Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Automatizálási és Alkalmazott Informatikai Tanszék

Szimulációs keretrendszer vitorlás hajók optimális viselkedésének és navigációjának vizsgálatára

Készítette
Jánoky László Viktor

Konzulens
Dr. Ekler Péter

Tartalomjegyzék

Összefoglaló

A dolgozat célja egy olyan szimulációs keretrendszer elkészítése, mely lehetővé teszi vitorlás hajók viselkedésének vizsgálatát megadott körülmények között, különböző célfüggvények mellett. A vitorlázásban nagyon sok gyakorlat megfigyeléseken, tapasztalatokon alapul, a rendszer segítségével ezeket szeretném többek között alátámasztani vagy javítani. Az így futtatott szimulációk választ adnak például olyan komplex kérdésekre, mint amiket egy hajó útvonalválasztása vet fel.

Egy hajó navigációja során nem elég a távolságot és földrajzi adottságokat figyelembe venni, a szél előrejelzés, időjárás, áramlatok, más hajók és tereptárgyak is mind fontos tényezők, nem is beszélve az egyes manőverek költségeiről. Ha ehhez még hozzátesszük az alternatív meghajtási móddal rendelkező hajók szempontjait is, kifejezetten komplex problémát kapunk.

Célom, hogy létrehozzak egy keretrendszert ami lehetővé teszi a fentebb említett hatások figyelembevételével való vizsgálatát a megadott stratégiáknak, viselkedéseknek. A rendszer lehetővé teszi, hogy egyedi viselkedést definiáljunk a vizsgálati alany hajóknak, illetve figyeljük ennek eredményességét. Végül a szimuláció és valamilyen optimalizálási módszer segítségével megtalálhatjuk az adott helyzetre a jó válaszokat.

A téma aktualitását egyrészt a kedvtelési célú vitorlás hajózás népszerűségének növekedése, másrészt a kereskedelmi hajókon megjelenő kisegítő sárkányvitorlák adják. Egy ilyen kisegítő vitorla pár százalékos üzemanyag megtakarítást nyújt, ami a globális teherszállítás piacán hatalmas megtakarításnak számít.

A rendszer segítségével számos hasznos következtetést vonhatunk le a vizsgált kérdésekben és várhatóan később akár hasonló problémakörre is alkalmazható lesz a módszer, például a légi közlekedés vizsgálatára.

Abstract

The goal of this work is to create a simulation framework to be used in simulating a sailboat's behavior with given parameters and goals. In sailing many practices are based on observations or previous experiences, with the use of the system I hope to back these up, or improve them. These simulations can give us answers for complex problems, like finding the best course for a sailboat.

When calculating a route for a sailboat, one must consider the distance, geography, wind, currents, other ships and obstacles, not mentioning the total costs of maneuvers. When this is done for hybrid propulsion ships the problem becomes increasingly hard.

My goal is to create a framework for observing behaviors and strategies subjected to the aforementioned effects. The system should let us define and observe a unique behavior to the subject boat used in the simulation. With the simulation and a chosen optimization technique we can find the proper answers for the given situations.

The actuality of the subject comes from the emerging popularity of recreational sailing and the appearance of auxiliary kite sails on large commercial vessels. These kite sails provide a few percentage of fuel saving, which on the scale of global freight traffic has a huge impact.

I hope to make many useful observations in the examined questions by using the system. I also like to think that the method could be used in other domains, like air traffic.

1. Bevezetés

1.1 A dolgozat célja

Dolgozatom célja egy olyan szimulációs eszköz elkészítése, amellyel szimulálni lehet egy vitorlás hajó környezetét és viselkedését, így vizsgálva különböző stratégiákat és hatásokat. Az eszköz segítségével összehasonlíthatóvá vállnak a különböző, eddig empirikus alapú stratégiák és legjobb gyakorlatok a hajózás különböző problémái terén.

Olyan komplex problémák megoldásához igyekszem eszközöket nyújtani, mint egy hajó optimális útvonalválasztása, versenystratégiák meghatározása, vagy hajó-hajó interakciók vizsgálata.

A program segítségével konkrét algoritmusokat és stratégiákat vizsgálhatok meg egy előre definiált cél függvényében, értékelve és finomítva őket. Reményeim szerint az így előálló stratégiák, később valós körülmények között is felhasználásra kerülhetnek. Például egy adott hajó paramétereire optimalizált útkereső algoritmus később felhasználható lehet egy navigációs szoftverben.

Az eszköz fő funkciója tehát egy vitorlás hajó környezetének és rá ható fizikai hatásoknak a szimulálása, úgy hogy a hajó viselkedését egy a felhasználó által definiált stratégia írja le. A másodlagos funkció pedig az előzőek függvényében ennek a stratégiának a vizsgálata és finomítása.

1.2 A téma aktualitása

A téma aktualitását egyrészt az egyedisége adja, több hasonló szimulációs szoftver létezik más problémakörökben, de a vitorlázáshoz még nem találni ilyet. Például a közúti forgalom szimulálására több eszköz is a rendelkezésünkre áll, mind dobozos termék formájában (például a TransModeler [1]), mind pedig open-source projektként (például SUMO – Simulation of Urban MObility [2]).

A vitorlás hajózás terén azonban olyan eszköz amelynek az elsődleges célja a korábban megfogalmazott vizsgálat és finomítás még nem létezik. A legtöbb vitorlás szimulátor jelenleg leginkább szórakoztató (Sail Simulator [3], SailX [4]), vagy oktatási célú (a NauticEd [5]), mint a fizikai interfésszel is rendelkező VS1 [6].

A probléma aktualitása mellett szól továbbá a vitorlázás mint sport népszerűségének növekedése is, a három évente megrendezésre kerülő, legnevesebb földkerülő vitorlás verseny (a Volvo Ocean Race) televíziós közvetítettsége például a 2011-12-es évadban az előzőhöz képest 42%-al nőtt [7]. Emellett a hazai példákról sem szabad elfeledkeznünk, Európa legnagyobb tókerülő versenye, az évente megrendezésre kerülő Kékszalag is évről évre egyre több versenyzőt vonz, a használt technikák látványos fejlődése mellett. A 2014 és 2015-ös versenyen már a vitorlázás technológiájának élvonalába tartozó GC32 típusú hydrofoil katamarán is részt vett.

A sport népszerűségén kívül még egy további fontos új formája is terjed a vitorlázásnak, a kereskedelmi hajózásban megjelenő kisegítő sárkányvitorlák. Egy ilyen vitorlával, egy út alkalmával akár 10-35% üzemanyag megtakarítható (a gyártó SkySails [8] szerint), de már csak pár százalékos megtakarítás esetén is, globális skálán nézve óriási pénzügyi hatása lehet egy ilyen technológiának.

2 2. A probléma bemutatása

2.1 Szakterületi alapfogalmak

A magyar nyelvben nagyon sok idegen eredetű szó van a vitorlázás fogalmainak leírására. Sok esetben ezeknek nincs magyar megfelelőjük, vagy nagyon nehézkesek. A legtöbb vitorlás szakszó német, holland, esetleg angol eredetű. Dolgozatomnak nem célja a vitorlázás ismertetése, ám az egyes alapfogalmak ismerete elengedhetetlen az elkészült program funkcióinak megértéséhez.

Azok az olvasók akik rendelkeznek vitorlás alapismeretekkel, ezt a szekciót nyugodtan átugorhatják, a többiek számára pedig igyekeztem ahol lehet a legérthetőbb, magyar nevét használni az egyes fogalmaknak.

2.1.1 Mértékegységek

A hajózás világában a mai napig nem széles körben elterjedt az SI mértékegységek használata, leginkább történelmi okok miatt. Az átállást nehezíti, hogy a legtöbb adatforrás (térképek, meteorológia, hajók adatai) gyakran a régi, tradicionális mértékegységekben van megadva. Mivel munkám során számos ilyen adatforrást felhasználtam, valamint több képletben is előfordulnak ilyen mértékegységek, ezért álljon itt egy lista a fontosabbakról.

Fizikai jellemző	Tradicionális mértékegység	Jelölés	SI mértékegység	Váltószám
Hajó sebessége	csomó	kts	méter / szekundum	1 kts = 0.5144 m/s
Hajó dimenziói	láb	ft	méter	1 ft = 0.3048 m
Szél erősség	beaufort	bft	nincs (skála)	-
Szél sebesség	csomó	kts	méter / szekundum	1 kts = 0.5144 m/s

Táblázat 1. Hajózási mértékegységek

2.1.2 Fogalmak

A következőkben a legfontosabb fogalmak kerülnek bemutatásra.

Relatív szélirány:

A szél irányvektora és a hajó hossztengelye között bezárt szögek közül a kisebbik.

Valódi szél:

A mozdulatlan megfigyelő által tapasztalt szél.

