TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Jan Veližanin 222880IAIB
Matthias Kaunimäe 213120IAIB

ROBOTLAEVADE NAVIGATSIOONISITUATSIOONIDE VISUALISEERIMINE SIMULAATORIS

Bakalaureusetöö

Juhendaja: Gert Kanter
Ph.D
Kaasjuhendaja: Hanna-Britt Reinpõld
BA

Tallinn 2025

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Jan Veližanin, Matthias Kaunimäe

06.06.2025

Annotatsioon

Käesoleva bakalaureusetöö eesmärgiks on luua praktiline veebirakenduse prototüüp teadustöö võrgustiku jaoks, mille abil saab visualiseerida ja analüüsida võimalikes ohusituatsioonides autonoomsete robotlaevade manöövreid. Lõputöö kirjutamise ajal teadustöö võrgustikku kuuluvad Tallinna Tehnikaülikooli Tugevalt tagatud tarkvara laboratoorium, mis on peamine projekti eestvedaja, Eesti Mereakadeemia, Åbo Akademi ja Tunis El Manar Ülikool.

Lõputöö on osa mereohutusüsteemi projektis, mille lõppeesmärk on luua veebirakendus, mis pakub kasutajatele, eelkõige merenduse tudengitele ja merendusohutus ekspertidele, tööriista millega saab analüüsida laevade manöövreid ning arendada autonoomsete laevade navigatsioonialgoritme.

Antud lõputöö tulemusena on valminud veebirakenduse prototüüp, mille abil saab kasutaja, robotlaevade operaator, analüüsida autonoomsete laevade omavahelisi manöövreid erinevates tingimustes. Visualiseerija on kaardirakendus, mis töötleb laeva teekonna faile ning jagab informatsiooni tabelite ja kuvatavate kaardikihtide vahel. Manöövrid on defineeritud COLREG reeglistiku järgi. Lõputöö kirjutamise ajal on veebirakenduse kasutaja rollis tellija.

Töö esimeses osas kirjeldatakse mereohutussüsteemi projekti üldiselt, selle erinevaid osi, visualiseerija näidistööd ShipB ning veebirakenduse funktsionaalseid ja mittefunktsionaalseid nõudeid. Töö teises osas kirjeldatakse arendust ning kasutatud tehnoloogiaid, serveri infrastruktuuri, valminud rakendust ning tulemuste valideerimist.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 25 leheküljel, 7 peatükki, 12 joonist, 1 tabel.

Abstract

Visualization of Robot Ship Navigation Situations in a Simulator

The purpose of this Bachelor's thesis is to develop a prototype of a practical web application for a network of academic partners. At the time of writing the thesis the network of academic partners consists of Tallinn University of Technology research group The Strongly Backed Software Laboratory, which is the primary researcher for the project, The Estonian Maritime Academy, Åbo Akademi and University of El Manar Tunis.

The final goal of the project, which goes beyond the scope of the thesis, is to create a web application primarily designed for maritime students and maritime safety experts, which allows for the development and analysis of autonomous robot ships navigation algorithms.

The result of this thesis is a web application prototype that is a simulation of different maneuvers that autonomous robot ships performed in potentially different situations. It takes a file with paths of ships and divides the information into map layers and tables for a comfortable viewing and analysis. The maneuvers are defined by COLREG rules. The role of the user of the application is filled by the commissioner.

The thesis is in Estonian and contains 25 pages of text, 7 chapters, 12 figures, and 1 table.

Lühendite ja mõistete sõnastik

API	Rakendusliides (*Application Programming Interface*)
Ajatempel	Ajahetk JSON failis (*Timestamp*)
COLREG	Merenduse ohutuse tagamise konventsioon (*Convention on the International Regulations for Preventing Collisions at Sea*)
GIF	Digitaalne helita video formaat (*Graphics Interchange Format*)
GML	Geograafiamärgistuskeel (*Geography Markup Language*)
JPG	Digitaalne pildi formaat (*Joint Photographic Experts Group*)
LLM	sisendeid töötlev tehisintellekt (*Large Language Model*)
MPZ	Liikumise keeluala (*Movement Prohibited Zone*)
NMA	Navigatsioonimärkide andmekogu
Prolog	Loogika programmeerimiskeel
WFS	Veebifunktsioonide teenus (*Web Feature Service*)

Sisukord

Joonistе loetelu

Tabelite loetelu

1. Sissejuhatus

Tugevalt tagatud tarkvara laboratoorium uurib tugevalt tagatud tarkvara arendamise teooriaid, meetodeid ja tööriistu, spetsialiseerudes nii tõestustele (sertifitseeritud tarkvara) kui ka testimisele [1].

Tallinna Tehnikaülikooli Eesti Mereakadeemia on ainus omalaadne merendusalast rakenduskõrgharidus-, magistri- ja doktoriõpet andev ning erialaseid uuringuid viljelev kompetentsikeskus Eestis [2].

Åbo Akademi on rootsi keeles õpetav ülikool Turku linnas Soomes . Åbo Akademi ülikoolis uuritakse ja õpetatakse inseneeriat, majandust, kunsti, sotisaalteadusi ja pedagoogikat [3].

Tunis El Manari on ülikool, mis loodi 1987. aastal. Ta on Tuneesia suurim ülikool ning hõlmab mitmeid valdkondi (tervis, vesi, keskkond, energia jne) [4].

Mereohutus on olnud oluline teema merenduse ajaloos, kuid see on tõusnud erilise teravusega päevakorda seoses autonoomsete robotlaevade kiire arenguga. Samuti on merendus tänapäeval enimlevinud kauba transpordi viis: üle 80% kogu maailma kaubandusest toimub mereteedel [5]. Merenduse tähtsus kaubanduses soosib maksimaalselt odavat ja kiiret transportimist, kuid sellega kaasnevad riskid, mille tõttu tuleb arvestada ohutusega. Tänapäeval on ohutuse tagamiseks loodud COLREG, Convention on the International Regulations for Preventing Collisions at Sea, reeglistik, mida peavad järgima kõik laevad [6].

Selleks, et COLREG reeglistikku rakendada autonoomsetele robotlaevadele on vaja marsruudi analüüsimise tarkvara. Antud lõputöö raames on arendatud välja veebirakendus (edaspidi visualiseerija), kus saab visualiseerida ja analüüsida laevade manöövreid tagantjärele erinevates situatsioonides.

2. Taustauuring

Peatükis kirjeldatakse mereohutussüsteemi projekti ning selle osi koos andmevooga, visualiseerija eelmist prototüüpi ning laevade kohustuslikke manöövreid erinevates navigatsioonisituatsioonides.

2.1 Mereohutusüsteemi projekt

Mereohutussüsteemi projekt, "Two phase path planning for fuel-efficient safe navigation", on ettevõtmine, et arendada välja autonoomsete robotlaevade jaoks navigatsioonisüsteem, mis planeerib laeva marsruute ohutult ja kütusesäästlikult [7].

2.1.1 COLREG Data Map Editor

Lõputöö kirjutamise ajal on COLREG Data Map Editor kaardiveebirakendus, mis kogub kokku andmed erinevatest avalikest allikatest. Kasutaja saab nendele andmetele lisaks sisestada enda andmed, milleks on laeva poolt läbitud marsruut või marsruudid, ilmastiku tingimused, erinevad takistused ja lubatud navigatsioonialad. Veebirakendus konverteerib informatsiooni prolog failiks ning edastab selle prolog verifitseerijasse (vt Joonis 1).

Tegemist on veebirakendusega, mida arendab Rene Kroon magistritöö raames. Selle veebirakenduse arendus toimub paralleelselt käesoleva projekti arendusega.

2.1.2 Prolog verifitseerija

Lõputöö kirjutamise ajal prolog verifitseerija saab sisendiks COLREG Data Map Editorist Prolog faili (vt Joonis 1). Verifitseerija genereerib Data Map Editori andmete põhjal kitsendustesüsteemi ja kontrollib verifitseeritavate omaduste kehtivust kitsendustesüsteemil. Verifitseerija töö tulemusena väljastatakse kitsendustele vastavad laevade trajektoorid koos vastavate mudeli parameetritega (trajektooride pikkused, kütusekulu jm). Deterministlikud trajektoorid saadetakse visualiseerijasse ja trajektooride rakendusse ning deterministlikud ohuhinnangud saadetakse ASV AI captain (ROS2) tarkvarasse (vt Joonis 2). Samuti on võimalik lähteandmeid käsitsi faktidena sisestada, et genereerida näiteks katseandmeid [7].

2.1.3 Ontoloogia tagarakendus

Ontoloogilia tagarakendus on LLM (*large language model*) mudel, mis koostab ontoloogiat, millest saab tuletada erinevaid kasutajakeeli.

Tegemist on rakendusega, mida arendab Liia Jerjomenko bakalaureuse töö raames. Selle veebirakenduse arendus toimub paralleelselt käesoleva projekti arendusega.

