# Miinoitteen tehokkuuden arviointi

Niko Sairo

**Perustieteiden korkeakoulu**

Kandidaatintyö
Espoo 10.05.2021

**Vastuuopettaja**

Prof. Ahti Salo

**Työn ohjaaja**

DI Juho Roponen

Aalto-yliopisto
Perustieteiden
korkeakoulu

Aalto-yliopisto
Perustieteiden
korkeakoulu

**Aalto-yliopisto, PL 11000, 00076 AALTO**
**www.aalto.fi**
**Tekniikan kandidaatintyön tiivistelmä**

**Tekijä** Niko Sairo

**Työn nimi** Miinoitteen tehokkuuden arviointi

**Koulutusohjelma** Teknillinen fysiikka ja matematiikka

**Pääaine** Matematiikka ja systeemitieteet **Pääaineen koodi** SCI3029

**Vastuuopettaja** Prof. Ahti Salo

**Työn ohjaaja** DI Juho Roponen

**Päivämäärä** 10.05.2021 **Sivumäärä** 20+10 **Kieli** Suomi

**Tiivistelmä**
Merimiinat ovat merisodankäynnin välineitä, joilla pyritään vahingoittamaan tai upottamaan aluksia ja estämään merialueiden käyttöä. Merimiinoja on käytetty tehokkaasti useissa sodissa. Varman saatavuuden ja suhteellisen edullisen hinnan ansiosta merimiinat ovat myös hyödynnettävissä valtioille, joilla on pieni sotilasbudjetti.

Tässä kandidaatintyössä tutustuttiin merimiinatyyppeihin, miinanraivaukseen ja merimiinoitteiden mallintamisen historiaan. Kirjallisuuskatsauksen pohjalta luotiin ohjelmisto, jolla voidaan arvioida merimiinoitteen tehokkuutta. Miinoituksen tehokkuuden arviointimenetelmänä käytettiin Monte Carlo -simulaatiota, jossa generoidaan suuri määrä satunnaisia reittejä miinoitteen läpi. Ohjelmisto mahdollistaa miinoitteen tehokkuuden arvioinnin ohjelmiston käyttäjän valitsemalla kartalla. Miinan räjähdyksen aiheuttamaa vahinkoa arvioitiin miinan räjähdyksestä syntyvän paineiskun ja kaasukuplan avulla.

Ohjelmistoa testattiin esimerkkitarkastelussa, jossa satunnaisesti generoitu miinoite luotiin käsin piirrettyyn karttakuvaan. Kyseisen miinoitteen tehokkuutta tarkasteltiin ajamalla miinoitteen läpi 10 000 neljän reitin sarjaa. 40 000 reitin Monte Carlo -simulaatio osottautui laskennallisesti raskaaksi, sillä ohjelmiston ajamiseen kului aikaa lähes puolitoista tuntia. Esimerkkitarkastelua varten luotu miinoite säilytti vaarallisuutensa sarjan loppupään reiteillä ja yksikään miinoista ei räjähtänyt vahingoittamatta alusta. Miinoitteen tehokkuuden arviointi ei ole yksiselitteistä ja siksi useamman tehokkuusmittarin käyttäminen on perusteltua. Lisäksi Monte Carlo -simulaatiossa on punnittava, mikä on riittävä iteraatioiden lukumäärä. Iteraatioiden lukumäärän kasvattaminen parantaa tehokkuusmittarien tarkkuutta, mutta samalla laskenta-aika kasvaa.

**Avainsanat** merimiinoite, miinanraivaus, MATLAB, Monte Carlo -simulaatio , Simple Initial Threat

Aalto-yliopisto
Perustieteiden
korkeakoulu

**Aalto University, P.O. BOX 11000, 00076 AALTO**
**www.aalto.fi**
**Abstract of the bachelor's thesis**

**Author** Niko Sairo

**Title** Evaluation of Minefield Efficiency

**Degree programme** Engineering Physics and Mathematics

**Major** Mathematics and Systems Sciences **Code of major** SCI3029

**Supervisor** Prof. Ahti Salo

**Advisor** M.Sc. Juho Roponen

**Date** 10.05.2021 **Number of pages** 20+10 **Language** Finnish

**Abstract**

Naval mine is a naval warfare device whose purpose is to damage or sink ships or to block the usage of a designated sea area. Naval mines have been used effectively in several wars. Good availability and relatively low price allow even countries with low military budget to utilize naval mines.

This thesis explored naval mine types, minesweeping, and history of naval minefield modeling. A tool was also built to evaluate minefield efficiency. The used evaluation method was Monte Carlo simulation in which a large amount of routes are generated through the minefield. The software makes it possible to evaluate a minefield on a map selected by the software user. Mine damage is assessed by the shock wave and gas bubble caused by the explosion.