Látszólagos szél:

A mozgásban lévő hajó által tapasztalt szél, a valódi széltől annyiban tér el, hogy a hajó mozgásából származó menetszél hozzáadódik.

Fordulás (Tackolás):

A hajó szél felé történő csapásváltása, azaz a hajó úgy vált irányt, hogy a forduló közepéig a széllel egyre hegyesebb szöget zár be (élesedik), majd az átfordulás pillanatában a szél pont szemből (0˚-on) fúj

Perdülés (Halzolás):

A hajó széltől elfele történő csapásváltása, azaz a hajó úgy vált irányt, hogy a művelet közepéig a széllel egyre tompább szöget zár be, majd a váltás pillanatában a szél pont hátulról (180˚-ról) fúj.

2.1.3 Vitorlás hajó és részei

A következőkben a legfontosabb hajó részek kerülnek ismertetésre, melyekre a dolgozat hivatkozik.

2.1.3.1 Hajótest

Egy vitorlás hajó egyik legfontosabb tulajdonsága a hajótest formája, típusa és darabszáma. Ezen paraméterek számos kombinációja létezik, ám ezek ismertetése nem férne bele a dolgozat terjedelmébe, így ettől eltekintve innentől kezdve csak az egytestű, úszó vagy sikló hajótesteket vizsgáljuk.

Úszó hajótest:

A tradicionális hajótest, amely a vízbe merítve képez felhajtó erőt. Az úszó hajó mozgása közben keletkező hullámok komplex interakciója miatt egy empirikus felső határ adható a hajó által normál körülmények között elérhető maximális sebességre. Ezt az alábbi formula írja le, ahol v_{max} a maximális sebesség csomóban, L_{wl} a vízvonal hossza lábban.

$$v_{max} \approx 1.34 \times \sqrt{L_{wl}}$$

Sikló hajótest:

Az úszó hajótest korlátjaink megkerülésére fejlődtek ki a sikló hajótestek, ezt a modern anyagtudomány, leginkább a kompozitok megjelenése tette lehetővé. A sikló hajót nem lassítja a saját maga által keltett hullám, hanem egy bizonyos sebesség elérése után „felül rá", mint egy szörfdeszka.

2.1.3.2 Kiegyensúlyozás

Ha egy vitorlás hajót nem hossztengelyben ér a szél, akkor valamelyik oldalirányba erőt fog kifejteni, ennek ellensúlyozására több módszer létezik, ezek közül párat felületesen ismertetve.

Többtestű hajók:

Egy többtestű hajót a testek közötti távolságból adódó erőkar és a testre ható nehézségi erő fog kiegyensúlyozni.

Ballasztos hajók:

Egy egytestű hajót általában valamilyen ballaszt tart egyensúlyban, ez lehet a hajótestben, vagy a hajótesten kívül. A hajótesten kívüli ballaszt ugyanakkora erőt kisebb súllyal tud kifejteni a nagyobb erőkar miatt, ezáltal a hajó teljes tömege kisebb maradhat azonos visszabillentő erő mellet (ugyanakkor a hajótest szerkezetének komplexitása nő). A legtöbb Bermuda rig-es vitorlás hajó külső ballasztos (vagy más néven tőkesúlyos).

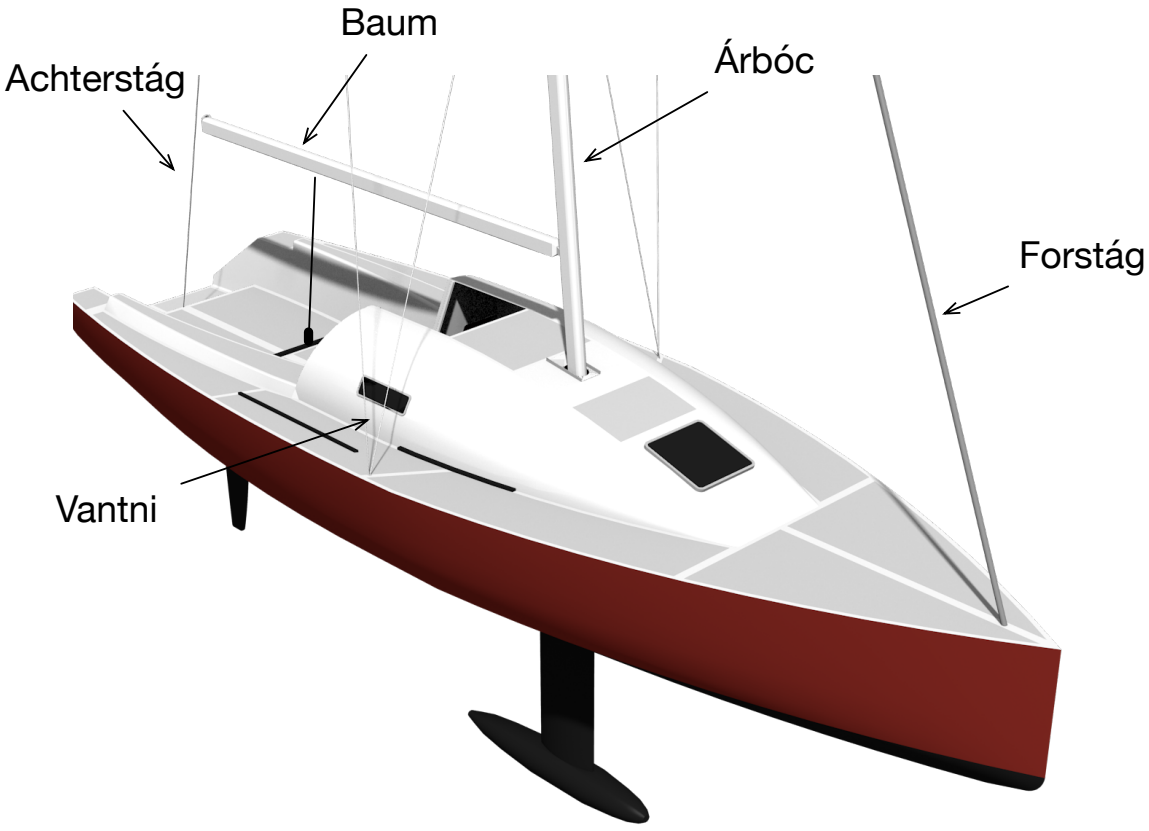
Ballaszt nélküli hajók:

Ha a fellépő oldalirányú erők nem jelentősek, vagy a hajó stabilitása csak másodlagos szempont gyakran elhagyható a ballaszt. Ilyenkor vagy csak a hajótest saját súlya képezi a visszabillentő erőt, vagy az valamilyen más módon képződik, például a legénység koordinált mozgásával.

2.1.3.3 Vitorlázat, rudazat, kötélzet

A másik kulcsjellemzője egy vitorlás hajónak a vitorlázat, rudazat és kötélzet összessége (angolul rig). Ez szabja meg, hogy a hajó milyen vitorlákat tud vinni, milyen szél irányokra tud közlekedni, milyen hatékonysággal. Vitorlázat típusból megszámlálhatatlan létezik, mind-mind külön altípusokkal, ezek bemutatása szintén nem témája a dolgozatnak. Jelen esetben csak a modern vitorlázásban legelterjedtebb Bermuda rig-et fogjuk vizsgálni, de később látni fogjuk, hogy a rendszer könnyedén alkalmazható más típusú vitorlázatokra is.

A Bermuda rig egy rendkívül sokoldalú vitorlázat, segítségével a szélhez képest akár 15˚-fokig is lehet vitorlázni (azaz a hajó és a szél által bezárt szög 15˚ vagy nagyobb).



Ábra 1. Bermuda rig rudazat

Árbóc:

A hajó hossztengelyén elhelyezkedő, függőleges merevített pózna ami a vitorlák bekötési pontjait hivatott tartani.

Baum:

Az árbóc alsó részéhez elforgatható módon rögzített, a hajó hátrafele irányúló vízszintes merevítő rúd.

Forstág:

Az árbóc első merevítője (általában drótkötél, opcionálisan tekerhető alumínium sínnel bevonva az orrvitorla feltekeréséhez).

Vantni:

Az árbóc oldalsó merevítői, általában acélsodrony, állítható feszességgel.

Achterstág:

Az árbóc hátsó merevítője, opcionális, gyakran az enyhén árbóc mögé helyezett vantink helyettesítik.