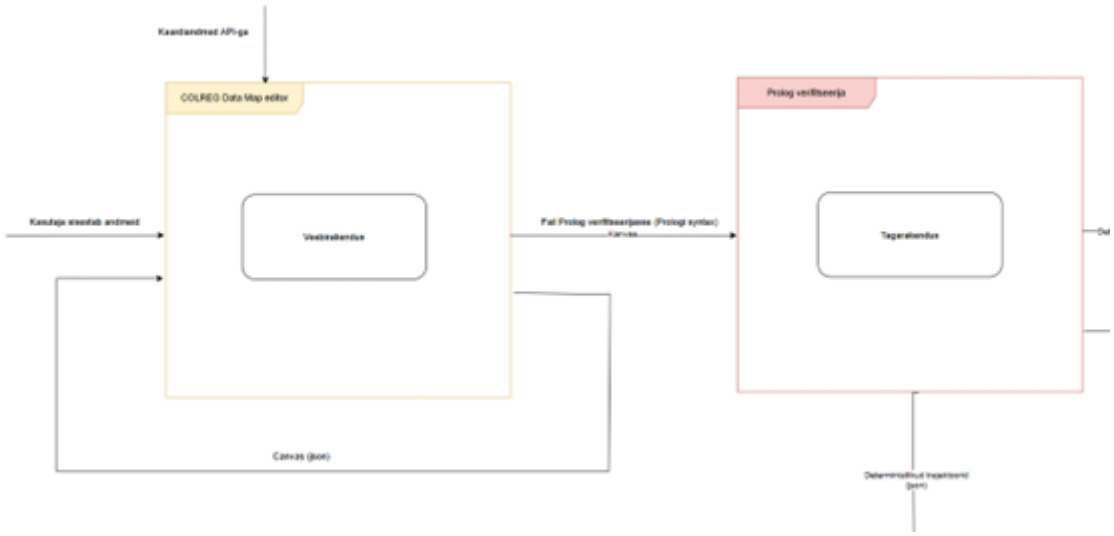
2.1.4 Stohhastiline modelleerija

Stohhastiline modelleerija koosneb kahest tagarakendusest: Tõenäosustevalideerija ning Bayesi võrkude generaator. Mõlemad rakendused saavad sisendi Prolog verifitseerijast, kust tulevad deterministlikud trajektoorid JSON formaadis (vt Joonis 3). Tõenäosusevalideerija väljund on usalduskoridori kujul joonis XY teljestiku peal JPG formaadis ning laevade tõenäolise asukoha soojuskaart GIF formaadis. Väljund edastatakse ASV AI captain (ROS2) tarkvarasse (vt Joonis 2).

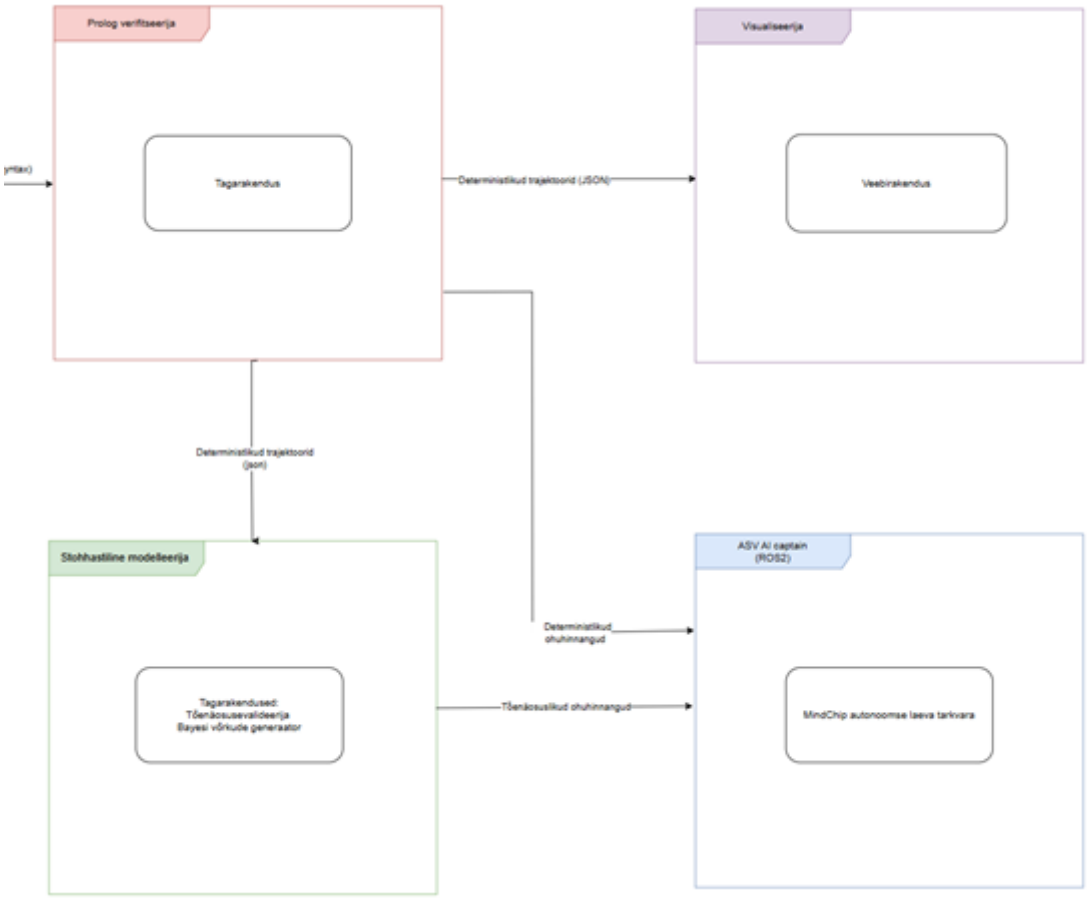
Tegemist on rakendusega, mida arendab Edvard Kallas bakalaureuse töö raames. Selle veebirakenduse arendus toimub paralleelselt käesoleva projekti arendusega.

2.1.5 ASV AI captain (ROS2)

ASV AI captain on autonoomse robotlaeva tarkvara, mis on arendatud MindChip ettevõtte poolt. Antud tarkvara on autonoomse robotlaeva tehisintellekt, mis juhib laeva [8]. Tarkvara saab prolog verifitseerijast sisendiks deterministlikud ohuhinnangud ning trajektooride rakendusest tõenäosuslikud ohuhinnangud (vt Joonis 2).



Joonis 1. Mereohutussüsteemi projekti arhitektuur lõputöö kirjutamise ajal. Andmeevoo esimene osa



Joonis 2. Mereohutussüsteemi projekti arhitektuur lõputöö kirjutamise ajal. Andmeevoo teine osa

Joonis 3. Mereohutussüsteemi projekti arhitektuur lõputöö kirjutamise ajal. Andmeevoo kolmas osa

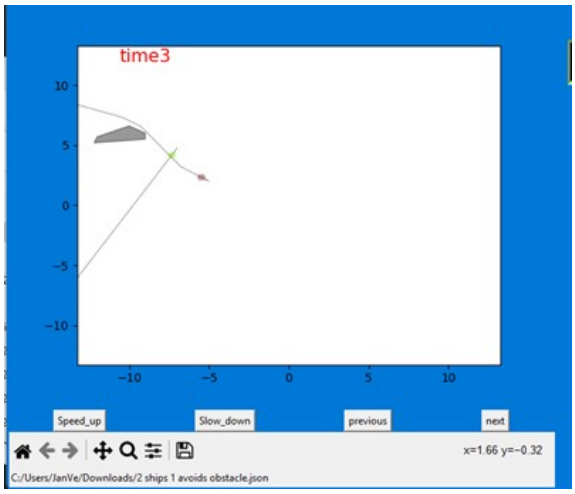
2.2 Varasem visualiseerija prototüüp ShipB

ShipB ei ole avalikult kättesaadav rakendus. Tegemist on käsitööga tellija poolt, et testida prolog verifitseerija faile ning selleks, et illustreerida visualiseerija nägemus. EDIT

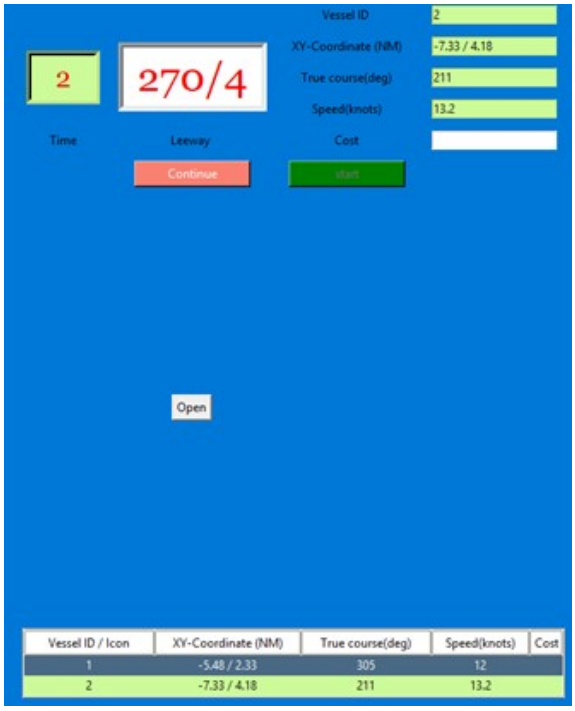
ShipB rakendus võtab vastu faile ning visualiseerib laeva marsruute koos nende trajektooridega. Samuti esineb dünaamiline tabel, mis kuvab laevade informatsiooni antud ajahetkel (vt Joonis 4 ja 5). EDIT

ShipB rakenduses esinevad järgmised puudused:

- Ei ole võimalik kaardikihte lülitada sisse või välja, kõik andmed on kuvatud.
- Ei ole visuaalset kaarditagatausta, kus saaks kuvada geograafiliste koordinaatide järgi laevu.
- Ei ole MPZ rikkumiste kontrolli.
- Ei ole ohtlike ajatemplite kontrolli.
- Ei ole manöövri ennustamist.
- Kasutajaliides ja kasutajakogemuse disain ei arvesta visuaalsete häiretega.
- Tegemist on rakendusega, mida peab alla laadima.
- Rakendus ei ole pidevalt aktiivne, tuleb iseseisvalt käivitada.
- Ei ole võimalik suhelda teiste rakendustega.



