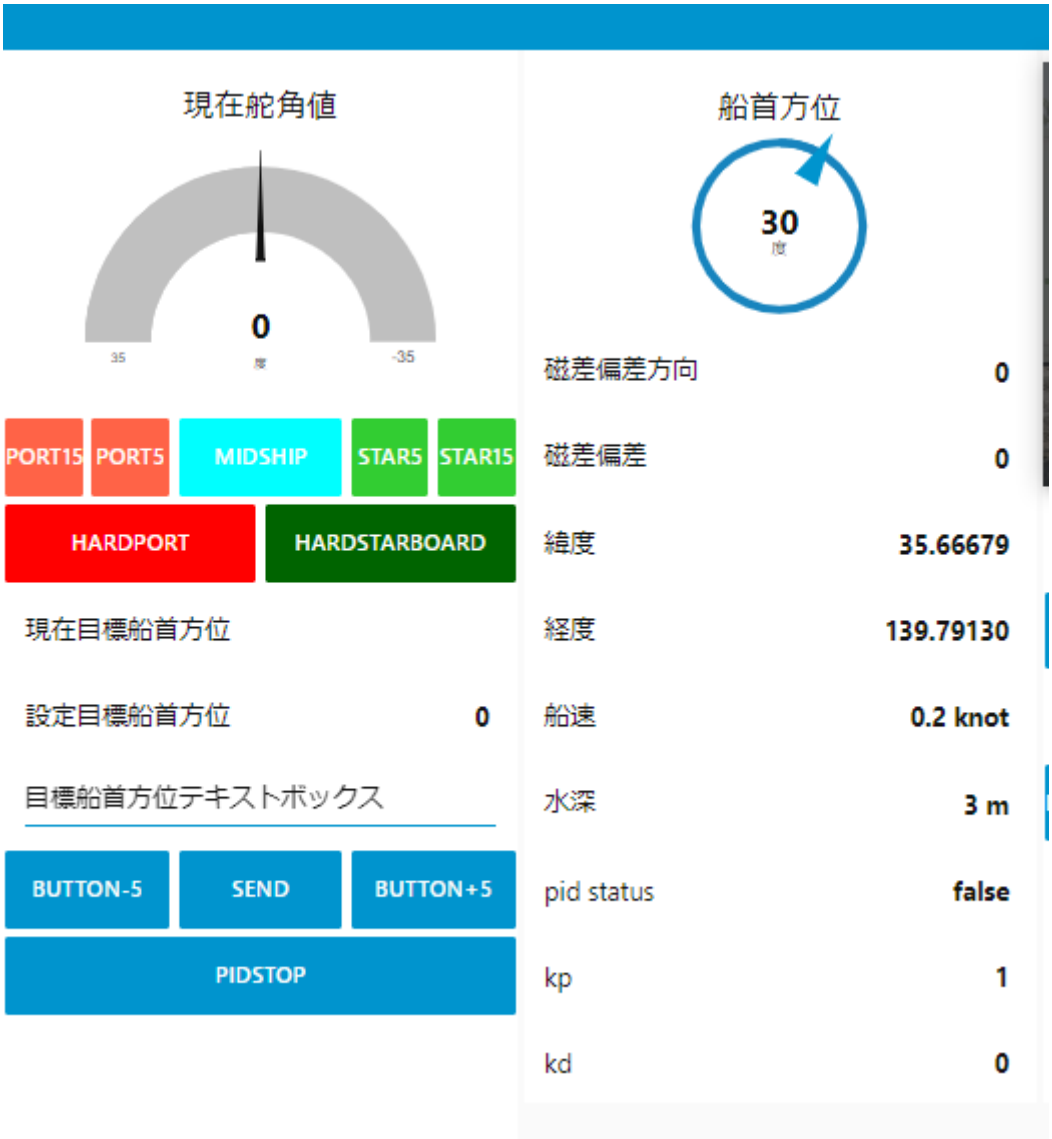


図4.2　Node-RED操作画面

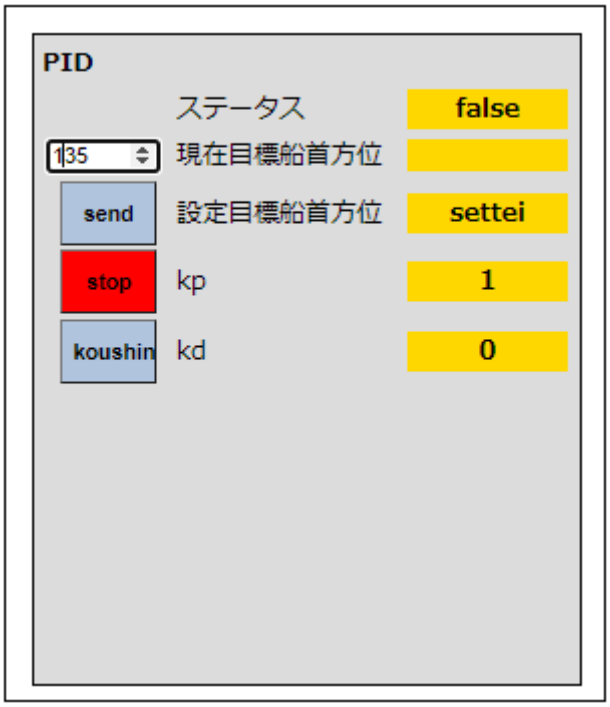


図4.3　Webページ操作画面

## 4.3 実験

上述した制御式とらいちょうＩを用いて実験を行った．実験を行った海域は図4.4に示す比較的安定した場所で行った．出力は約８kWと一定の値としている．

図4.4　実験場所

## 4.4 結果，考察

図4.5が今回行った実験の結果である．図4.5の結果を見ると，風や波の影響もあり目標船首方位に対して安定はしていないが，おおむね目標船首方位に対して実際の船首方位が追従している．また250度から240度に変針を行った場合でも，目標船首方位に追従していることが分かる．図4.6が人間を行った場合の船首方位のグラフである．人間が直線に航行を行う場合，GPSによる船首方位の精度や風や波の影響などによって船首方位は常に変動している．そのため図4.5の制御結果の船首方位の変動は航行に影響は無いと考えられる．

また図4.7が制御を行った時の，目標船首方位と現在船首方位のグラフであり，また図4.8がその時の船舶の軌跡である．図4.7の結果をみると，目標船首方位が290度から310度あたりでは船首方位が到達していない．これは実験当日図4.8の右上方面である北東からの風[11]が吹いており，これが船舶に対して影響を及ぼしているものと考えられる．