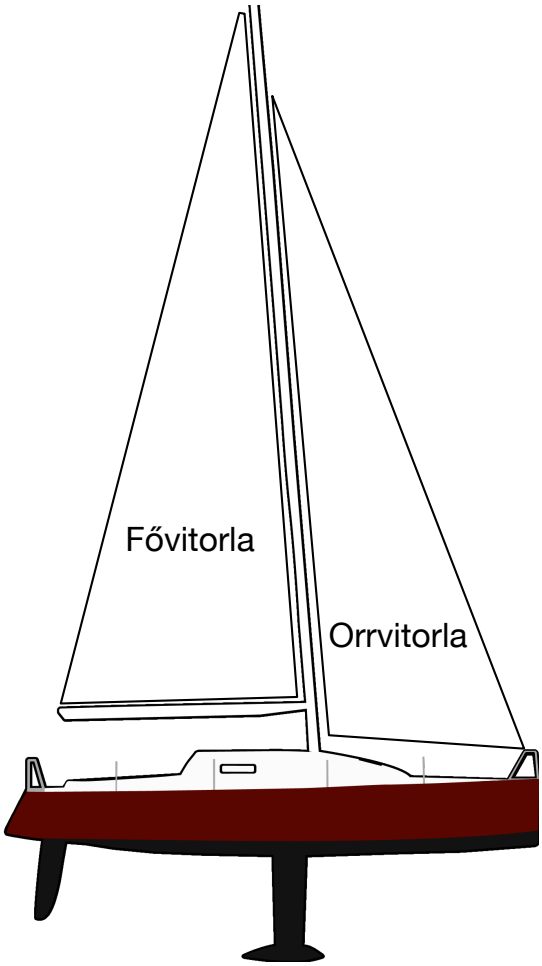
Alapvitorlák:

A Bermuda rig alap vitorlázata két vitorlából áll: egy orrvitorlából (mérete szerint lehet fock/jib vagy genoa) és egy fővitorlából (nagyvitorla/grósz). A háromszög alakú orrvitorla első éle gyakran egyúttal az árbóc első merevítője is (forstág), a vitorla másik oldala az éllel szemközti csúcsnál van általában két kötéllel rögzítve a hajótesten lévő csigákhoz (amik gyakran sínen mozgathatók).

A fővitorla hajó orra felőli élének alja általában fixen rögzítésre kerül a baum és árbóc találkozásánál, teteje az árbóc csúcsánál, alsó élének hátsó sarka állítható módon a baum végén.

Alapvitorlák típusai:

- Orrvitorla
    - Fock
    - Genoa
- Nagyvitorla



Ábra 2. Bermuda rig alapvitorlázat

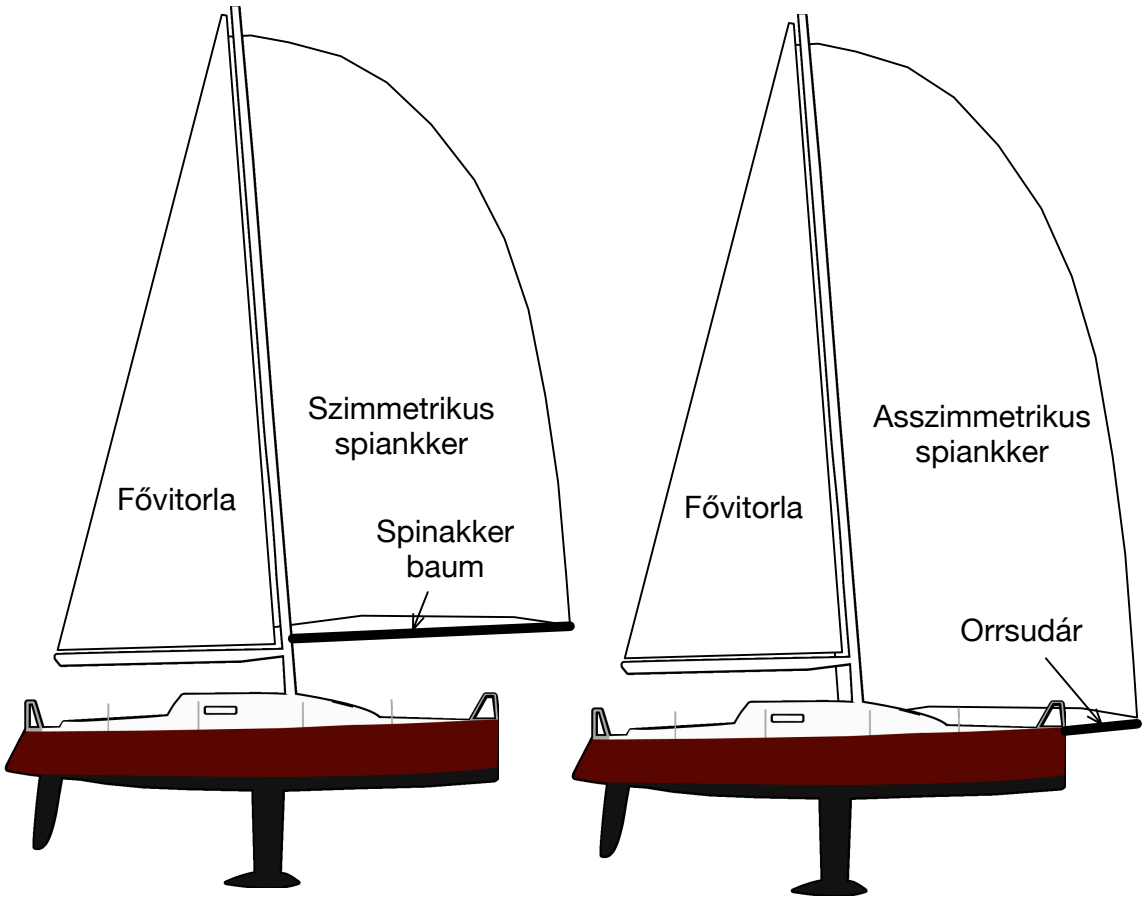
Az alapvitorlázat önmagában általában a 15˚-től 90˚-ig terjedő tartományban használatos, itt nyújtja a legjobb teljesítményt. Az ettől eltérő relatív szélirányok esetén valamilyen kiegészítő vagy bőszeles vitorlát érdemes használni.

Bőszeles vitorlák:

A 90˚-nál nagyobb szögben érkező szél legjobb kihasználásának érdekében úgynevezett bőszeles, vagy hátszél vitorlákkal szokás kiegészíteni az alapvitorlázatot. Ezeknek több típusa létezik, aszerint hogy milyen irányú és erejű szélre vannak optimalizálva. Általában könnyű nylon anyagból készülnek és az alapvitorlázatnál jóval nagyobb méretűek. Közös tulajdonságuk, hogy három ponton vannak rögzítve, ami közül az egyik mindig az árbóc.

A szimmetrikus változatot spinakkernek nevezik, ez az árbócon túl két kötéllel a hajó két oldalához van rögzítve, és egy spinakker baum nevű mozgatható távtartóval stabilizálva a szél felőli oldalon.

Az aszimmetrikus változatot aszimmetrikus spinakkernek, genakkernek vagy reachernek nevezik, attól függően hogy milyen irányú szélre van optimalizálva. A szimmetrikus változattal ellentétben ezt nem tartja el spinakker baum, ehelyett az első csúcsa a hajó elejére, vagy egy (akár kitolható) orrsudárra van rögzítve.



Ábra 3. Szimmetrikus és asszimetrikus spinakker

Típusai:

- Szimmetrikus spinakker
- Aszimmetrikus spinakker (genakker)

A bőszeles vitorlákkal történő műveletek általában a legbonyolultabbak és leglassabbak közé tartoznak egy hajón (a méretek és fellépő erők miatt), ezt az optimális stratégiák keresésénél mindenképpen figyelembe kell venni.

Kiegészítő vitorlák:

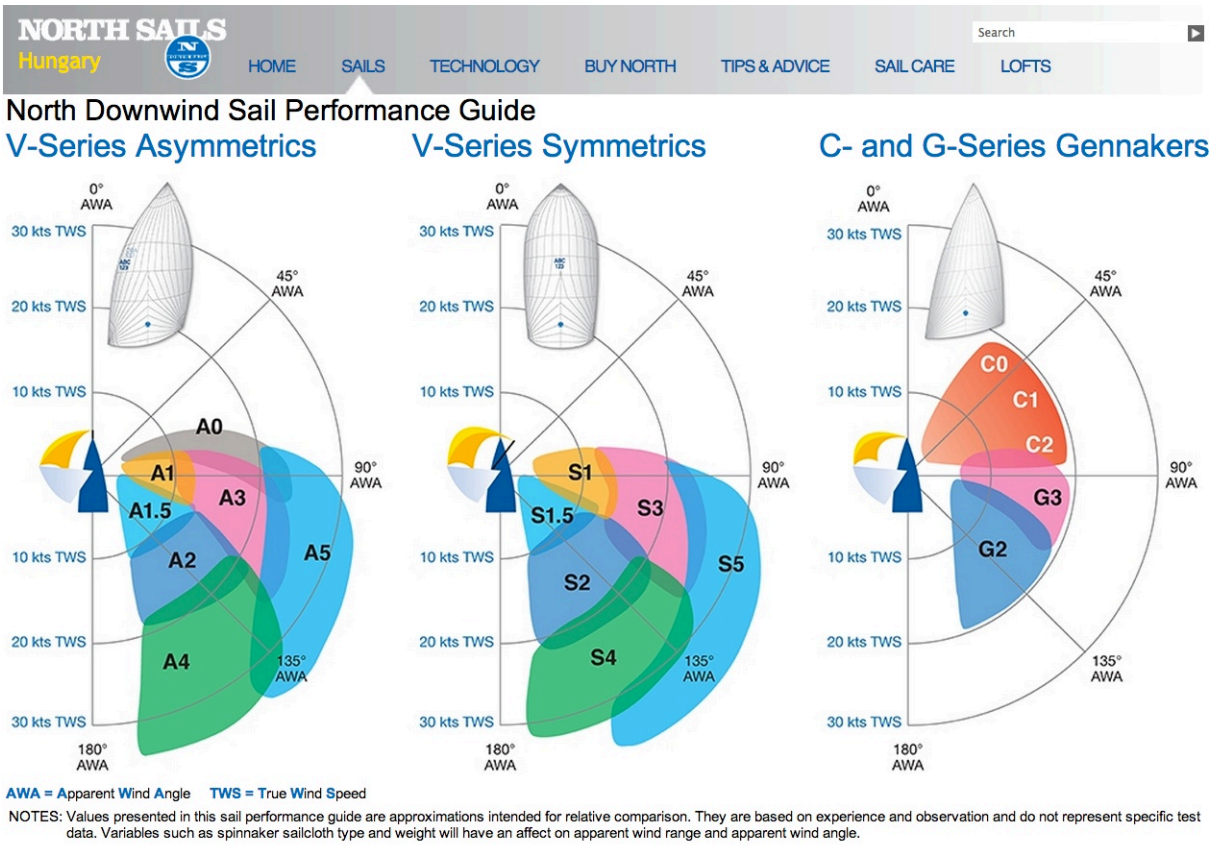
Az eddig felsoroltakon kívül léteznek még több, speciális helyzetre kitalált vitorlák is, ezek általában valamelyik korábban már említett vitorla speciális változatai. Például a Code 0 ami egy kis szeles átmenet egy genoa és egy asszimetrikus spinakker között, vagy a vihar fock ami egy nagyon kis méretű fock, a viharos időjárásban való közlekedéshez.

2.1.4 A vitorlázás fizikája

Ezen jellemzők összessége szabja meg, hogy egy vitorlás hajó az adott viszonyok között hogyan viselkedik. További példáinkban a legelterjedtebb kombinációt, egy egytestű, tőkesúlyos, Bermuda riges hajót fogunk vizsgálni.

Polár diagram:

A gyártók egy hajó vagy vitorla viselkedését az adott szélviszonyok között egy polár diagrammon szokták megadni. Erre láthatunk példát alább:



Ábra 4. Polár diagram példa (forrás: [9])

Egy ilyen diagrammról leolvashatjuk, hogy adott látszólagos szélirány (AWA – Apparent Wind Angle), valódi szélsebesség (TWS – True Wind Speed) függvényében melyik típusú vitorla optimális, mekkora a hajó által elérhető maximális sebesség, vagy mekkora az optimális dőlés (attól függően hogy miről szól a diagram).

A szimuláció szempontjából leginkább az elérhető maximális sebességet ábrázoló polár diagramm adatai érdekesek számunkra.

2.2 2.2 Kiemelt problémák

Leggyorsabb útvonal:

A korábban ismertetett fizikai jelenségekből és jellemzőkből következik, hogy a vitorlázásban, a más területeken triviálisnak számító kérdések komoly problémaként merülnek fel. Például a leggyorsabb útvonal megtalálása két pont között az alábbiaktól függhet:

- A hajó maximális sebessége az adott szélirány és szélerősség függvényében (polár diagramról leolvasható)
- Az útvonal bizonyos pontjain mért szélerősség és szélirány változásai
- Hullámzás, áramlatok
- Vízmélység, meder viszonyok

Ha csak az első szempontot figyeljük, akkor is előfordulhat, hogy egy hajó teljesítménygörbéjéhez több, egyenértékű leggyorsabb út tartozik.

Erre a problémára nehéz konkrét, numerikus választ adni. Ha lenne egy olyan algoritmusunk ami az összes adat ismeretében megmondja melyek az optimális útvonalak, annak a használhatósága is kérdéses, hisz az összes paramétert nagyon ritkán ismerjük.

Sokkal célravezetőbb lenne egy olyan stratégiával előállni, ami paraméterek viszonylag széles skáláján képes, közel optimális útvonalkeresést biztosítani, így küszöbölve ki az ismeretlen tényezők okozta zavarokat, de feláldozva a mindig optimális megoldás megtalálását. Reményeim szerint a program segítségével találhatok egy ezeknek a kritériumoknak minél inkább megfelelő stratégiát.

Adott idejű útkeresés:

Másik nem triviális probléma egy adott idejű út megtalálása. Erre gyakran szükség lehet, például ha egy pontra csak egy adott időben lehet elérni (zsilipek, kikötők), vagy valamilyen ütemterv szerint akarunk haladni. Ekkor valamilyen másodlagos paraméterre kell optimalizálni, például a megtett út „kényelmességére” (milyen szögből érkeznek a hullámok, mennyire dől a hajó, stb.).

Ez a probléma szintén érdekes lehet a korábban már említett, teherhajókra szánt kiegészítő vitorla esetén is. Az egyik legfontosabb vívmánya a modern óceánjáró hajózásnak a kiszámíthatóság, ezen valamelyest ront a kisegítő vitorla (ez az egyik fő gátja az elterjedésének), ennek kompenzálására ha létezne valamilyen stratégia az messzemenő hatásokkal járna.

3 3. A szimulációs keretrendszer

3.1 A modell

A szimuláció diszkrét-idejű, diszkrét-terű, több-ágenses modellen alapul [10], melynek elkészítése során igyekeztem a lehető legtöbb paramétert konfigurálhatóra készíteni, ezáltal szélesítve a program felhasználhatóságának körét. A tervezés során az egyik legnagyobb kihívás a túlzott absztrakció és a modellezett világ pontossága közötti egyensúly megtalálása volt. Ebben elsősorban a saját vitorlás tapasztalataimra támaszkodtam, a tapasztalatok szerint döntően befolyásoló tényezőket igyekeztem fokozottan figyelembe venni.

3.1.1 Idő

A szimuláció diszkrét idő alapú, konfigurálható időlépésekkel, kezdő és végponttal. Alap beállítások szerint egy időlépés egy másodperc, hosszú távú szimuláció esetén ezt azonban érdemes nagyobbra választani a hosszú futásidő és nagy méretű eredmény fileok elkerülése érdekében.

Értelemszerűen minél kisebb időlépést választunk annál pontosabb lesz a szimuláció, a legkisebb választható lépésköz *1 ms*, azonban ekkor már nem lényegesen pontosabb a szimuláció, mint például *10 ms* vagy *100 ms*-nál.

3.1.1 Világ

A szimulált világ véges, 2 dimenziós, négyzetrács által felosztott. Ez a felosztás ugyanakkor nem jelenti azt hogy az egyes objektumok (hajók, tereptárgyak) cellához lennének kötve. A négyzetrács célja elsősorban a szél és víz modellezésében segít, egy cellán belül a szélirány és szélerősség állandó, ugyanakkor például egy vitorláshajó egyszerre több cellát is elfoglalhat, mindegyik hatását figyelembe véve.

A világ a szimulációs idő múlásával diszkrét állapotok sorozatában van, minden következő állapot az előző állapotokból és az ágensek cselekvéseiből számolható.

A kezdeti értékek megadhatók cella szinten, így lehet például időjárási adatokat betölteni egy külső forrásból (a rendszernek nem célja az időjárás modellezése), de az egyszerűbb mintákat tudja generálni is a rendszer (egyenletes, véletlenszerű vagy gradiens szél).