Joonis 4. ShipB rakenduse laeva trajektoori visualiseerimine



Joonis 5. ShipB rakenduse laeva informatsiooni kuvamine

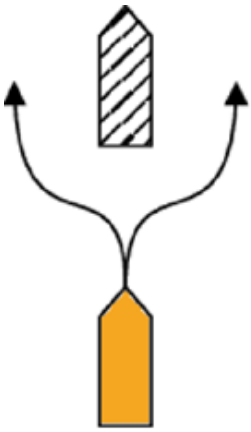
2.3 Manöövri tüübid

Käesolevas lõputöös käsitletakse COLREGi reeglites kolme peamist kohustuslikku manöövrit laevanduses: ristumine, otsene vastasseis ja möödasõit. Manöövrid on defineeritud COLREG reeglistiku järgi [6].

2.3.1 Möödasõit

COLREG reeglistikus on möödasõit kirjeldatud nelja puntkiga [6] (vt Joonis 6):

- Laev, mis sooritab möödasõitu ei tohi häirida laeva, millest sõidetakse mööda.
- Laeva loetakse möödasõitvaks, kui ta läheneb teise laeva suunast, mis on rohkem kui 22,5 kraadi ahtrikiilust tahapoole, see tähendab sellises asendis suhtes mööda mineva laevaga, et öisel ajal näeks ta ainult mööda mineva laeva ahtrituld, kuid mitte kummagi parda tulesid.
- Kui laev ei ole kindel, kas ta teeb möödasõitu teisest laevast, peab ta eeldama, et ta teeb.
- Mistahes hilisem muudatus kahe laeva vahelistes suunasuhetes ei tee mööduvast laevast ristuvat laeva käesolevate reeglite tähenduses, ega vabasta teda kohustusest hoiduda mööda minevast laevast, kuni ta on täielikult möödunud ja ohutult eemaldunud.

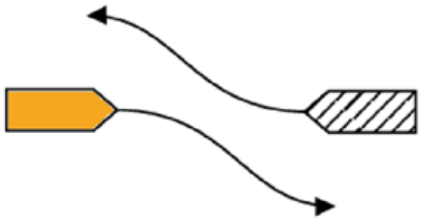


Joonis 6. COLREG Reegel 13: Möödasõit [9]

2.3.2 Vastaskurssidel lähenemine

COLREG reeglistikus on otsene vastasseis kirjeldatud kolme punktiga [6] (vt Joonis 7):

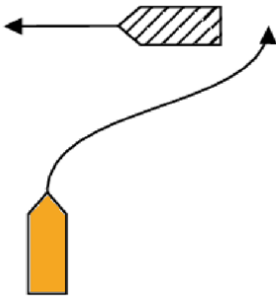
- Kui kaks mootorlaeva lähenevad teineteisele vastastikustel või peaaegu vastastikustel kurssidel nii, et on olemas kokkupõrkeoht, peab kumbki neist muutma kurssi paremale, et mõlemad mööduksid teineteise vasakust pardast.
- Olukord klassifitseeritakse kui vastaskurssidel lähenemine juhul kui kui üks laev näeb teist enda ees või peaaegu enda ees ning öösel näeks tema parda-, topi- ja ahtritulesid.
- Kui laev kahtleb, kas selline olukord on vastaskurssidel lähenemine, peab ta eeldama, et see on, ja tegutsema vastavalt.



Joonis 7. COLREG Reegel 14: Vastaskurssidel lähenemine [9]

2.3.3 Lõikuvatel kurssidel lähenemine

Lõikuvatel kurssidel lähenemine on olukord, kus kaks mootorlaeva lõikuvad nii, et nende vahel on kokkupõrkeoht. Laev, kelle paremal küljel asub teine laev, peab hoiduma teisele teele jäämast ning võimaluse korral vältima teise laeva eest läbi sõitmist (vt Joonis 8) [6].



Joonis 8. COLREG Reegel 15: Lõikuvatel kurssidel lähenemine [9]

3. Analüüs

Käesolevas peatükis analüüsitakse projekti funktsionaalsed ja mittefunktsionaalsed nõudeid, mis on formuleeritud tellija poolt. Funktsionaalsed nõuded on tehnilised ning peamiselt hõlmavad enda alla, mida peab süsteem tegema sisendite ja väljunditega. Mittefunktsionaalsed nõuded on mitte tehnilised, näiteks kasutajakogemus, lihtsus, süsteemi kiirus [10].

3.1 Visualiseerija funktsionaalsed nõuded

- Veebirakendus saab sisendi prolog verifitseerijast ning teisendab sisendandmed visualiseerimiseks sobivale kujule.
- Veebirakendusse saadetud sisendandmed kuvatakse laevade informatsiooni dünaamilises tabelis.
- Kasutaja saab manuaalselt sisestada sisendfaili.
- Veebirakendus genereerib laevadele MPZ, ehk Movement Prohibited Zone, joone (vt Tabel 1).
- Veebirakenduse simulaatori taustaks on interaktiivne maailma kaart.
- Interaktiivsel kaardil on rippmenüü, milles saab kaardikihte lülitada sisse ja välja. Korraga saab kuvada mitu kihti.
- Veebirakenduses saab minna edasi-tagasi ajatemplite vahel.
- Veebirakenduses saab laevade trajektooride aegridades käsitsi minna valitud ajatemplile.
- Veebirakendus näitab, mis ajatemplil on hetkel simulatsioon, ning mis on maksimaalne ajatempel.
- Veebirakenduses on simulatsiooni esitlusrežiim, mille kiirust saab muuta. Simulatsiooni kiirus on kuvatud kasutajale.
- Veebirakenduse esitlusrežiimi saab panna pausile, taaskäivitada ja minna algusesse tagasi.
- Veebirakendus otsustab sisendfaili põhjal, mis manöövriga on tegemist.
- Veebirakendus leiab sisendfailist kõik ajatemplid, kus on ohtlik olukord.
- Veebirakendus on telefonis kasutatav.

3.2 Visualiseerija mittefunktsionaalsed nõuded

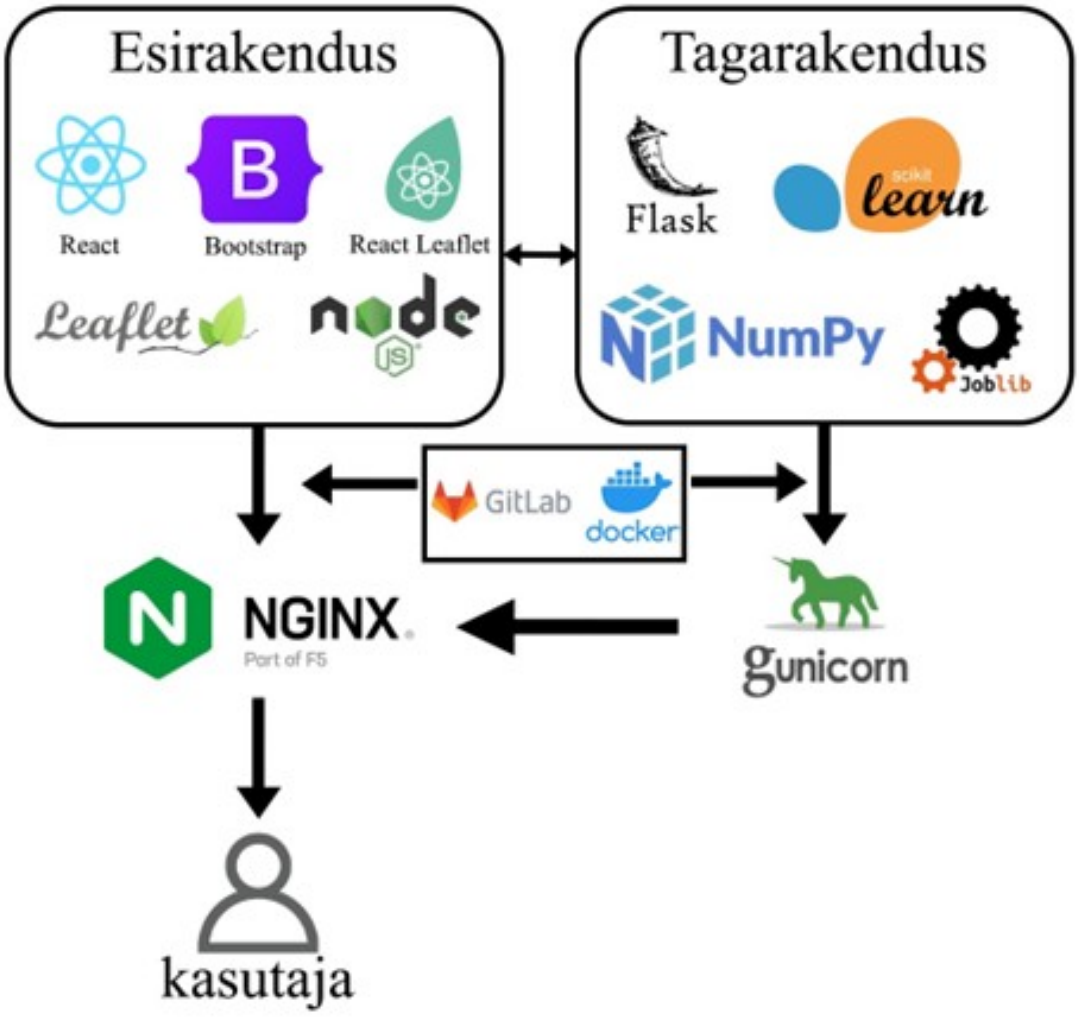
- Veebirakendus peab olema kasutajasõbralik, sealhulgas arvestama visuaalsete häiretega.
- Veebirakenduses on öö- ja päevarežiim.
- Veebirakendus töötab ööpäevaringselt ilma tõrgeteta.
- Veebirakendust on võimalikult lihtne ja loogiline kasutada.
- Veebirakendus on sõltumatu, ei ole vaja midagi muud alla laadida.
- Veebirakendus peab olema inglise keeles.