The software was tested with an example where a random minefield was generated for a hand drawn map. The efficiency of the minefield was evaluated by running 10 000 series of four paths. A 40 000 path Monte Carlo simulation proved to be computationally heavy, because the run time was almost one and a half hours. The minefield proved to maintain the effectiveness and not a single mine exploded without damaging any ships. The evaluation of minefield efficiency is not unambiguous and as a result, the use of multiple measures of effectiveness is justified. In addition, one should consider what is the right number of iterations of Monte Carlo simulation. Increasing the number of iterations leads to more accurate results but it also increases the run time.

**Keywords** naval minefield, minesweeping, MATLAB, Monte Carlo simulation, Simple Initial Threat

# Sisällysluettelo

**Tiivistelmä** **3**

**Tiivistelmä (englanniksi)** **4**

**Sisällysluettelo** **5**

**1 Johdanto** **6**

**2 Tutkimuksen tausta** **6**
2.1 Miinoitteiden mallinnus . . . . . 7

**3 Malli** **8**

**4 Karttakuvan käsittely ja reittien generointi** **12**

**5 Esimerkki ohjelman käytöstä** **14**

**6 Yhteenveto** **17**

# 1 Johdanto

Merimiinat ovat merisodankäynnin välineitä, joiden käytöllä pyritään vahingoittamaan tai upottamaan aluksia ja estämään merialueiden käyttöä. Merimiinat ovat yksinkertaisen toimintaperiaatteensa vuoksi merisodankäynnin välineiksi suhteellisen edullisia ja niitä on käytetty tehokkaasti useissa sodissa (Reinbach, 2007). Useat valtiot valmistavat merimiinoja, mikä on johtanut siihen, että niiden saatavuus kansainvälisessä asekaupassa on hyvä. Hyvän saatavuuden ja edullisen hinnan ansiosta merimiinat ovat pientenkin sotilasbudjetin omaavien valtioiden hyödynnettävissä (Romberger, 1996). Merimiinoite koostuu yhdestä tai useammasta merialueelle sijoitetusta merimiinasta, jotka voivat erota ominaisuuksiltaan. Merimiinoitteen suunnittelulla pystytään edistämään miinoitteesta saatavaa hyötyä. Merimiinoitteen tehokkuuden arviointiin soveltuva työkalu tukee suunnitteluprosessia antamalla informaatiota suunnittelun tuloksista.

Merimiinoitteiden tarkka mallintaminen on haastavaa, mistä johtuen malleissa päädytään usein tekemään mallia yksinkertaistavia oletuksia tai arvioita. Yleisin malleissa esiintyvä yksinkertaistus on, että miinojen välisiä vuorovaikutuksia ei huomioida. Merimiinoitteet voivat myös koostua useista ominaisuuksiltaan eroavista merimiinoista, mitä ei olla huomioitu vanhemmissa malleissa. Joitakin mallintamiseen liittyviä ongelmia on pyritty ratkaisemaan hyödyntämällä Monte Carlo -simulaatiota, mutta tarkkojen tulosten saanti Monte Carlo -simulaationa on laskennalisesti raskasta.

Tässä kandidaatintyössä ei arvioida mitään olemassaolevaa miinoitetta, vaan työn tarkoituksena on luoda ohjelma, jolla pystytään arvioimaan miinoitteen tehokkuutta yleisemmin. Ohjelma mahdollistaa käyttäjän valitseman miinoitteen tehokkuuden arvioinnin käyttäjän valitsemassa vesistössä. Ohjelma mahdollistaa miinojen erojen huomioinnin räjähdeaineen massan ja herätteen kantaman osalta. Miinoitteen muodostavien miinojen sijainnin oletetaan olevan eksakti. Arviointimentelmänä käytetään Monte Carlo -simulaatiota, jossa generoidaan suuri määrä satunnaisia reittejä miinoitetussa vesistössä. Generoiduilla reiteillä koetuista miinakontakteista voidaan muodostaa jakaumia, jotka antavat informaatiota miinoituksen tehokkuudesta. Tutkimuksessa keskitytään kiinnitettyihin heräte- ja kosketusmiinoihin. Lisäksi miinojen ylikulkuominaisuutta ja miinojen keskinäisiä vuorovaikutuksia ei huomioida. Ylikululla viitataan miinan ominaisuuteen, jossa miina ei räjähdä ensimmäisen aluksen kohdalla, vaan esimerkiksi vasta viidennen aluksen kohdalla.