Mértékegységek:

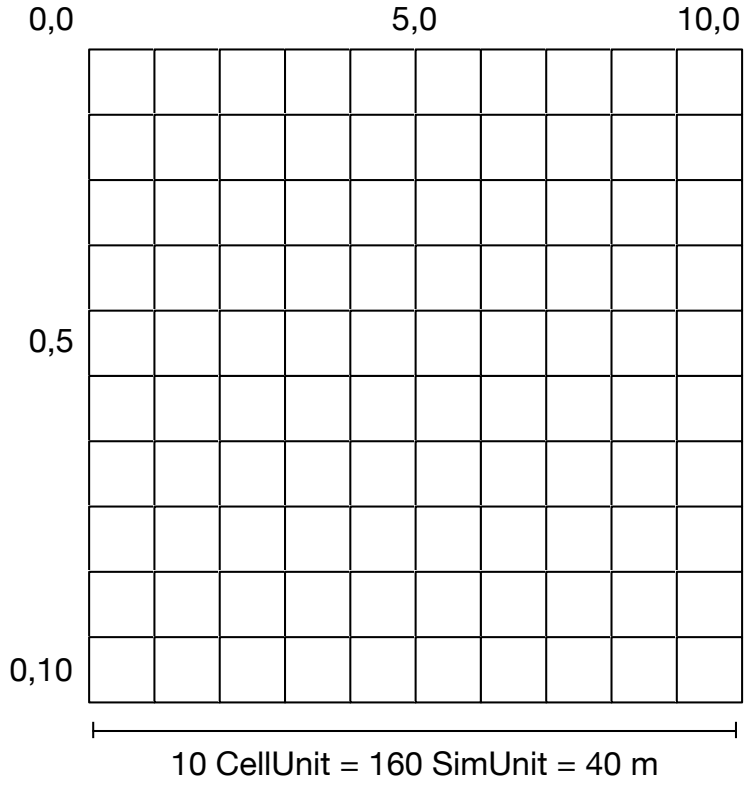
A szimulációban a távolság mértékegysége a *SimUnit*, azáltal hogy ennek az aránya a méterhez konfigurálható, tetszőleges felbontású és méretű szimulációt készíthetünk (alapból *1 SimUnit = 0.25 m*).

A négyzetrács könnyebb kezelhetőségének érdekében bevezettem a *CellUnit* egységet is, azaz egy cella szélességét, egy ilyen egység alap beállítások szerint 16 *SimUnit*, azaz 4 méter.

A skálázás szempontjából invariáns mértékegységek belső előfordulásainál, mint a szélsebesség vagy egy hajó sebessége igyekeztem az SI-beli mértékegységeket használni. Ugyanakkor a felületen inkább a tradicionális hajózási mértékegységeket jelenítettem meg, de ez terveim szerint később konfigurálható lesz.

Koordináta rendszer:

A világban a *(0,0)* koordináta a bal felső sarkot jelenti, a négyzetrács egy cellájának koordinátája szintén a bal felső sarkát adja meg, a korábban említett *CellUnit*-ban.



Ábra 5. A világ koordináta rendszere

Egy cella az alábbi értékeket tartalmazza:

- Koordináták *(x, y)*, a világon belüli koordinátái ennek a cellának (cella egységben kifejezve)
- Szélirány a *[0˚, 360˚]* tartományon belül, azt jelenti hogy honnan fúj a szél, ahol a *0˚* pont északról, a *90˚* keletről fújó szelet jelent.
- Szélerősség *m/s*-ban mérve
- Vízmélység m-ben megadva
- A cellára ható lokális hatások
    - Például egy vitorlás hajó „kitakarja” a szelet, ezáltal mögötte gyengébb erősséggel és megváltozott iránnyal fog fújni

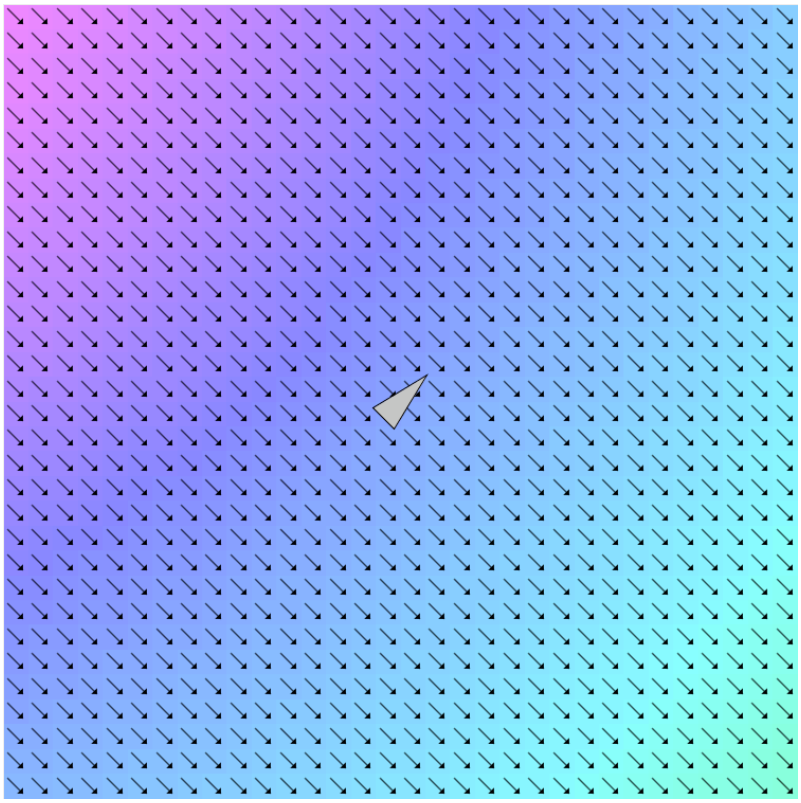
Időjárás:

Az időjárási adatok származhatnak külső forrásból vagy a már említett formákból amiket generálni tud a rendszer, ezek az alábbiak:

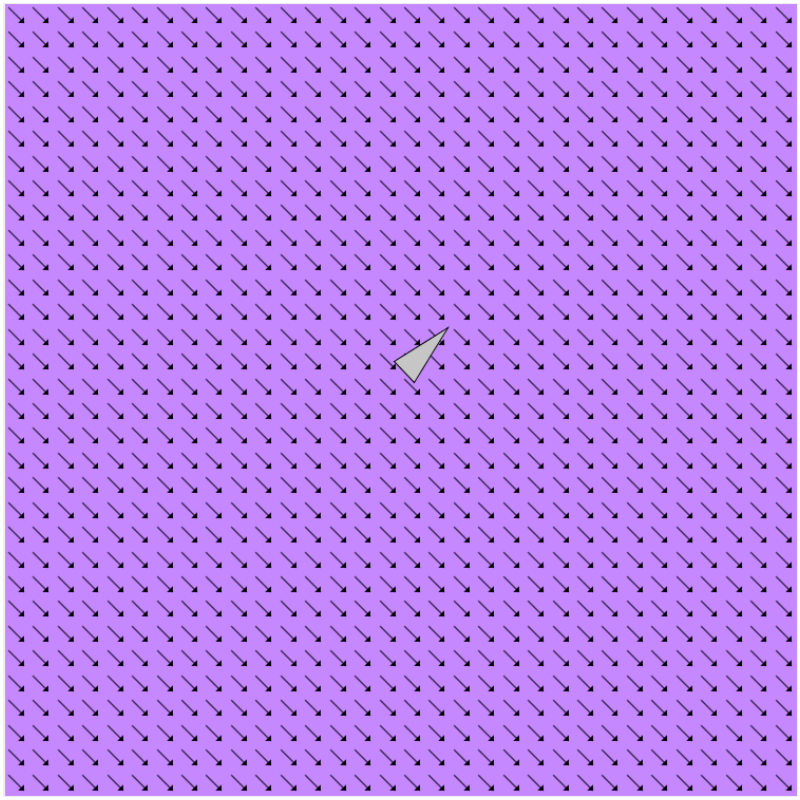
- Egyenletes szél, minden cellában azonos irányból, azonos sebességgel fúj
- Gradiens szél, egy pontban a legerősebb és az innen mért távolsággal csökken
- Véletlenszerű szél, minden cellában véletlenszerű erővel és iránnyal

A generált szelek a valóságban nagyon ritkán fordulnak elő ilyen formában. Mindazonáltal nagyon jól használható eszközt biztosítanak egy-egy stratégia kontrollált körülmények közötti összehasonlításához.

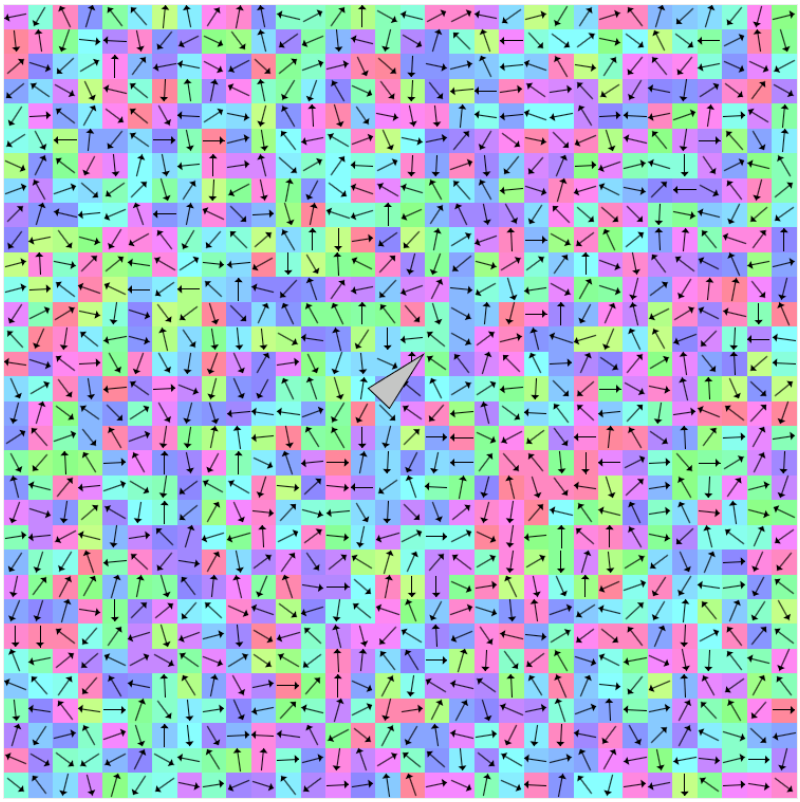
A megjelenítés során a szél irányát a nyilak jelölik, erősségét a színek. A HSB (Hue Saturation Brightness) színskálának a *Hue* értéke áll egyenes arányban a szél erősségével, azaz durva becsléssel minél nagyobb erejű a szél annál kisebb hullámhosszú (és nagyobb frekvenciájú) szín tartozik hozzá.