4. Arendus

Käesolevas peatükis käsitletakse Visualiseerija arhitektuuri, kasutatud tehnoloogiaid ning põhjendatakse valikuid.

4.1 Arhitektuur

Visualiseerija koosneb kahest põhikomponendist: esirakenduse kasutajaliides ning tagarakenduse masinõppe ja andmetöötlus osa. Visualiseerija on veebirakendus, mis on töötlusahelaga pidevalt serveris üleval ning kasutajale kättesaadav (vt Joonis 9).



Joonis 9. Visualiseerija arhitektuur

4.2 Esirakendus

Järgnevas alapealtükis kirjeldatakse esirakenduses kasutatavaid tehnoloogiaid.

4.2.1 React

React on JavaScripti teek, mis teeb veebiarenduse lihtsamaks kasutades komponente funktsionaalsuse kokku panemiseks [11].

React valiti välja selle lihtsuse pärast, spetsiaalselt kaardirakenduste loomiseks loodud React-Leaflet teegi olemasolu pärast ning autorite varasema kogemuse põhjal.

4.2.2 Leaflet ja React-Leaflet

Leaflet on JavaScripti teek, mida kasutatakse peamiselt interaktiivsete kaardiveebirakenduste loomiseks [12].

React-Leaflet on JavaScripti teek, mis lihtsustab Leafleti kasutamist Reacti veebirakenduste arendamisel, muutes funktsionaalsused komponentideks [13].

Leaflet oli valitud arendustööks kahel põhjusel: Leaflet on loodud kaardirakenduste arendamise lihtsustamiseks ning Leafleti jaoks on olemas teek React-Leaflet, mis on loodud spetsiaalselt Reacti jaoks.

4.2.3 Bootstrap

Bootstrap on avatud lähtekoodiga raamistik, täpsemalt on see HTMLi, CSSi ja JavaScripti komponentide kogu. Bootstrap võimaldab luua mobiilisõbralikke veebirakendusi [14].

Bootstrap sai valitud, sest nimetatud teegi komponentidega on mugav arendada esirakendust ning autoritel on sellega varasem kogemus.

4.2.4 Jest

Jest on avatud lähtekoodiga JavaScripti testimise raamistik, mis keskendub lihtsusele. Raamistik on optimiseeritud paljude tehnoloogiate jaoks, sealhulgas React [15]. Jest sai valitud, sest antud raamistik teeb Reacti komponentide testimise lihtsaks.

4.3 Tagarakendus

Järgnevates alapeatükkides kirjeldatakse tagarakenduse tehnoloogiaid.

4.3.1 Node.js

Node.js on tasuta, avatud lähtekoodiga ja platvormideülene JavaScripti käituskeskkond, mis võimaldab arendajatel luua servereid, veebirakendusi, käsureatööriistu ja skripte.

Node.js sai valitud selleks, et esirakendust saaks lihtsasti katsetada ning hiljem majutada serverile ja suhelda Nginx-iga [16].

4.3.2 Flask

Flask on Pythoni raamistik, mis võimaldab kiiresti ja lihtsalt luua keerulisi tagarakendusi. Flask on ülesehitatud kergelt ja modulaarselt, mis sobib hästi nii väikestele kui ka suurematele veebirakenduste projektidele [17].

Flask sai valitud selle pärast, et masinõppe mudeli arenduseks on kõige mõistlikum kasutada Pythonit, sest see on selles valdkonnas enim levinud programmeerimiskeel. Flask on Pythoni põhine raamistik, mis võimaldab luua tagarakenduse ning selle kaudu pääseb lihtsalt masinõppe mudelile ligi.

4.3.3 Scikit-learn

Scikit-learn on avatud lähtekoodiga pythoni teek, mis on mõeldud andme analüütika ja masinõppe mudelite loomiseks. Ta on arendatud NumPy, SciPy ja Matplotlib teekide baasil, mis on kõik andmeanalüütikaks mõeldud teegid [18].

Scikit-learn sai valitud, sest ta lihtsustab masinõppe mudelite kirjutamist ning autoritel on teegiga varasem kogemus.

4.3.4 Joblib

Joblib on Pythoni teek, mida saab kasutada väiksemate konveierite arendamiseks. Joblib on optimiseeritud NumPy massiivide töötluseks [19].

Joblib sai valitud selle põhjal, et teek võimaldab jooksutada masinõppe mudeli instantse.

4.3.5 NumPy

NumPy on avatud lähtekoodiga Pythoni teek andmeanalüütikaks, mis pakub massiivi- ja maatriksitoiminguid ning matemaatilisi funktsioone [20].

NumPy sai valitud lähtudes selle lihtsusest ja sobilikkusest masinõppe mudeli loomiseks ning autorite varasemast kogemusest.

4.3.6 PyTest

Pytest on avatud lähtekoodiga Pythoni raamistik, mis võimaldab kirjutada lihtsaid ja loetavaid teste [21].

Pytest sai valitud lähtudest selle sobilikkusest ja lihtsusest Pythoni mock testide kirjutamiseks ning autorite varasemast kogemusest.

4.3.7 Algoritm

Algoritmi valimiseks katsetasid autorid Google Collab keskkonnas kahte algoritmi. Ansambelõpe Random Forest klassifikaator ning Multi-layer Perceptron klassifikaator. Mudeli treenimisel kasutati 20% andmestikust valideerimise andmestikuna. Tulemused ja andmestik dataset_maneuver on kättesaadavad käesoleva lõputöö GitLabi repositooriumist. Igat mudelit treeniti viis korda.

Random Forest Classifier on masinõppe mudel, mis kombineerib mitmeid otsustuspuude põhjal tehtud ennustusi, et suurendada täpsust ja vähendada üleõppimist [22]. Mudeli treenimisel tuli täpsuse tulemuseks kõigis katsetes 100%. Tulemus viitab overfitting-ule, sest katseandmete testimisel ei ole vigu.

Multi-layer Perceptron klassifikaator on tehisnärvivõrk, mis koosneb mitmest omavahel ühendatud kihist ja suudab õppida keerukaid mustreid klassifitseerimisülesannete lahendamiseks [23]. Mudeli treenimisel tuli täpsuse tulemuseks kolmes katses 100%. Tulemus viitab taaskord overfitting-ule.

Mõlema algoritmi testimisel ei olnud tulemustes märkimisväärset erinevust. Overfitting tuleneb andmestiku väikesest mahust. Autorid valisid ansambelõpe masinõppe mudeli oma varasema kogemuse põhjal.

4.4 Infrastruktuur

Järgnevates alapeatükkides kirjeldatakse infrastruktuuri tehnoloogiaid.

4.4.1 Gitlab CI/CD

Gitlab CI/CD on pidev tarkvaraarenduse konveiermeetod, millega saab peatamata jooksutada, testida ja avaldada koodi või muudatusi [24].

Tegemist on teenusega, mida saab kasutada ainult GitLabi repositooriumites, millest tulenes valik ehitada töötlusahel sellega.

4.4.2 Docker

Docker on tarkvara, millega saab luua isoleeritud konteinereid, mis jooksutavad programmi koodi. Selle eeliseks on see, et ühekorra pakendatud tarkvara saab jooksutada peaaegu igalpool, kui docker on toetatud [25].

Docker sai valitud selleks, et veebirakenduse töötlusahelat saaks lihtsalt ja muretult üle viia teisele paltformile, kui selleks tekib vajadus.

4.4.3 Nginx

Nginx on HTTP-veebiserver, pöördproxy ehk vahepealne server, mis proxy'b päringu õigesse serverisse, TCP/UDP proxy-server ja meiliproxy-server [26].

Nginx sai valitud selleks, et saaks serverile majutatud veebirakendusele tagada ligipääsu kasutajale ning lubada esirakendusel suhelda tagarakendusega.

4.4.4 Gunicorn

Gunicorn on Python'i WSGI HTTP-server UNIX-i jaoks. See kasutab eelhargnemisega tööprotsesside mudelit (pre-fork worker model). Gunicorni server ühildub laialdaselt erinevate veebiraamistikega, on lihtsa ülesehitusega, vähese ressursikasutusega ning kiire toimimisega [27].