# 2 Tutkimuksen tausta

Merimiinat voidaan jaotella eri luokkiin esimerkiksi niiden toimintaperiaatteen tai sijainnin mukaan. Toimintaperiaatteen mukaan merimiinat voidaan jakaa kosketus- ja herätemiinoihin. Kosketusmiinat laukeavat aluksen fyysisestä kosketuksesta. Herätemiinat laukeavat aluksen aiheuttamasta herätteestä tai herätteiden yhdistelmästä. Herätemiinan teho perustuu räjähdyksen luomaan vedenalaiseen paineiskuun. Yleisimpiä herätetyyppejä ovat ääni-, paine- ja magneettiherätteet. Ääniherätteellä varustettu miina reagoi aluksen liikkumisen synnyttämään ääneen. Paineherätteellä

varustettu miina reagoi paineen muutokseen, joka aiheutuu aluksen liikkuessaan syrjäyttämästä vesimäärästä. Magneettiherätteellä varustettu miina reagoi paikallisen magneettikentän muutokseen, joka aiheutuu ohi kulkevasta aluksesta. Herätemiinojen etuna kosketusmiinoihin verrattuna on, että niiden ei tarvitse koskettaa alusta. Täten ne voivat laueta ja vahingoittaa paljon kauempana sijaitsevaa alusta. (Levie, 1992)

Sijainnin mukaan merimiinat voidaan jakaa kelluviin ja ankkuroituihin miinoihin sekä pohjamiinoihin. Kelluvat miinat päästetään vapaasti ajelehtimaan merialueelle. Niiden ongelmana kuitenkin on, että niillä on riski aiheuttaa vahinkoa siviileille ja vesistöä miinoittaneelle taholle johtuen kontrolloinnin puutteesta. Ankkuroidut miinat puolestaan kiinnitetään meren pohjaan vaijerilla, jota kutsutaan syvyytinvaijeriksi. Kosketusmiinat joudutaan usein ankkuroimaan lähelle veden pintaa, jotta kontakti alukseen olisi mahdollinen. Tämä tekee kosketusmiinoista alttiimpia mekaaniselle miinanraivukselle. Herätemiinat voidaan ankkuroida huomattavasti kauemmas vedenpinnasta ja riittävän matalissa vesissä ne voidaan myös laskea merenpohjaan, jolloin niistä voidaan sijainnin mukaan käyttää nimitystä pohjamiina. (Levie, 1992)

Miinanraivaus on miinantorjunnan keino, jossa miinat tehdään vaarattomaksi purkamalla tai räjäyttämällä ne kontrolloidusti. Miinanraivaustapa riippuu paljon raivattavien miinojen tyypistä. Mekaanisessa miinanraivauksessa miinanraivaajaksi kutsuttu alus vetää perässään punnuksilla ja kohoilla varustettua vaijeria, johon on kiinnitetty yksi tai useampia leikkureita. Leikkureiden on määrä katkaista syvyytinvaijeri, jolla kosketusmiina on kiinnitettynä meren pohjaan. Syvyytinvaijerin katketessa miina nousee pintaan, minkä jälkeen se voidaan tehdä vaarattomaksi. Heräteraivauksessa puolestaan pyritään simuloimaan herätemiinojen laukaisevaa herätettä ja näin räjäyttämään miinat turvallisesti. Miinoja voidaan myös raivata tykistötulella ja sukeltamalla. (Reinbach, 2007)

Miinanraivauksen vaikeuttamiseksi on myös kehitelty menetelmiä. Esimerkiksi yhdistettyjen herätteiden käyttäminen herätemiinoissa hankaloittaa raivausta. Herätemiinoihin voidaan myös ohjelmoida ylikulkuja, jolloin miina räjähtää esimerkiksi vasta viidennen herätteen aiheuttavan aluksen kohdalla.

## 2.1 Miinoitteiden mallinnus

Miinoitteita on mallinnettu yksinkertaistetusti erinäisissä merisodankäyntiä laajemmin kuvaavissa malleissa. Näistä yksi esimerkki on Yhdysvaltojen merivoimien aikanaan käyttämä Enhanced Naval Warfare Gaming System (ENWGS). Kyseisessä mallissa miinojen tarkkaan sijaintiin ei oteta kantaa, vaan miinoituksen ja aluksen välisiä vuorovaikutuksia arvioidaan miinoituksen pinta-alan, aluksen miinoituksessa kulkeman matkan ja miinojen herätteen kantaman avulla. Mikäli alus on vaikutuksessa miinan kanssa, miina ja alus poistetaan mallista, ellei alus ole määritelty miinanraivaajaksi.