Ábra 6. A szimulált világ grádiens szél esetén



Ábra 7. Egyenletes szél



Ábra 8. Véletlenszerű szél

Szereplők:

A világban egy fajta szereplő vagy ágens van jelenleg (de továbbiak hozzáadásának lehetőségével), a vitorlás hajók. Egy vitorlás hajó rendelkezik statikus jellemzőkkel (hosszúság, polár görbéi, lehetséges vitorlái, stb.), dinamikusan változó állapottal (pozíció, sebesség), a lehetséges műveletek listájával (fordulás, vitorla váltás) és a viselkedését leíró stratégiával.

- Statikus jellemzők
  - Fizikai jellemzők: hosszúság, szélesség, magasság, tömeg
  - Vitorlázat és a hozzátartozó teljesítmény adatok
- Dinamikus jellemzők
  - Pozíció (*SimUnit*-ban megadva)
  - Haladási irány (*0˚-360˚*)
  - Sebesség (m/s)
  - Éppen használt vitorla

- Műveletek
    - Lehetséges műveletek
    - Folyamatban lévő műveletek
- Stratégia
    - Célok

3.1.2 A szimuláció lefolyása

A szimuláció eredménye a világ diszkrét időpontokban felvett állapotainak összessége. Egy-egy ilyen időpontban az állapot kiszámítása így történik:

1. Fizikai szimulációs lépés:

Az előző állapot alapján a világ fizikai állapotának szimulálása, ez magában foglalja:

- Az időjárási adatok frissítését (ha szükséges)
- Az egyes hajók (ágensek) új állapotainak kiszámítását az előző állapot és akciók alapján
- A világ új állapotának kiszámítását a hajók által okozott hatások függvényében

2. Logikai szimulációs lépés:

Az új állapotot átadjuk az egyes szereplők stratégiájának amik eldöntik az alkalmazandó akciókat.

- Például, ha a szél változott, a hajó dönthet úgy hogy követi, vagy elfordul.
- Egy akció általában egy lépésig tart, például ha a hajó elfordul, és a fordulási sebessége 30˚ / másodperc, akkor egy fél másodperces lépésközben az akció egy 15˚-os fordulás lesz.
- Ha az akciónak állapotot kell átvinnie (például a vitorla csere művelete hol tart) azt a szimulációs keretrendszer nem kezeli, csak az átadott folyamatban lévő akciókat tárolja, betartásukról a stratégiának kell gondoskodnia (ezáltal lehet „csaló” stratégiákat írni, bár értelmetlen).

3. Az új állapot előáll:

Az előző két lépés végeztével előáll a világ új állapota, ezt eltároljuk és ha még tart a szimulált idő visszalépünk az első lépésre, immáron az új állapotból kiindulva.

3.2 Technikai részletek

3.2.1 Nyelv

A programot 2.11-es verziójú Scala nyelven fejlesztettem, így minden JVM6-os környezetben futtatható.

Több érv is szólt a Scala mellet amikor azt kerestem milyen nyelven fejlesszem a programot. Java-ból megvolt a szükséges tapasztalatom egy ilyen program implementálásához, de a korábban írt szimulációs eszközök fejlesztésékor levont következtetések alapján nehézkesnek bizonyulhat, és könnyen áttekinthetetlenné válhat a kód a nyelv sajátosságai miatt, még jól szervezett struktúra esetén is.

A Java-nál jóval tömörebb szintakszissal rendelkező Scala, a nyelv tervezői által javasolt funkcionális stílussal együtt sokkal tisztább, jól olvashatóbb és kiegészíthetőbb kódot eredményez jelen esetben.

Emellett ugyanakkor a Scala nem távolodik el a Java-s világ jól ismert erősségeitől, a kódból natívan hívhatók a Java osztályok és metódusaik, így az ökoszisztéma számtalan külső könyvtárától sem kell megválni.

A Scala mellet szólt továbbá az „Actors and messages” minta nyelvi szintű támogatása [11]. Ez a tervezési filozófia lehetővé teszi üzenetek és üzenet fogadók könnyű kezelését, ami jelen esetben nagyon megkönnyíti a több ágenses rendszer működését.

3.2.2 Könyvtárak

Külső függőséggel nem rendelkezik a program, a beépített Scala és Java könyvtárakon kívül egyedül a Breeze [12] nevű, open-source, lineáris algebrai és számítási segéd könyvtárat használtam, amit a futtatható .jar fileban mellékeltem.

3.2.3 Felület

A felületet a Java-s világból ismert Swing, Scala-s csomagoló könyvtárával készítettem [13]. Bár a modern GUI fejlesztésben a deklaratív felületek örvendenek népszerűségnek (WPF, JavaFX, Qt Quick) a Scala nyújtotta nyelvi eszközök segítségével a Swing-es felületet is hasonlóan lehet leírni.

4 4. A rendszer használata

4.1 Adatforrások

Időjárás:

A szimuláció során lehetőség van mind a kezdeti, mind az idő függvényében változó időjárás megadására. Ennek formája egy .csv file soraiként történik az alábbi oszlopokkal:

X koordináta (CellUnit)	Y koordináta (CellUnit)	Idő (Másodperc)	Szél erősség (m/s)	Szél irány (fok)

Táblázat 2. Szél adatok CSV formátum

(Alap beállítás szerint a hiányzó lépésközökben lineáris interpolációval kapott értékekkel számol a rendszer.)

Hajó adatai:

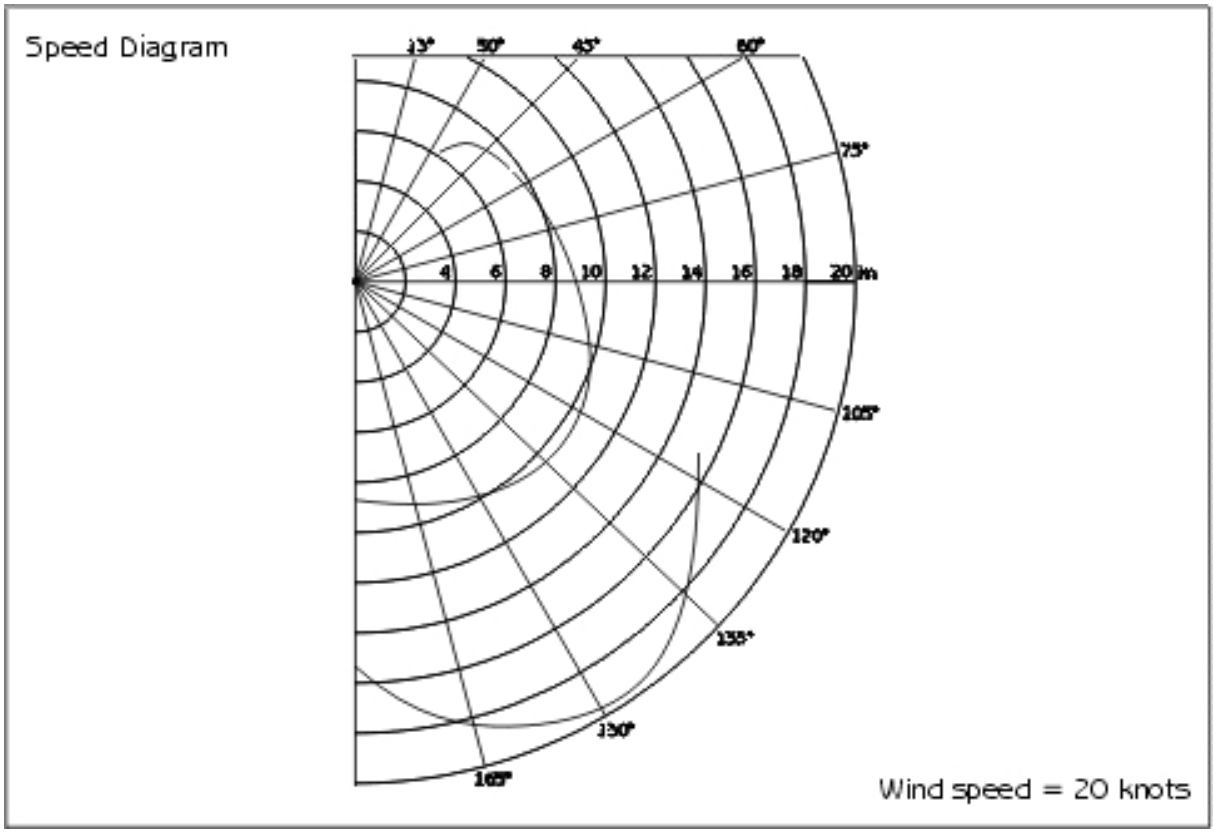
Egy hajóról a szimuláció szempontjából legérdekesebb adatok a teljesítmény jellemzői. Ezt a korábban már említett polár diagrammon szokták a gyártók elérhetővé tenni, ezek közül is azok az érdekesek, amik a hajó maximális elérhető sebességét mutatják az adott vitorlázattal az a adott szélirány és sebesség függvényében.