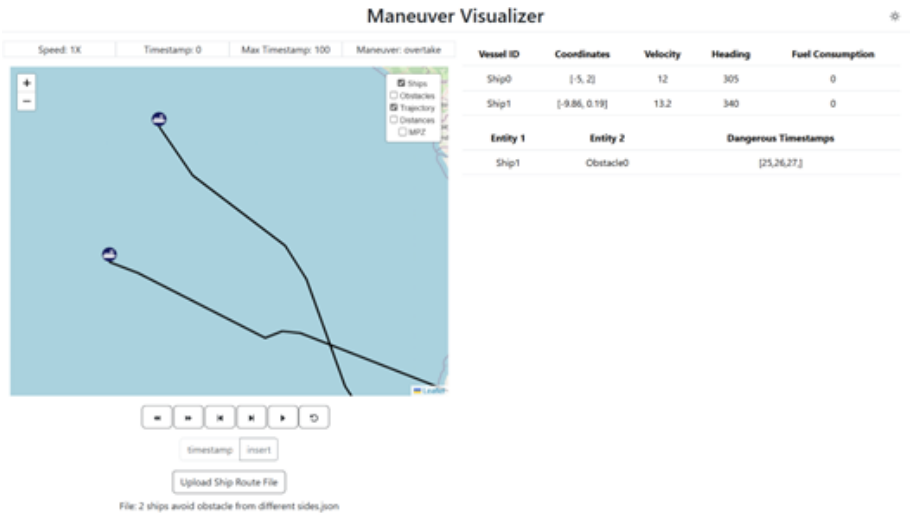
Gunicorn sai valitud selleks, et Flaski tagarakendust saaks majutada serverile ja et see saaks suhelda Nginx-iga.

5. Rakenduse ülevaade

Käesolevas peatükis kirjeldatakse valminud prototüüpi ning funktsionaalsusi, mis olid loodud, aga jäid lõpp-prototüübist välja. Lõputöö kirjutamise ajal asub rakendus internetis üleval http://193.40.156.123/ hüperlingil. Konveier on GitLabi repositooriumi harus "pipeline-test".

5.1 Esirakendus

Visualiseerija täidab kahte peamist ülesannet: simuleerib üles laetud failist laevade teekonda ning kuvab informatsiooni iga ajatempli kohta. Prolog verifitseerijast tulnud fail tuleb üles laadida nupust „Upload Ship Route File“ (vt Joonis 10).



Joonis 10. Visualiseerija kasutajaliides

5.1.1 Simulaator

Visualiseerijasse saab laadida prolog verifitseerijast tulnud JSON faili, mis saadetakse tagarakendusse töötlemiseks ning informatsioon jagatakse kaardikihtide vahel (vt Tabel 1). Informatsioon kuvatakse seejärel nii tabelites, kui ka informatsioonikastides. Simulaatori paremal üleval nurgas on dünaamiline rippmenüü, kust saab lülitada kihte sisse ja välja. Kihid ilmuvad dünaamiliselt olenevalt faili sisust, näiteks kui laeva marsruudil takistusi ei esinenud, siis *obstacles* kihti ei ilmu valikusse.

Tabel 1. Kaardikihid

Kiht	Kirjeldus
Ships ehk laevad	Laevade kiht kuvab laeva ikoonid nende koordinaatidel antud ajahetkel. Laeva peale vajutades kuvatakse laeva ID.
Obstacles ehk takistused	Takistuste kiht kuvab mittelaevatava ala, kus asuvad vrakid, takistused, kivid, jne. Takistuse peale vajutades kuvatakse takistuse ID.
Trajectories ehk trajektoorid	Trajektooride kiht kuvab kogu laeva teekonna igal ajahetkel, et seda oleks mugavam analüüsida. Trajektoorile vajutades kuvatakse sellel liikuvat laeva ID'd.
Distances ehk distantsid	Laevade vahel on MPZ ehk *movement prohibited zone*, see on distants, mida laevad peaksid üksteisest ja takistustest hoidma. Distantside kiht kujutab laevade vahelisi jooni, mis näitavad laevade omavahelist kaugust. Jooned toimivad soojusjoontena ning vahetavad värvi olenevalt distantsist. Visualiseerija võrdleb laevade ja objektide vahelist kaugust. ■ Joon on roheline kui distants on 1000m või üle. ■ Joon on kollane kui distants on alla 1000m ■ Joon on punane kui distants on alla 500m
MPZ (*Movement Prohibited Zone*) line	MPZ joon on 6 minuti ajavektor, mis on jagatud 50neks segmendiks. Olukord on klassifitseeritud ohtlikuna kui laev läheneb 50m lähedusse.

5.1.2 Ülejäänud kasutajaliides

Visualiseerija kuvab faili sisu laevade kohta dünaamilises tabelis: laeva ID, koordinaadid, liikumiskiirus, liikumissuund, kütusekulu. Samuti kuvatakse informatsioonikastides simulatsiooni praegust kiirust, hetke ajatemplit, maksimaalset ajatemplit ning kahe laeva vahelist manöövritüüpi.

Samuti esineb ohtlike ajatemplite tabel. Tabel kuvab üksuste paare, näiteks kaks laeva või laev ja takistus, ning näitab veerus *Dangerous Timestamps*EDIT ajatempleid, kus on toimunud MPZ rikkumine. Informatsioon on saadud tagarakendusest.

Simulaatori all on nupud, millega saab:

- Simulatsiooni kiiremaks või aeglasemaks muuta.
- Simulatsiooni ajatemplite vahel edasi-tagasi liikuda. Kui minna maksimaalselst ajatemplist üle, läheb simulatsioon tagasi ajatemplile 0.
- Simulatsioon sisse lülitada või pausile panna.
- Simulatsioon taaskäivitada
- Uue laeva manöövri faili lisada, kasutajaliides kuvab, mis fail on hetkel kasutuses.
- Ajatemplit manuaalselt sisestada

5.1.3 Kasutajaliides ja kasutajakogemus

Kasutajaliides on disainitud lähtudes ligipääsetavuse põhimõtetest, eeskätt visuaalsete häiretega kasutajaid arvestades [28].

Kasutajaliidese infoedastus põhineb peamiselt kujunditel ja tekstil. Kogu liides (v.a simulaator) on mustvalge ning on võimalus vahetada öö- ja päevarežiimi vahel, et puhata silmi. Simulaator ise on samuti kujundatud värvinägemise häiretega kasutajaid arvesse võttes:

- Laevad on sinised ringid, mille keskel on ikoon.
- Takistused on hallid polügoonid.
- Trajektoorid on mustad jooned.
- Soojusjoon on roheline, kollane või punane.
- MPZ joon on sinine ning eristub laevadest oma kuju poolest.

Kasutajaliideses olev tekst on dünaamiliselt muutuv erinevatele ekraani suurustele, tänu sellele on võimalik teksti suurust kohandada kasutajale vajalikuks suuruseks.

5.2 Tagarakendus

Tagarakendus koosneb peamiselt kahest funktsionaalsusest: manöövri ennustamise masinõppe mudelist ning ohtlike ajatemplite arvutamisest.

5.2.1 Masinõppe mudel

Tagarakenduse masinõppe mudel koosneb kahest API liidese lõpp-punktist: /api/train-model ning /api/predict-maneuver.

Train-model lõpp-punkt käivitab konveieri, mis hoiab treenitud masinõppe mudeli instantsi kasutaja sessiooni lõpuni.

Predict-maneuver edastab JSON faili masinõppe mudelisse ning tagastab esirakendusele masinõppe mudeli vastuse.

Esimese faili sisestamisel esirakendus saadab mõlemasse lõpp-punkti POST päringud, et treenida mudel, mis jääb sessiooni ajaks tööle. Teine päring saadetakse manöövri ennustamisse, kust tuleb vastus. Edaspidiste faili sisestamistega saadetakse ainult üks päring /api/predict-maneuver-isse.

5.2.2 Ohtlikud ajatemplid

Tagarakendusel on olemas API lõppunkt /api/dangerous-timestamps. Faili sisestamisel edastab esirakendus JSON faili POST päringuga lõppunkti, kus töödeldakse fail, et arvutada välja, kus toimuvad MPZ rikkumised.

Lisaks JSON faili töötlemisele arvutab tagarakendus välja MPZ joone koordinaadid. MPZ joon on 6 minutiline ajavektor. Visualiseerija jagab ajavektori 50 segmendiks. Visualiseerija klassifitseerib ohtliku olukorra kui laev läheneb 50m lähedusse.

Iga koordinaadi punkt JSON failis vastab ühele meremiilile. Üks meremiil on 1852 meetrit [29]. Laev rikub MPZ, kui ta läheneb teisele laevale või takistusele 1000m lähedusse, kui tegemist on laeva vööriosaga, siis distants tekib 6 minutilisest ajavektorist.