Kehittyneemmät mallit keskittyvät miinoitteen suunnitteluun, miinanraivaukseen tai molempiin. Miinoitteiden suunnitteluun käytettyjä malleja ovat muunmuassa Uncountered Minefield Planning Model (UMPM) ja Analytical Countered Minefield Planning Model (ACMPM). UMPM huomioi miinan ja aluksen välisen etäisyyden, kun arvioidaan miinan räjähdyksen alukseen aiheuttamaa vahinkoa

(Romberger, 1996). ACMPM on kuin UMPM, mutta lisäksi se pystyy käsittelemään useista eri miinatyypeistä luotuja miinoitteita ja suunnittelussa voidaan myös huomioida vastapuolen miinantorjuntaa (Washburn ja Kress, 2009).

Miinantorjuntaan keskittyviä malleja puolestaan käytetään raivaussuunnitelmien luomiseen ja testaamiseen. Käytettyjä malleja ovat muun muassa Navy's Non Uniform Coverage Evaluator (NUCEVAL), Uniform Coverage Planner (UCPLAN), Cognitive Planning Aid (COGNIT) ja MIXER. NUCEVAL arvioi sille syötettyä raivaussuunnitelmaa, kun taas UCPLAN tulostaa raivaussuunnitelman halutulle puhdistustasolle. NUCEVAL ja UCPLAN ovat nykystandardeilla liian yksinkertaisia malleja, sillä ne eivät käsittele informaatiota miinojen lukumääristä ja tyypeistä. COGNIT on malli, joka huomioi mainitut yksityiskohdat sekä mahdollistaa vastapuolen miinantorjunnan huomioimisen. COGNITin ongelma on, että sillä voidaan simuloida vain yhdenlaisia miinatyyppejä ja miinanraivaajia kerralla. (Romberger, 1996) MIXER on malli, jota voidaan hyödyntää useista erilaisista miinatyypeistä koostuvien miinoitteiden raivausten suunnitteluun. MIXER tarjoaa mahdollisuuden luodun raivaussuunnitelman arviointiin ja se voi myös muodostaa tehokkaita raivaussuunnitelmia. (Washburn, 1995) Näiden mallien miinojen ja miinanraivaajien väliset vuorovaikutuksen vastaavat UMPM:ssä määriteltyjä miinojen ja miinanraivaajien välisiä vuorovaikutuksia. Edellä esiintyvien mallien lisäksi on luotu malleja, jotka mahdollistavat miinoitteiden ja miinantorjunnan tarkemman simuloinnin. Yksi tällaisista malleista on Total Mine Simulation System (TMSS) (Timothy ja Floore, 2011). TMSS:n ongelmana on, että se on yksinkertaistetumpiin malleihin verrattuna hidas, mikä rajaa sen käyttökohteita.

Miinoituksen tehokkuuta on pyritty arvioimaan erilaisilla tehokkuutta kuvaavilla mittareilla. Eräs yksinkertainen tehokkuusmittari on Simple Initial Threat (SIT), joka tarkoittaa todennäköisyyttä sille, että ensimmäinen miinoituksen läpi kulkeva alus uppoaa tai vahingoittuu miinan vaikutuksesta (Redmayne, 1996). SIT:n ongelmana on, että se ei anna informaatiota ensimmäisen aluksen jälkeen miinoituksen läpi kulkeviin aluksiin kohdistuvasta uhkasta. Miinantorjunnassa käytettyjä tehokkuusmittareita ovat muun muassa miinoitteen raivaamiseen vaadittu työmäärä, puhdistustason keskiarvo ja raivauksessa syntyneiden uhrien odotusarvo. Vaikkakin miinoitteen suunnittelu ja miinantorjunnan suunnittelu eivät tähtää samaan päämäärään, ne ovat vahvasti linkittyneenä toisiinsa peliteorian näkökulmasta, missä miinoittaja voi yrittää huomioida raivaajan vastatoimet, raivaaja voi yrittää huomioida vastatoimien vastatoimet ja niin edelleen (Washburn ja Kress, 2009). Kyse on asetelmasta miinoittaja vastaan raivaaja, jossa toistaiseksi raivaaja on ollut alakynnessä.

# 3 Malli

Tässä työssä käytetään tätä työtä varten kehitettyä mallia, joka pohjautuu UMPM:ään hyödyntämällä samoja parametreja, kuten miinan herätteen kantamaa ja miinan vahinkokäyrää. Mallissa miinan oletetaan räjähtävän varmasti, mikäli aluksen ja miinan välinen etäisyys on miinan herätteen kantamaa pienempi. Miinan räjähdysto-

dennäköisyys on

$$P_a(x) = \begin{cases} 1, & \text{jos } x \leq d \\ 0, & \text{muuten,} \end{cases} \tag{1}$$

missä miinan ja aluksen etäisyys on $x$ ja miinan herätteen kantama on $d$. Myös raivauksen oletetaan onnistuvan varmasti, mikäli miina ei räjähdä ($P_a(x) = 0$), aluksen raivaussäde on vähintään yhtä suuri kuin miinan vedenpinnalle projisoidun sijainnin ja aluksen välinen etäisyys ja aluksen raivaussyvyys on vähintään yhtä suuri kuin miinan syvyys. Miinan raivauksen onnistumisen todennäköisyys on