A szimulációhoz ezeket az adatokat tárolja a program, a látszólagos szélsebesség és irány szerint rendezve. Az alábbi formában (a szél adatokhoz hasonló .csv fileban) lehet új értéket felvinni egy-egy vitorla összeállításhoz.

AWA (Apparent Wind Angle)	TWS (True Wind Speed)	Max boat speed (m/s)

Táblázat 3. Hajó teljesítmény adatok CSV formátum

(Itt szintén lineáris interpolációval kerülnek kiegészítésre a hiányzó adatok)



Ábra 9. Példa, Nautic 311 polar diagram

Ha nem szeretnénk betölteni teljesítmény adatokat, a program tud generálni jellemző, vagy csak elméleti szinten létező polár görbéket, például minden irányba egyforma hatékonyságú hajókról (ezek leginkább a teszteléshez nyújtanak segítséget).

A legtöbb gyártó által megadott polár görbe szimuláción alapul, ideális körülmények feltételezésével (megfelelően beállított, nem fáradt/megnyúlt vitorlák, gyári erősségű hajótest). Nehezíti a pontos adatok megismerését, hogy egy hajó teljesítmény adatainak méréséhez nem rendelkezünk megfelelően kontrollált környezettel.

4.2 Stratégiák és viselkedések

Saját stratégiát Java vagy Scala nyelven tudunk definiálni, jelenleg csak fordítás időben de a jövőbeni tervek között szerepel a dinamikusan betölthető stratégiák támogatása is.

Minden stratégiának az alábbi absztrakt osztályból kell származnia (ez azért nem interfész vagy Scala esetén *trait*, hogy megkönnyítse a Java-Scala interakciót, a Scala absztrakt osztályai közvetlen hívhatók Java-ból is).

```
abstract class Strategy[actorType <: Actor, actionType <: Action[actorType]] {
    def step(actor : actorType, world: World) : List[actionType]
}
```

A stratégiák között is megkülönböztetünk több típust, például az útvonal választó stratégiákat, amelyek célja egy konkrét koordináta elérése a szimulált világban. Ezek leírhatók az alábbi osztállyal.

```
abstract class TravelStrategy[actorType <: Actor, actionType <:
Action[actorType]](val goalPosition : Coordinate[SimUnit]) extends
Strategy[actorType, actionType]{
    ...
}
```

Konkrét példán illusztrálva az alábbi kód egy olyan stratégiát ír le, ami az adott pontba próbál eljutni egy egyenes vonalon (ami nem túl hatékony stratégia egy vitorlás hajó esetén).

Minden szimulációs lépés során az ágens megvizsgálja, hogy a menetiránya milyen szöget zár be a cél felé mutató irányvektorával, majd e szerint (a hajó fordulékonyságának figyelembevételével) korrigál.

```
class DirectLineStrategy(goalPosition : Coordinate[SimUnit]) extends
TravelStrategy[Sailboat, SailingAction](goalPosition){

    override def step(sailboat : Sailboat, world: World):  List[SailingAction]
= {

val dir = DenseVector[Float](sailboat.position.x.toFloat -
goalPosition.x.toFloat, sailboat.position.y.toFloat - goalPosition.y.toFloat)
        val north = DenseVector[Float](0, 1)

        val newHeading = Math.acos((dir dot north)/(dir.length * north.length))
        val delta = sailboat.heading - newHeading.toInt
        val max = sailboat.params.turnSpeed *
SimulationUnits.timeStepInMilliseconds/1000

        if(delta > 0) {
            List(new Turn(Math.min(delta, max)))
        } else {
            List(new Turn(Math.max(delta, -max)))
        }
    }
}
```

4.3 A felület

Az elkészült felület elsődleges funkciója a szimulációs eredmények vizualizálása, a szimulációk futtatásához szükséges konfigurációs feladatokhoz egyelőre még nem készült felület, de a jövőbeni terveim között szerepel.

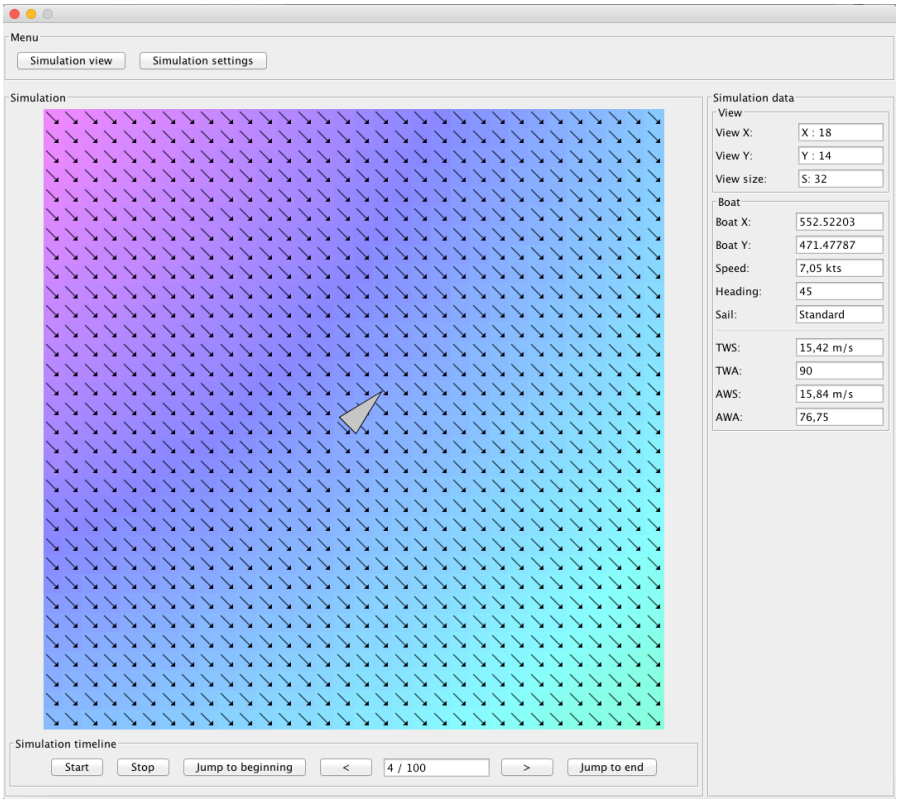
Az indítás után megjelenő felületen a lefuttatott szimuláció eredményeit tekinthetjük meg. Ehhez előre meg kellett adni, még a konfiguráció során a kívánt paramétereket, amelyek alapján a program indítása után lefut a szimuláció.

Az ablak középső részén, a „Simulation” címkével ellátott panelben látjuk a kiválasztott pillanatban a világ állapotát. Az egyes cellákban a színek jelölik a szél erősségét (a korábban leírt módon), a nyilak pedig az irányát. Az egyes hajókat a méretarányos szürke háromszögek jelölik, melyek iránya a hajó irányát mutatják.