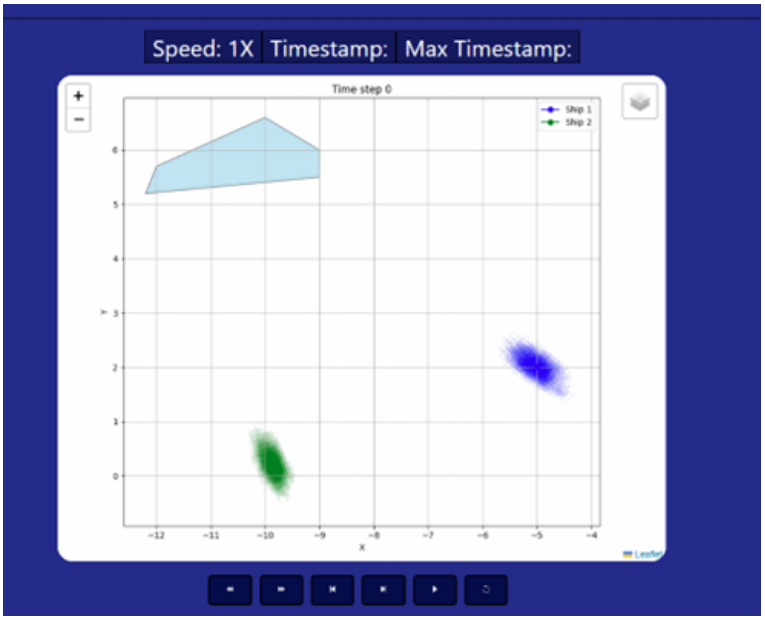
5.3 Prototüübist väljajäänud funktsionaalsused

Antud peatükis käsitletakse realiseerimata jäänud nõudeid või arendatud nõudeid, mis jäid siiski projekti lõpptulemusest välja ning selle põhjused.

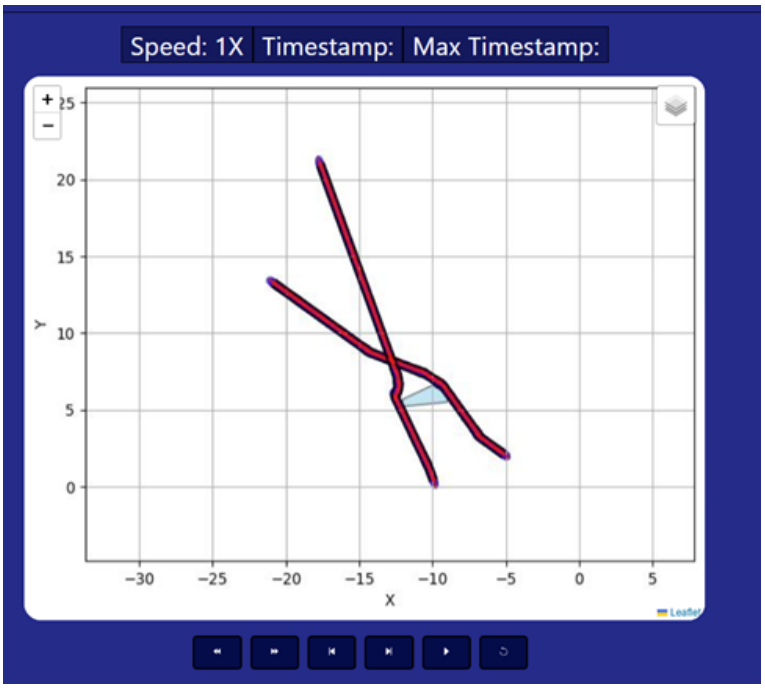
5.3.1 Soojuskaart ja usalduskoridor

Projekti algse plaani järgi pidi visualiseerija saama sisendit ka stohhastilisest modelleerijast. Käesoleva töö kirjutamise ajal soojuskaardi väljundfail on GIF formaadis ning usalduskoridori JPG formaadis.

Leaflet võimaldab pidevalt jälgida interaktiivse kaardi piiri. Seda funktsiooni kasutades arendasid autorid kaardikihid mis pidevalt uuendavad pildifaili asukohta kaardi peal. Usalduskoridor on JPG fail, seega üks pilt, mis lülitub sisse ja välja (vt Joonis 12). Soojuskaardi GIF failis on samapalju kaadreid kui laeva marsruudi failis ajatempleid. Soojuskaardi töötlemiseks kasutasid autorid teeki gifuct-js, mis võimaldab lihtsalt laadida, dekodeerida ja töödelda GIF faile [30]. Igal ajatemplil kuvatakse vastavat kaadrit (vt Joonis 11).



Joonis 11. Soojuskaardi GIF faili visualiseerimine



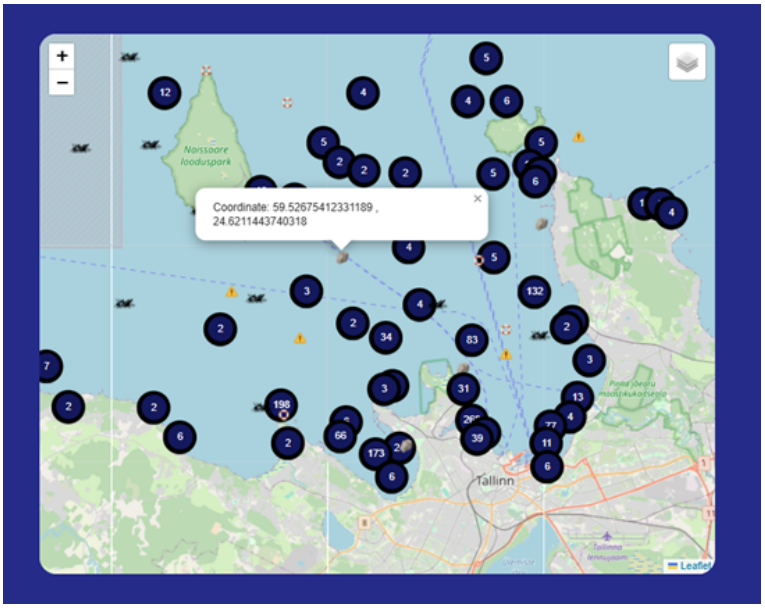
Joonis 12. Usalduskoridori JPG faili visualiseerimine

Lahendus jäi projektist välja, sest ta ei olnud praktiliselt kasutatav, ega sobinud veebirakenduse visuaalse stiiliga ning tulevikus plaanitakse uuendada stohhastilise modelleerija väljundfaili tüüpe.

5.3.2 Maailmakaardi adnmete pärimine ja visualiseerimine

Projekti algse plaani järgi pidi visualiseerija pärima andmeid erinevatest avalikest allikatest.

Visualiseerija esimene prototüüp päris andmeid Nutimerest, mis on Maa- ja Ruumiameti kaardirakendus, mis visualiseerib erinevat informatsiooni Eesti ranniku kohta [31]. Kõik need andmed on avalikult kättesaadavad WFS-i kaudu (*web feature service*), mis on geograafiliste andmete pärimise teenus. Visualiseerija päris GeoJSON faili ning töötles selle GML-i (*Geography Markup Language*), mis on esirakendusele töötlemiseks sobiv formaat. Edasi visualiseerija töötles ja kuvas andmed simulaatoris. Andmemahu suuruse tõttu ja müra vähendamise eesmärgil võeti kasutusele Leaflet.marketcluster, mis on leafleti kaardi markerite gruppeerimise pistikrakendus [32] (vt Joonis 13).



Joonis 13. Andmete kuvamine ja klasterdamine

Andmete pärimine ja kuvamine kihtidena jäi Visualiseerija projektist välja, sest COLREG Data Map Editor on selle jaoks sobilikum veebirakendus.

5.3.3 Rakenduse ühendamine

Algselt oli plaanis ühendada rakendused ja saada andmevoog tööle. Käesoleva lõputöö kirjutamise ajaks ei jõutud kõiki rakendusi ning nende lõpp-punkte valmis. Samuti pole otsustatud, milline hakkab olema projekti JSON faili lõplik struktuur ning kuidas andmevoog hakkab töötama. Eelnevalt nimetatud põhjuste tõttu jäi rakenduste ühendamine lõputöö projektist välja.

6. Tulemuste analüüs ja valideerimine

Käesoleva bakalaureuse töö kasutaja rolli täidab tellija, mistõttu autorid valideerisid oma tulemusi kolmel meetodil: automaattestid, regulaarne funktsionaalsuse demonstreerimine tellijale ning võrdlus ShipB rakendusega

6.1 Automaattestid

Autorid kirjutasid automaattestid selleks, et valideerida veebirakenduse funktsionaalsust. Samuti tulevikus on lihtsam automaattestid testivad:

- Faili sisestamist.
- Vale failitüübi ja vale JSON struktuuriga faili sisestamist.
- Mock testid esirakendsue API-dele.
- Mock testid tagarakenduse API-dele.
- Mudeli treenimine ilma andmestikuta.
- Mudeli edukas treenimine.

6.2 Võrdlus ShipB rakendusega

ShipB on tellija väljatöötatud rakendus, mis visualiseerib laeva marsruudifaile, mis tulevad prolog verifitseerijast. Visualiseerijas on kõik samad funktsionaalsused olemas:

- Faili sisu kuvamine simulaatoris.
- Dünaamiline tabel laeva informatsiooniga.
- Ajatempli näitamine.
- Simulatsiooni kiiruse muutmine, peatamine ja taaskäivitamine.

Veebirakendus on parem, sest tellija poolt formuleeritud funktsionaalsused on implementeeritud lisaks ShipB funktsionaalsusele.