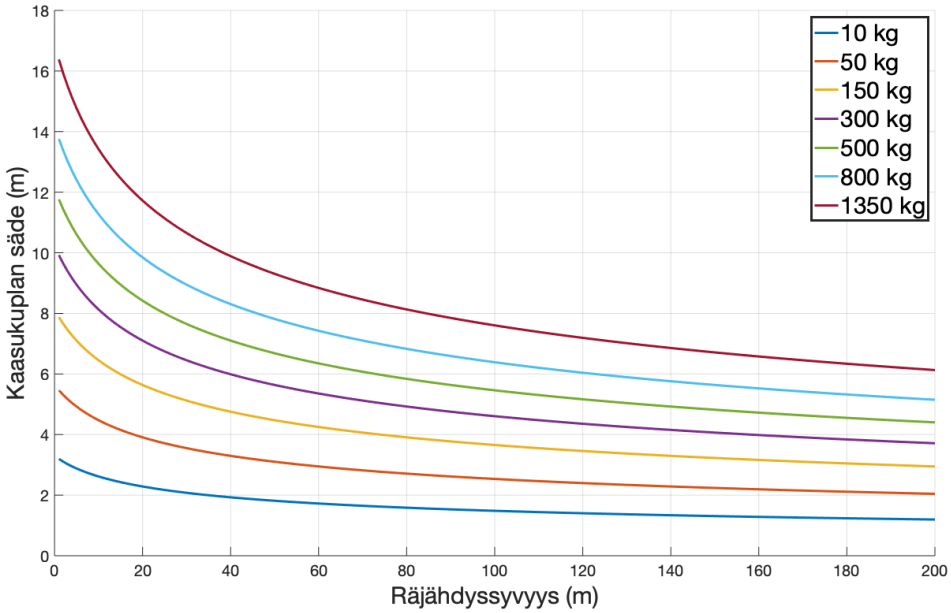
$$P_s(x_{2d}, h) = \begin{cases} 1, & \text{jos } P_a(x) = 0,\ x_{2d} \leq s_{2d} \text{ ja } h \leq s_h \\ 0, & \text{muuten,} \end{cases} \tag{2}$$

missä $x_{2d}$ on miinan vedenpinnalle projisoidun sijainnin ja aluksen välinen etäisyys, $h$ on miinan syvyys, $s_{2d}$ on aluksen raivaussäde ja $s_h$ on aluksen raivaussyvyys.

Miinan räjähdyksen vaikutuksia arvioitaessa aluksen ja miinan välisenä etäisyytenä käytetään miinan herätteen kantamaa. Miinan räjähtäessä miinakontaktit voidaan karkeasti jakaa kosketusräjähdyksiin ja kosketuksettomiin räjähdyksiin. Kosketusräjähdyksessä miinan räjähdys aiheuttaa reiän aluksen runkoon tai miina räjähtää niin lähellä alusta, että alus joutuu vuorovaikutukseen miinan räjähdyksen synnyttämän kaasukuplan kanssa. Molemmissa kosketusräjähdyksiksi luokiteltavissa tapauksissa seuraukset alukselle ovat tuhoisat. Kaasukuplan maksimisäde noudattaa kaavaa

$$R_{pmax} = 1.53 \sqrt[3]{\frac{m}{1 + 0.1H}}, \tag{3}$$

missä $m$ on räjähdeaineen massa ja $H$ on syvyys, jossa räjähdys tapahtuu. (Szturomski, 2015) Kaasukuplan maksimisäteen arvoja on esitelty kuvassa 1.




Kuva 1: Kaasukuplan maksimisäde $R_{pmax}$ räjähdeaineen massan ja rähdyssyvyyden funktiona (Szturomski, 2015).