今回行った実験はある程度の出力を出し実験を行った．しかし船舶が低速の場合風や波の影響を大きく受けるので制御が難しくなると思われる．

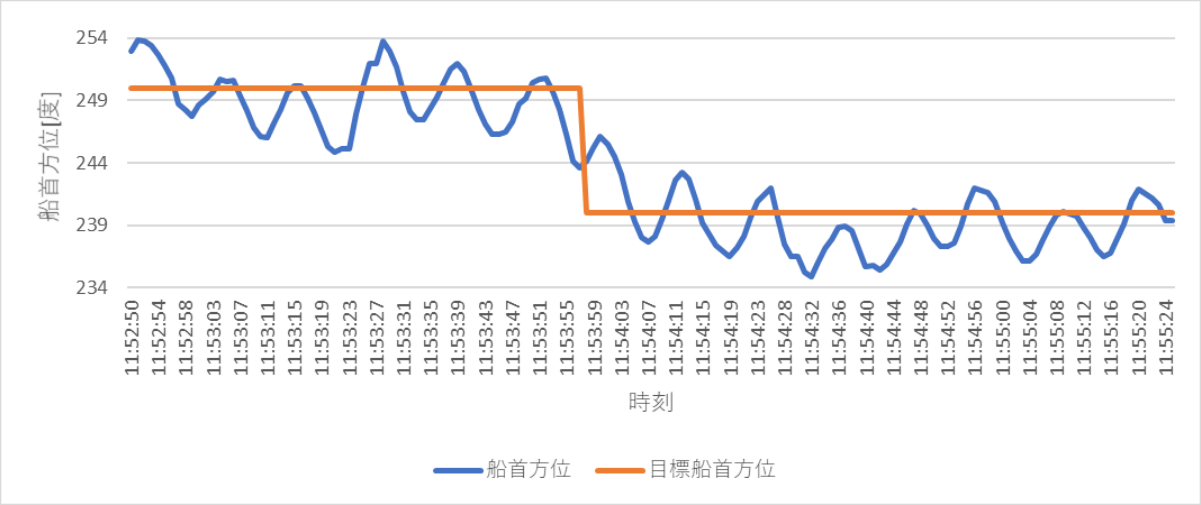


図4.5　実験結果

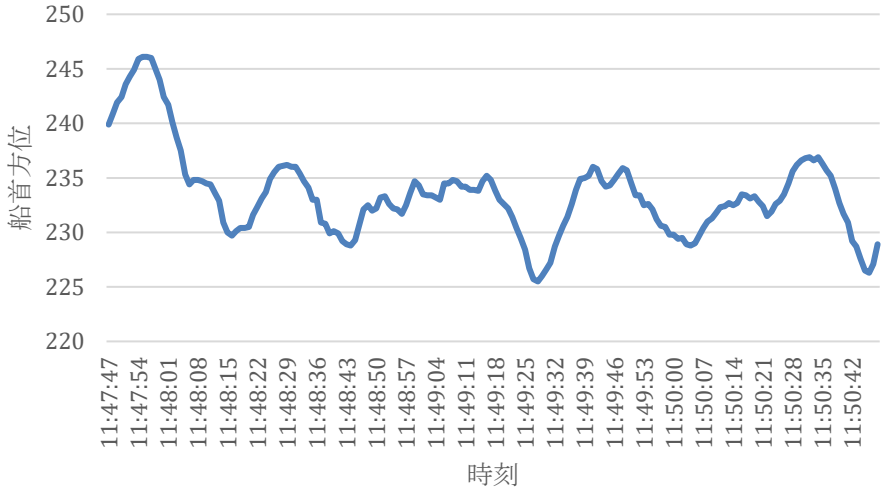


図4.6　人間の操船

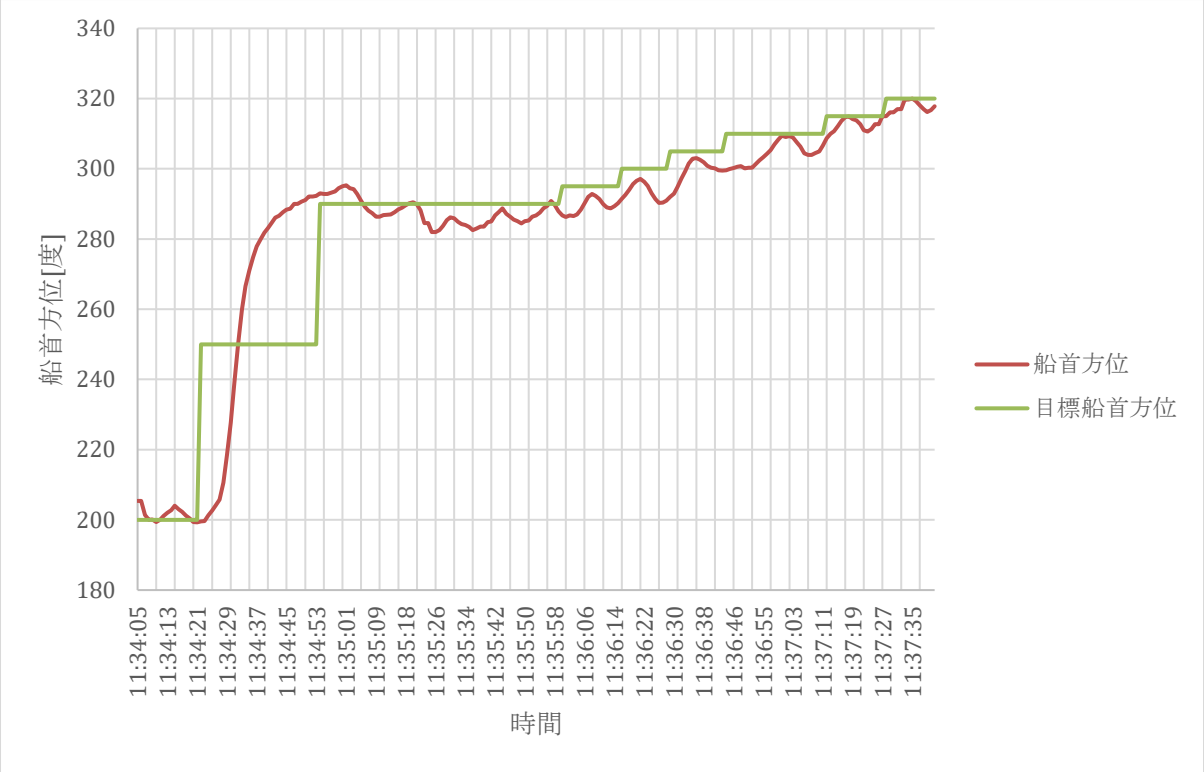


図4.7　実験結果

図4.8　実験結果の軌跡

# 5. 結言

現在，世界中のほとんどの輸出入物資は船舶によって海上輸送されている．海難の8割が人為的要因によって発生しており，また船員不足のひっ迫が予想されている．これらの理由を改善することや船舶運航の効率化などを目的に，自動運航船の開発が世界中で行われている．自動運航船は自律運航船や遠隔操船，監視システム，その間を繋ぐ通信システムなどで構成されている．本研究室でも遠隔操船，監視について研究を行ってきた．既存の遠隔監視，操船システムではCRDを利用し遠隔監視・操船を行ってきたが，ユーザーインターフェースを含めて通信をしており通信量の多さに懸念があったことや，航行データがローカルPCに保存しているため水没によるデータ消失の可能性があった．そのため本論文ではクラウドサービスを利用した遠隔操船・監視システムの開発を行った．