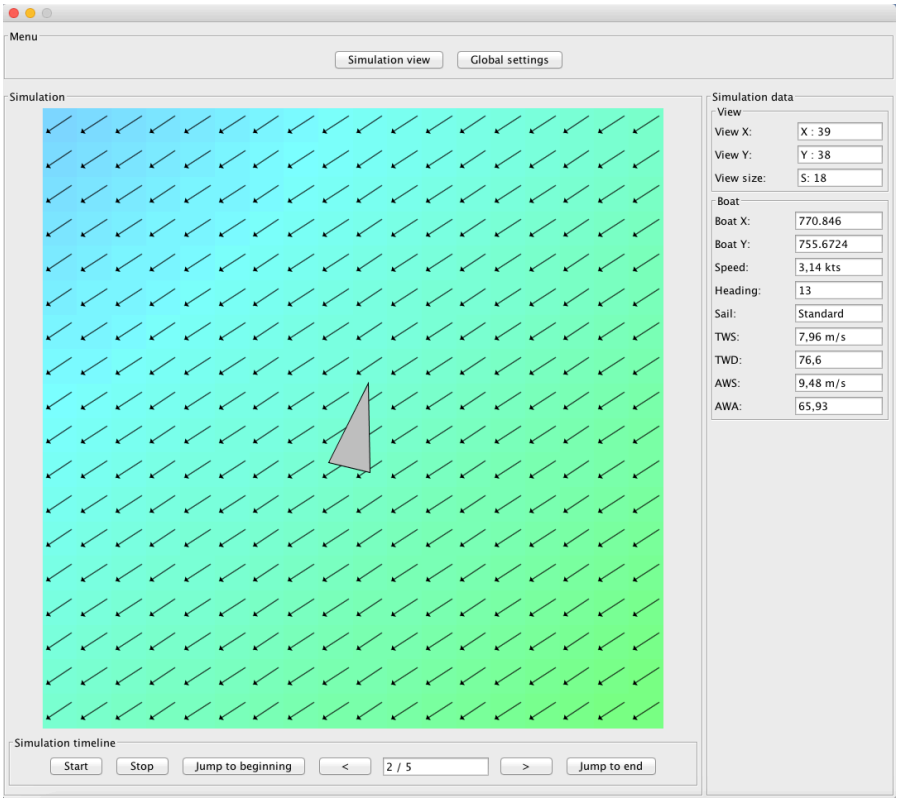
Az alsó „Simulation timeline” menü segítségével léptethetjük az időpillanatokat, míg jobboldali „Simulation data” panelen láthatjuk az éppen aktuális időpillanatra vonatkozó adatokat.

Funkció	**Gomb**
Kamera léptetése nyugatra	Kurzor balra
Kamera léptetése keletre	Kurzor jobbra
Kamera léptetése északra	Kurzor fel
Kamera léptetése délre	Kurzor le
Kamera távolítása	Numpad+
Kamera közelítése	Numpad-

Táblázat 4. Funkciók billentyű kiosztása



Ábra 10. A program felülete



Ábra 11. A program felülete

5 5. Szimulációk

5.1 Stratégiák

A program segítségével több, a vitorlázásban elterjedt, tapasztalati úton felállított stratégiát kívánok megvizsgálni, ezek közül pár példa.

5.1.1 Max VMG

A vitorlázók körében elterjedt vélekedés, hogy a legoptimálisabb út megtalálható az úgynevezett VMG szám folyamatos maximalizálásával. A VMG a Velocity Made Good rövidítésből ered, jelentése a hajó sebesség vektorjának a „jó irányba" mutató komponensének nagysága.

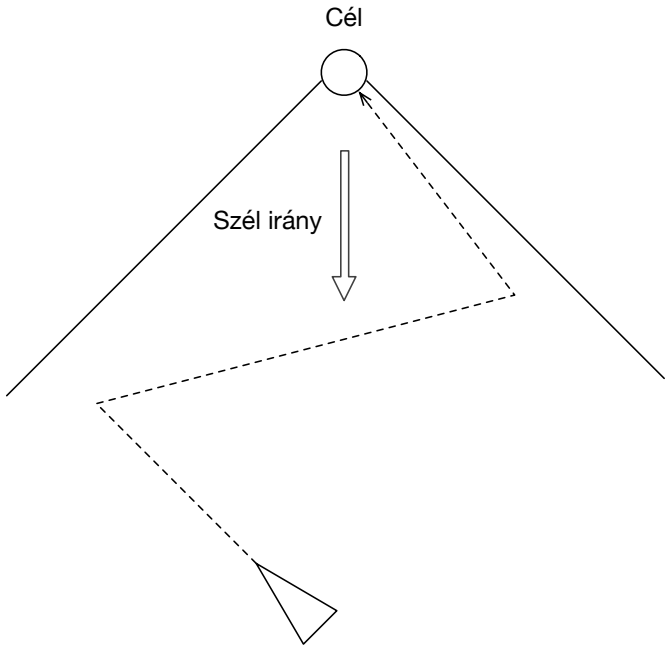
A „jó irány" lehet a hajó céljának iránya, de általában a szél irányát szokták alatta érteni, mivel ezt navigációs eszköz nélkül, egy egyszerű szélsebesség és iránymérővel megállapítható (a hajó sebességének ismeretében).

A VMG egyszerűsített kiszámolásának képlete az alábbi (ahol V a hajó sebessége, Θ a hajó hossztengelye és a cél irányvektora által bezárt szög):

$$VMG = V \cos \Theta$$

5.1.2 Z taktika

A Z taktika abban az esetben alkalmazható, ha a hajó célja a „szél felé” helyezkedik el, azaz ha a célból a szél irányában állított egyenes a hajót metszi.



Ábra 12. Z taktika

Lényege, hogy a hajó a célból indított derékszögű háromszög területén belül, egy Z alakú utat tesz meg 2 fordulóval, úgy hogy a Z közepe körülbelül a táv közepén helyezkedik el.

5.2 Tervek

A már létező stratégiák vizsgálatánál érdekesebb kérdés újakkal előállni. Ehhez először definiálni kell valamilyen alap algoritmust, például a Z taktika általánosítását az N fordulós, alfa szögön belüli taktikát.

Egy ilyen paraméterezett algoritmust utána valamilyen kereső algoritmussal (például evolúciós módszerrel) lehet optimalizálni az adott helyzetekre.

Végül akár egy szimulált világon belül is össze lehet hasonlítani több taktikát és azok egymásra hatását.

6 6. Összefoglalás

6.1 Értékelés

A dolgozat keretén belül a szimulációs keretrendszer készült el, annak éles környezetben való tesztelése a jövőbeli tervek között szerepel. A konkrét tervek között szerepel a korábban említett taktikák vizsgálata és a levont következtetése felhasználása új stratégiák előállítása céljából. A valós környezetben való tesztelés természetesen nagy mértékben az időjárás és hajózási lehetőségek függvénye is.

A technikai döntések közül nem bántam meg a Scala nyelv választását, úgy érzem nagyon hasznos és naprakész tudást szereztem a nyelv megismerésével, és izgatottan várom milyen új dolgokat tanulhatok a további fejlesztés során.

6.2 Jövőbeli tervek

Az előző pontban leírt stratégiákra és felhasználásra vonatkozó terveken kívül a szimulátor fejlesztését sem kívánom abbahagyni. A jövőben szeretném egyrészt teljesen konfigurálhatóvá tenni, a jelenlegi beégetett vagy csak kódból változtatható paramétereket mind kivezetni.

A felhasználói felületen is számos továbbfejlesztési lehetőség említhető meg, a több hajós szimulációk megkönnyítésétől kezdve a stratégiák hatásainak kijelzéséig.

A jövőben szándékozom a forráskódot nyílt projektként közkinccsé tenni.

A fejlesztés során igyekeztem a vitorlázás specifikus *domain* kódot különválasztani a szimulációs kerettől, így remélem később még más területeken is felhasználható lesz a program, vagy legalább bizonyos részei.

7 Ábrák jegyzéke

8 Táblázatok jegyzéke

9 Irodalomjegyzék

[1] TransModeler website, (http://www.caliper.com/transmodeler/), 2015.10.25

[2] SUMO Traffic Modeler, (http://sourceforge.net/projects/trafficmodeler/), 2015.10.25

[3] Sail Simulator, (http://www.sailsimulator.com), 2015.10.25

[4] SailX, (http://www.sailx.com/en/), 2015.10.25

[5] NatuicEd, (http://www.nauticed.org), 2015.10.25

[6] Binns, Jonathan R., Frank W. Bethwaite, and Norman R. Saunders. "Development of a more realistic sailing simulator." (2002).

[7] Volvo Ocean Race offical website, (http://www.volvooceanrace.com/en/news/6864_Cross-platform-audience-surge-for-the-Volvo-Ocean-Race-2011-12.html)

[8] SkySails offical website, (http://www.skysails.info), 2015.10.25

[9] North Sails Hungary offical website, (http://www.hu.northsails.com/SAILS/DownwindSails/DownwindSailPerformanceGuide/tabid/32999/language/en-US/Default.aspx), 2015.10.25

[10] Filippo Castiglione (2006) Agent based modeling. Scholarpedia, 1(10):1562.

[11] Odersky, Martin, Lex Spoon, and Bill Venners. Programming in scala. Artima Inc, 2008.

[12] Breeze Scala numerical library, (https://github.com/scalanlp/breeze), 2015.10.25

[13] Maier, Ingo. "The scala. swing package." Scala Improvement Process (SID) 8 (2009).