Merivoimien kosketusmiinoissa käytetyt räjähdeainemäärät vaihtelevat 50 kilogrammasta 200 kilogrammaan ja herätemiinoissa käytetyt räjähdeainemäärät vaihtelevat 80 kilogrammasta 750 kilogrammaan (Merisotilaan käsikirja, 2010). Kuvasta (1) havaitaan, että jopa 500 kilogramman räjähdeainelastauksella kaasukuplan maksimisäde jää alle kymmenen metrin, kun räjähdyssyvyys on kymmenen metriä. Täten karkeasti arvioiden yli kymmenen metrin päässä räjähdyksestä sijaitsevien alusten kokemia vaikutuksia voidaan arvioida pelkästään miinan räjähdyksen aiheuttaman paineiskun avulla. Räjähdyksen ollessa alle kaasukuplan säteen päässä aluksesta, kaasukuplan aiheuttama vahinko tulisi ottaa huomioon. Kaasukuplan aiheuttamaa vahinkoa on haastavaa arvioida numeerisesti, mutta aluksen lähikontaktissa kokema paine on kuitenkin laskennallista paineiskua suurempi (Kiciński ja Szturomski, 2020). Tässä mallissa kaasukuplan vaikutusta arvioidaan asettemalla kaasukuplan kanssa vuoravaikutuksessa olevan aluksen kokema paine niin suureksi, että se upottaa aluksen. Räjähdyksen aiheuttamalle paineiskulle on kokeellisesti määritelty erilaisia kaavoja ja vakioita joita esitellään taulukossa 1.

Taulukko 1: Tutkijoiden käyttämiä kaavoja ja vakioita räjähdyksen aiheuttaman maksimipaineen arvioimisessa trinitrotolueenille (TNT).

| **Nro** | Kaava | Parametrien kuvaukset | Esittäjä ja vuosi |
|---|---|---|---|
| **1** | $p_{max} = 52.3(\frac{\sqrt[3]{m}}{R})^{1.13}$ [MPa] | $m$ on räjähdeaineen massa (kg) ja $R$ on etäisyys räjähdyksen keskipisteestä (m) | R.Cole 1498 (Kiciński ja Szturomski, 2020) |
| **2** | $p_{max} = 21000(\frac{\sqrt[3]{m}}{R})^{1.13}$ [psi] | $m$ on räjähdeaineen massa (lb) ja $R$ on etäisyys räjähdyksen keskipisteestä (ft) | A.H. Keil (Keil, 1961) |
| **3** | $p_{max} = P_c(\frac{a_c}{R})^{1.13}$ [MPa] | $a_c = \sqrt[3]{\frac{3m}{4\pi\rho_{TNT}}}$ on palloksi muotoillun räjähdeaineen säde (m), $P_c$ on räjähdeaineen tiheydestä riippuva vakio ja $R$ on etäisyys räjähdyksen keskipisteestä (m) | Geers & Hunter (Geers ja Hunter, 2001) |

Mallissa miinan aiheuttamaa vahinkoa arvioidaan miinan räjähdyksen aiheuttaman paineiskun (MPa) avulla

$$p_{max} = \begin{cases} 1420\left[\frac{\sqrt[3]{\frac{3m}{4\pi 1520}}}{x}\right]^{1.13}, & \text{jos } x > R_{pmax} \\ 10 \cdot 1420\left[\frac{\sqrt[3]{\frac{3m}{4\pi 1520}}}{x}\right]^{1.13}, & \text{jos } x \leq R_{pmax}, \end{cases} \tag{4}$$

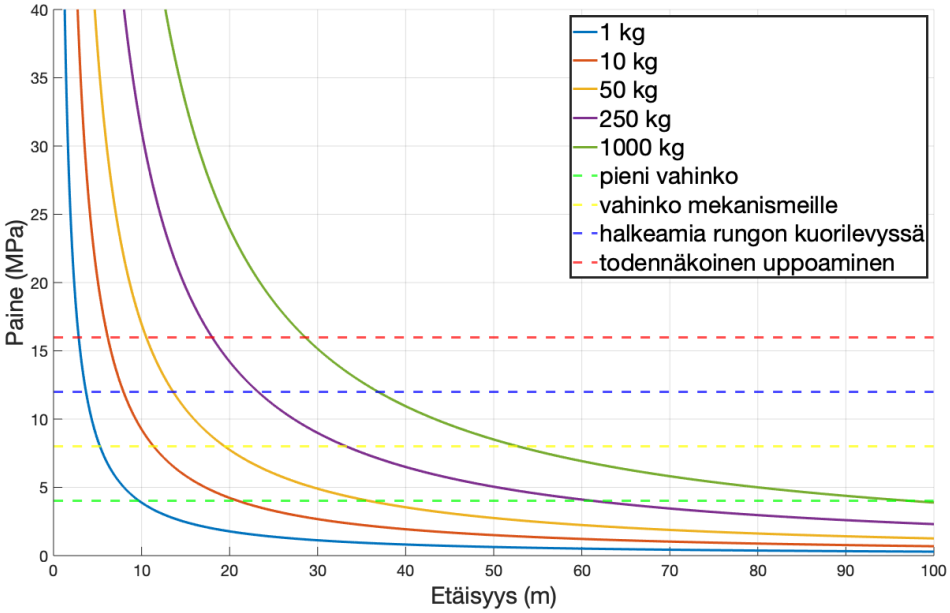
missä $x$ on miinan ja aluksen välinen etäisyys räjähdyshetkellä ja $m$ on räjähdeaineen massa. Mallissa käytetään TNT:tä, jonka tiheys on 1520 kg/m$^3$ ja sitä vastaava vakio $P_c = 1420$ MPa. Paineisku mukailee taulukon 1 kolmatta kaavaa, kun aluksen etäisyys räjähdyksestä on kaasukuplan maksimisädettä suurempi. Tapauksessa, jossa $x \leq R_{pmax}$, paineiskun suuruudeksi on approksimoitu kymmenkertainen arvo etäisyydellä $x$. Tämän kaavan valintaan päädyttiin, koska se on tuorein ja mahdollistaisi myös paineiskun arvioinnin eri räjähdysaineilla ja räjähdysaineen tiheyksillä. Paineiskun avulla voidaan määritellä vahinkoa kuvaava kategoriamuuttuja D, joka on esitelty taulukossa 2.