遠隔地と船舶間での航海データや操船指令の共有に，クラウドサービスの1つであるFirebase内のデータベースを使用し開発を行った．Firebase内にはデータベースにはFirestore Databaseと，Realtime Databaseの２つがありそれぞれ使用するシステムを構築し，またユーザーインターフェースについてもユーザーが変更しやすいNode-REDと様々な機器で使用可能なWebページを使用したシステムを構築し様々な要求に応えられるようにした．船舶と遠隔地間で共有する航行データの種類や更新頻度は，既存のシステムと同じとし遠隔監視・操船システムを構築した．本研究で構築したシステムでの実験の結果，CRDを使用したシステムと比べ，約99%通信量は削減された．また同じ航海データがUIに表示されるまでの時間を計測したところ，CRDを使用したシステムと比較し本研究で構築したシステムは約0.3秒の遅延があった．操舵席で人間が操船する場合には，1秒更新の航行データで問題なく操船可能であった．また既存のCRDを利用したシステムを使用した時でも，遠隔監視，操船可能であったことを考慮すると，この0.3秒の遅延は遠隔監視，操船に影響を及ぼさないと考えられる．さらに遠隔操船指令に対する操舵機，推進への遅延を計測した結果，船内で舵輪やスロットルレバーを使用する時とほとんど差が見られず遠隔操船が可能だった．

また本研究で構築したクラウドサービスを使用した遠隔監視，操船システムを利用したオートパイロット機能の開発を行った．目標船首方位だけを遠隔地から与え，制御自身は船舶に設置されたPCで行っている．PID制御を使用し実装を行った．また舵を切る限度を左右５度までとし舵の反応性を高めている．これらの条件の下でらいちょうＩを使用し実験を行った結果，設定方位に追従する制御を実現できた．

# 謝辞

本研究を行うにあたり，多大なるご指導やご助言を頂いた東京海洋大学　清水悦郎教授ならびに産学・地域連携推進機構　大出剛特任教授に深く感謝いたします．また，日常の議論を通じて多くの知識や示唆を頂いた産学・地域連携推進機構電池推進船プロジェクトの皆様やロボット工学研究室の皆様に深く感謝致します．

# 参考文献


1. 経済産業省，通商白書　2020，https://www.meti.go.jp/report/tsuhaku2020/pdf/02-01-03.pdf
2. 一般社団法人　日本船首協会[JP]，SHIPPING NOW 2020-2021，https://www.jsanet.or.jp/data/pdf/shippingnow2020-2021.pdf
3. 第7管区海上保安庁，海難の現況と対策，https://www.kaiho.mlit.go.jp/07kanku/gyoumu/kaiko/anzentaisaku/kainan/kainannogenkyoutotaisaku.pdf
4. 日本内航海運組合総連合会．内航船員の現状，https://www.naiko-kaiun.or.jp/union/union09/
5. 橋本博公,第4回　深層強化学習による避航操船の獲得と実証実験，KANRIN（咸臨），100号，2022年1月，p.30-34
6. 井上総一郎，森英雄，自動運航技術の開発―　MEURI2040　無人運航実証実験プロジェクト,ClassNK 技法，No.3，2021，p41-45
7. 山田智章，自動運航船関連技術に対する安全評価，KANRIN（咸臨），97号，2021年7月，p.36-39
8. 土田 侑, 仁科 卓海, 清水 悦郎，サイバーセキュリティを考慮した遠隔・自律操船システムの開発，ロボティクス・メカトロニクス講演会講演概要集，2020巻，2020
9. 総務省，クラウドサービスとは？｜インターネットを使ったサービス｜基礎知識｜国民のための情報セキュリティサイト，https://www.soumu.go.jp/main_sosiki/joho_tsusin/security/basic/service/13.html
10. Firebase，Firebase Pricing，https://firebase.google.com/pricing?authuser=2
11. 日本気象協会，東京(東京都)の過去のアメダス(2021年11月09日)，https://tenki.jp/past/2021/11/09/amedas/3/16/44132.html