6.3 Kasutajaga verifitseerimine

Lõputöö kirjutamise ajal autorid pidasid regulaarseid koosolekuid tellijaga, et demonstreerida tulemusi ja funktsionaalsusi. Tellija andis oma tagasiside, mis sai alati rakendatud.

6.4 Merenduse eksperdiga verifitseerimine

Autorid esitasid visualiseerija merenduse ekspertidele Margus Haljaste ja Eha Mugamäe Riigilaevastikust. Tagasisides ekspert soovitas muuta laeva MPZ joone ajavektoriks. Laevade puhul tüüpiline kiirusvektor on 6 minutit.

7. Kokkuvõte

Taustauuringu käigus selgus, et mereohutussüsteemi projektile on vaja arendada visualiseerijat, et praegused arendajad saaksid visualiseerida prolog verifitseerija faile ning, et autonoomsete robotlaevade operaatorid saaksid tulevikus laeva marsuutidel tehtud manöövreid analüüsida.

Käesoleva lõputöö raames loodi veebirakendus, mis pakub mereohutusüsteemi projektile prototüüpi autonoomsete robotlaevade marsruutide manöövrite visualiseerimiseks ning projekti arenduse ajaks paremat lahendust kui ShipB rakendus. Veebirakendus võimaldab visualiseerida prolog verifitseerijast tulnud JSON faile ning jagab nende informatsiooni kaardikihtide ja dünaamiliste tabelite vahel, et kasutaja saaks faili sisu analüüsida.

Veebirakendus on mõeldud prototüübina ning potentsiaalse lähenemisena visualiseerija arendamiseks.

Edasiarendamiseks on mitmeid võimalusi:

- Andmestikku on vaja suurendada. Tuleb välja mõelda rohkem olukordi ning tellija koormuse vähendamiseks õppida prolog verifitseerijas ise andmeid genereerima.
- Praegune masinõppe mudel ennustab ainult kahe laeva vahelisi manöövreid. Tulevikus peab käsitlema ka mitut laeva paari korraga.
- Tagarakenduses pole vaja arvutada ohtlike ajatempleid, see peaks toimuma ainult esirakenduses.
- Rakenduse sisendfail muutub tulevikus, tuleb oodata kuni lepitakse kokku lõppfaili formaat ning integreerida selle töötlemine.

Käesolev veebirakendus on majutatud Tallinna Tehnika Ülikooli serveril ning on saadav kõigile lingil http://193.40.156.123/.

Kasutatud kirjandus

[1] Tallinna tehnika Ülikool. *Tarkvarateaduse instituut uurimisrühmad*. Accessed: 07-04-2025. URL: https://taltech.ee/tarkvarateaduse-instituut/uurimisruhmad#p22611.

[2] Tallinna tehnika Ülikool. *TalTech Mereakadeemia*. Accessed: 14.03.2025. URL: https://taltech.ee/mereakadeemia.

[3] Åbo Akademi University. *About Åbo Akademi University*. Accessed: 23.03.2025. URL: https://www.abo.fi/en/about-abo-akademi-university/.

[4] University of Tunis El Manar. *Presentation of University of Tunis El Manar*. Accessed: 23.03.2025. URL: https://utm.rnu.tn/utm/fr/universite--presentation.

[5] UN trade developement. *Review of Maritime Transport 2024*. Accessed: 2025-05-08. URL: https://unctad.org/publication/review-maritime-transport-2024.

[6] International Maritime Organization. *Convention on the International Regulations for Preventing Collisions at Sea, 1972 (COLREGs)*. Accessed: 2025-05-13. 1972. URL: https://cil.nus.edu.sg/wp-content/uploads/2019/02/1972-Convention-on-Regulations-for-Preventing-Collisions-at-Sea.pdf.

[7] Aqsa; Mughees Abdullah; Khan Shehroz; Sudherbaabu Gaadha; Vain Jüri; Bennani Mohamed Taha; Truscan Dragos Ben Lahbib Hiba; Yaseen. *Two phase path planning for fuel-efficient safe navigation*. Accessed: 2025-05-12. 2025. URL: https://www.etis.ee/Portal/Publications/Display/82900e4a-2f6b-4b48-83ce-073c46864a1b.

[8] MindChip OÜ. *ABOUT - MindChip*. Accessed: 2025-05-08. URL: https://mindchip.ee/mesmerize/.

[9] Thomas Thuesen Enevoldsen. *Encounters described by COLREG rules 13-15*. Accessed: 2025-05-09. URL: https://www.researchgate.net/figure/Encounters-described-by-COLREG-rules-13-15-ie-overtaking-head-on-and-crossing_fig1_355872191.

[10] Pavel Gorbachenko. *What are Functional and Non-Functional Requirements and How to Document These*. Accessed: 2025-05-08. URL: https://enkonix.com/blog/functional-requirements-vs-non-functional/.

[11] Inc. Meta Platforms. *React – A JavaScript library for building user interfaces*. Accessed: 19.04.2025. URL: https://react.dev/.

[12] Vladimir Agafonkin and contributors. *Leaflet – an open-source JavaScript library for interactive maps*. Accessed: 19.04.2025. URL: https://leafletjs.com/.

[13] Paul Le Cam and contributors. *React-Leaflet – Integrating Leaflet with React*. Accessed: 19.04.2025. URL: https://react-leaflet.js.org/.

[14] X Corp. *Build fast, responsive sites with Bootstrap*. Accessed: 2025-05-09. URL: https://getbootstrap.com/.

[15] OpenJS Foundation and Jest contributors. *Jest*. Accessed: 2025-05-10. URL: https://jestjs.io/.

[16] OpenJS Foundation. *Node.js*. Accessed: 2025-05-10. URL: https://nodejs.org/en.

[17] Armin Ronacher and Pallets Projects. *Flask Documentation (Stable)*. Accessed: 19.04.2025. URL: https://flask.palletsprojects.com/en/stable/.

[18] Scikit-learn team. *scikit-learn Machine Learning in Python*. Accessed: 2025-05-09. URL: https://scikit-learn.org/stable/index.html.

[19] Joblib developers. *Joblib: running Python functions as pipeline jobs*. Accessed: 2025-05-10. URL: https://joblib.readthedocs.io/en/stable/.

[20] Travis Oliphant NumPy team. *NumPy The fundamental package for scientific computing with Python*. Accessed: 2025-05-10. URL: https://numpy.org/.

[21] Holger Krekel pytest-dev team. *pytest: helps you write better programs*. Accessed: 2025-05-10. URL: https://docs.pytest.org/en/stable/.

[22] GeeksforGeeks. *Random Forest Algorithm in Machine Learning*. Accessed: 2025-05-12. URL: https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/.

[23] GeeksforGeeks. *Multi-Layer Perceptron Learning in Tensorflow*. Accessed: 2025-05-12. URL: https://www.geeksforgeeks.org/multi-layer-perceptron-learning-in-tensorflow/.

[24] GitLab Inc. *GitLab Docs Get started with GitLab CI/CD*. Accessed: 2025-05-10. URL: https://docs.gitlab.com/ci/.

[25] Docker Inc. *Docker*. Accessed: 2025-05-12. URL: https://www.docker.com/.

[26] Igor Sysoev. *nginx*. Accessed: 2025-05-10. URL: https://nginx.org/en/.

[27] Benoit Chesneau Gunicorn Developers. *Gunicorn*. Accessed: 2025-05-10. URL: https://nginx.org/en/.

[28] Tom Graham Andre Goncalves. *Stop Designing for Only 85% Of Users: Nailing Accesibility In Design*. Accessed: 2025-05-12. URL: https://www.smashingmagazine.com/2017/10/nailing-accessibility-design/.

[29] National Oceanic and Atmospheric Administration. *What is the difference between a nautical mile and a knot?* Accessed: 2025-05-12. URL: https://oceanservice.noaa.gov/facts/nautical-mile-knot.html.

[30] Matt Way Nick Drewe. *A Simple to use javascript .GIF decoder.* Accessed: 2025-05-12. URL: https://www.npmjs.com/package/gifuct-js?activeTab=readme.

[31] Transpordiamet. *Nutimeri*. Accessed: 04.04.2025. URL: https://gis.transpordiamet.ee/nutimeri/.

[32] Dave Leaver. *Leaflet.markercluster*. Accessed: 2025-05-12. URL: https://leaflet.github.io/Leaflet.markercluster/.

Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

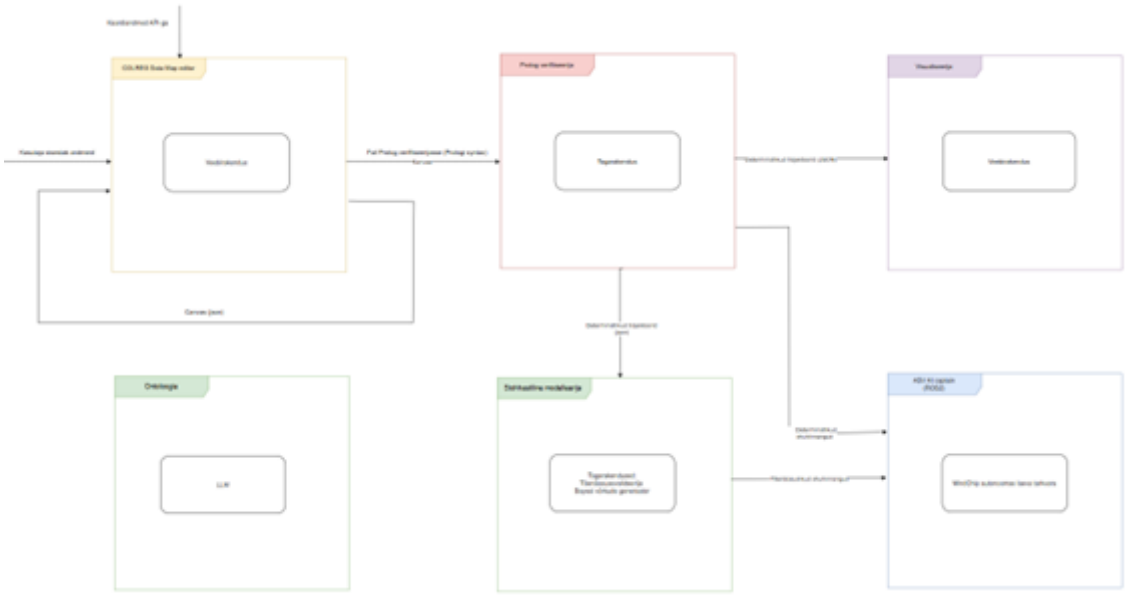
Mina, Jan Veližanin, Matthias Kaunimäe

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose "Robotlaevade navigatsioonisituatsioonide visualiseerimine simulaatoris", mille juhendaja on Gert Kanter and Hanna-Britt Reinpõld
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

06.06.2025

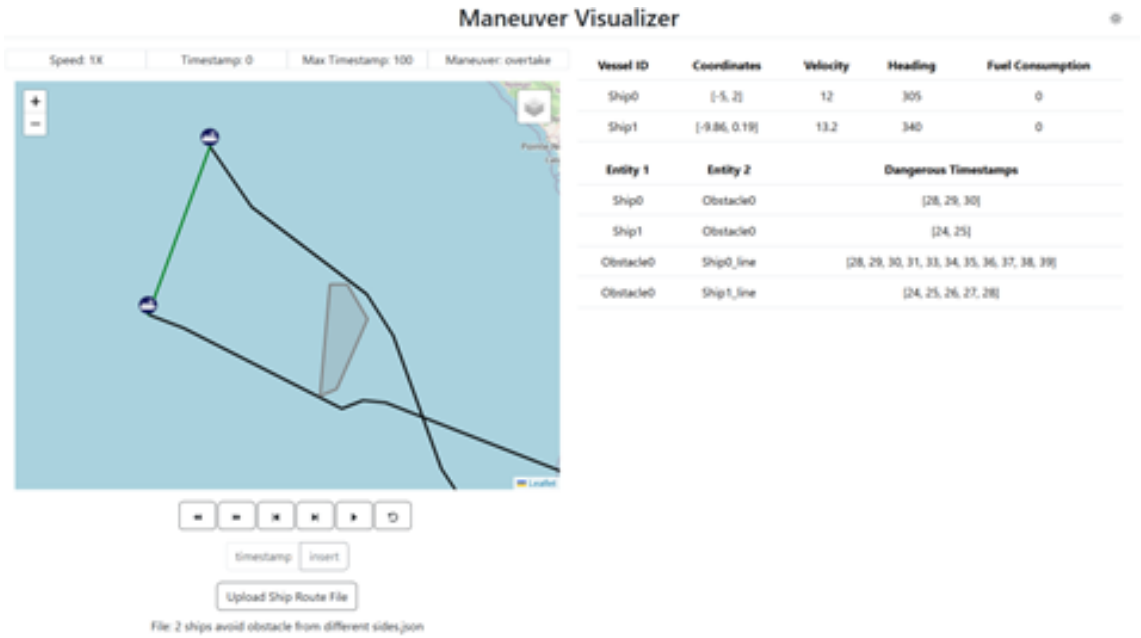
[1] Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

Lisa 2 - Mereohutussüsteemi arhitektuur täispilt

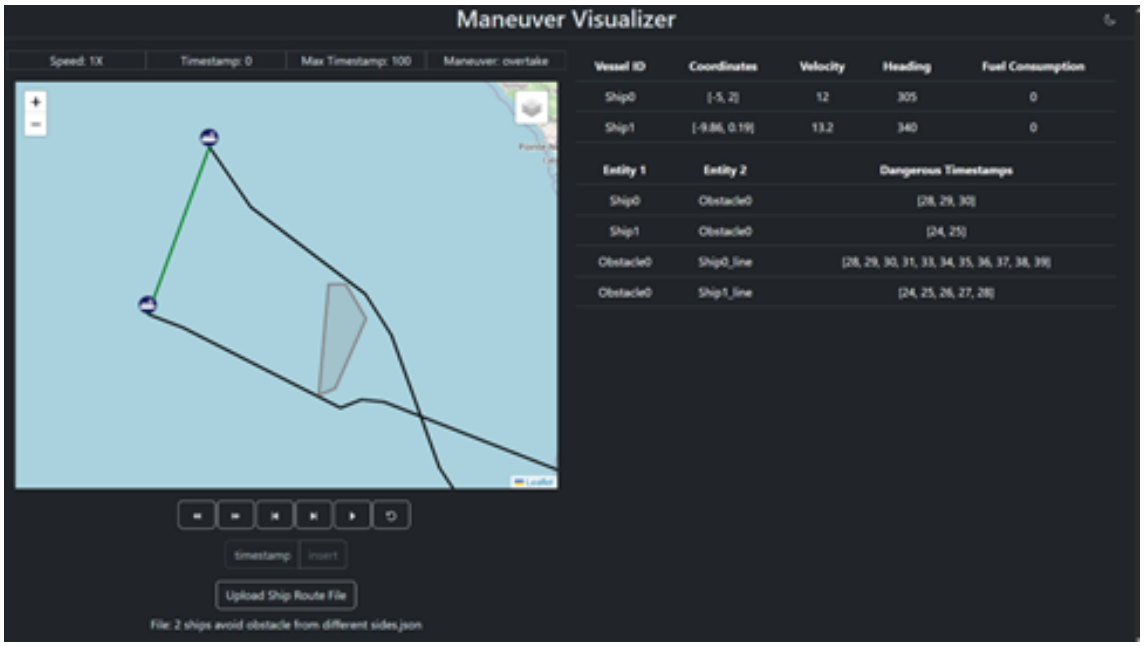


Joonis 14. Mereohutussüsteemi projekti arhitektuur

Lisa 3 – Visualiseerija öö- ja päevarežiim

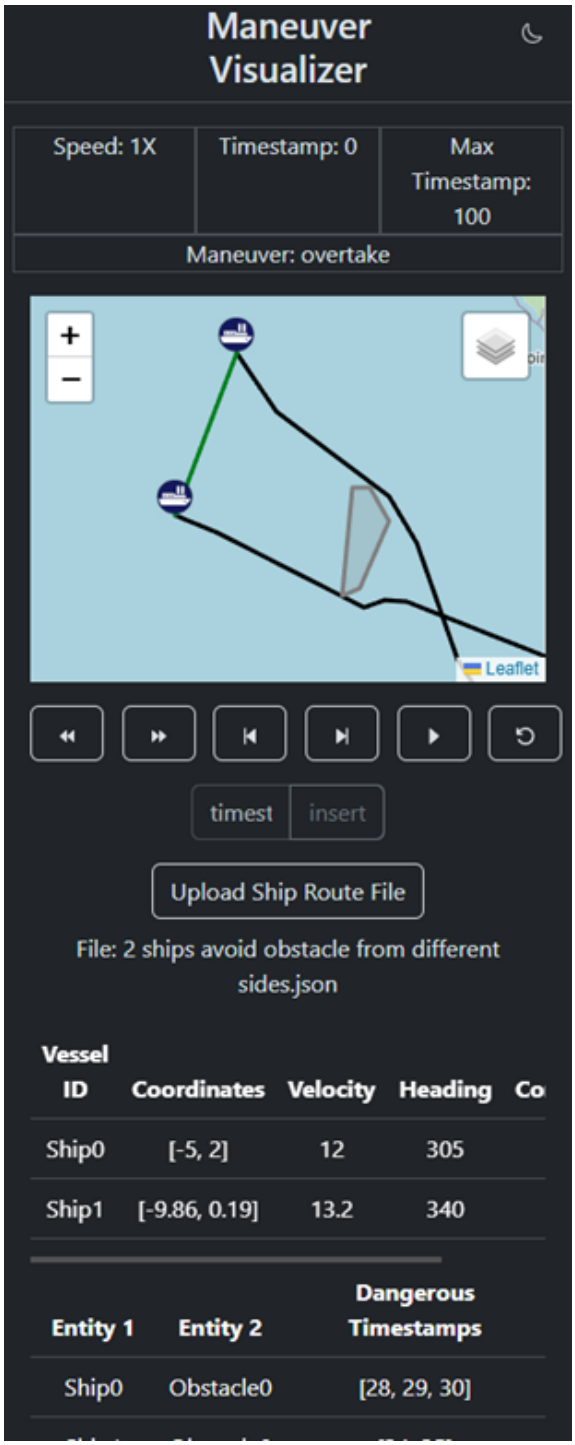


Joonis 15. Visualiseerija kasutajaliides päevarežiimis



Joonis 16. Visualiseerija kasutajaliides öörežiimis

Lisa 4 – Visualiseerija telefonivaade



17. Visualiseerija telefonivaade