Taulukko 2: Vahinkoa kuvaava kategoriamuuttuja D ja kategoriaan luokitellun paineiskun vaikutuksen kuvaus.

| Paineisku (MPa) | D | Paineiskun vaikutus |
|---|---|---|
| $> 16$ | 1 | todennäköinen uppoaminen |
| $12 - 16$ | 0.75 | halkeamia rungon kuorilevyssä |
| $8 - 12$ | 0.5 | vahinko mekanismeille |
| $4 - 8$ | 0.25 | pieni vahinko |
| $< 4$ | 0 | ei vahinkoa sota-aluksille |

Kuvassa 2 esitetään taulukon 1 kolmannen kaavan mukainen paineiskun maksimiarvo ja tätä vastaava aluksen kokema vahinko D. Laskettaessa aluksen reitillä

kokemia vuorovaikutuksia, koettu vahinko on koettua paineiskua vakaampi mittari, sillä koettu vahinko jättää liian pienet paineiskut huomioimatta ja skaalaa todella suuret paineiskut samanarvoisiksi muiden paineiskujen kanssa, jotka riittäisivät todennäköisesti upottamaan aluksen.




Kuva 2: Paineiskun $p_{max}$ maksimiarvo etäisyyden $R$ ja räjähdeaineen massan $m$ funktiona. Kuva kertoo myös erisuuruisten paineiskujen vaikutuksista sotilasaluksiin.

# 4 Karttakuvan käsittely ja reittien generointi

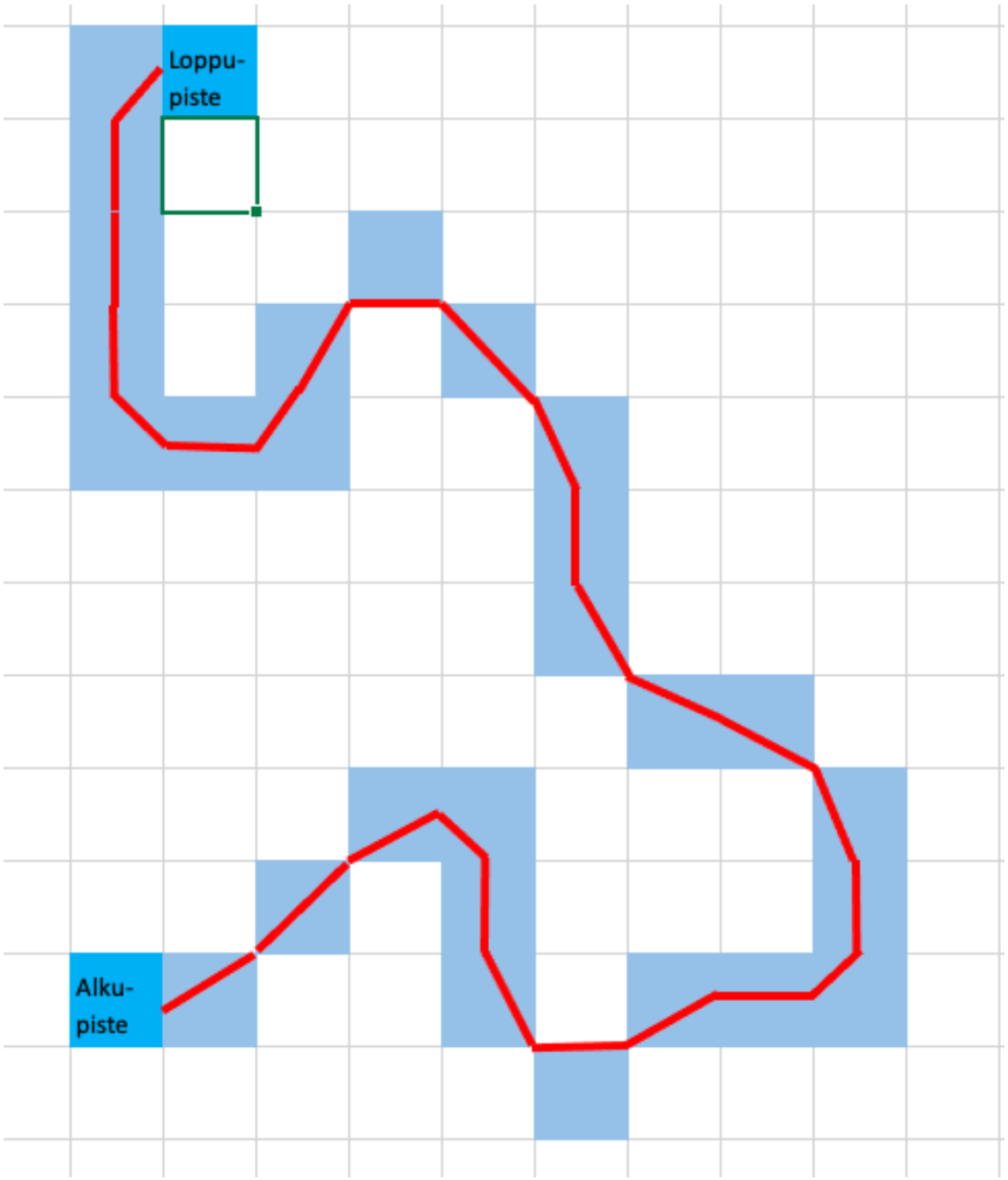
Ohjelman merkittävä ominaisuus on mahdollisuus hyödyntää käyttäjän itsensä valitsemia oikeita karttakuvia, mikä mahdollistaa miinoitteiden arvioinnin halutulla alueella. Ohjelmalle annetaan mustavalkokuva merialueesta, jossa valkoinen alue kuvaa merialuetta ja musta alue kuvaa rajoitettua aluetta, joka voi olla esimerkiksi maata tai jotain muuta aluetta, jonka yli alukset eivät voi kulkea. Kuva käännetään MATLABissa matriisiksi, jossa kuvan kukin pikseli edustaa yhtä matriisin solua. Täten vaaka-akselin indeksointi alkaa matriisissa kuvan vasemmasta reunasta ja pystyakselin indeksöinti alkaa matriisissa kuvan yläreunasta. Ohjelmalle määritellään myös skaala, joka ilmaisee, kuinka montaa metriä pikselin sivu edustaa. Pikselien väri määritellään RGB-värimallilla, missä värit muodostetaan käyttämällä punaisen, vihreän ja sinisen värisävyjen kombinaatioita. MATLABissa pikselin väri määritellään kolmiulotteisella värivektorilla $\vec{c} = (r, g, b)$, missä $r,\ g,\ b \in [0, 255]$ kun vektorin alkioiden luokkana käytetään 8-bittistä etumerkitöntä kokonaislukua. (Kumar ja Verma, 2010) Tällöin mustan pikselin $\vec{c} = (0, 0, 0)$ ja valkoisen pikselin

$\vec{c} = (255, 255, 255)$. Ohjelma muuntaa annetun kuvan kaksiväriseksi:

$$\vec{c} = \begin{cases} (0,0,0), & \text{jos } r \vee g \vee b < 128 \\ (255,255,255), & \text{muuten.} \end{cases} \tag{5}$$

Karttakuvan syöttämisen jälkeen kuvasta valitaan vähintään kaksi pikseliä, joiden välille reittejä generoidaan. Ensimmäinen valittu pikseli kuvastaa reitin alkupistettä ja viimeinen reitin loppupistettä. Mikäli pikseleitä on valittu enemmän kuin kaksi, ensimmäisen ja viimeisen pikselin väliset pikselit kuvaavat välipisteitä, joiden kautta reitit generoidaan. Tätä ominaisuutta voidaan hyödyntää tilanteissa, joissa alku- ja loppupisteen välillä on rajoitettu alue, joka ajaisi reittigeneraattorin umpikujaan. Kaikkien valittujen pikseleiden tulee sijoittua merialueelle.

Tämän jälkeen valittujen pisteiden välille generoidaan ohjelman käyttäjän valitsema määrä satunnaisia reittejä. Reitit generoidaan MATLABilla luodulla algoritmilla, joka valitsee satunnaisen suunnan suosien suuntia, jotka lyhentävät matkaa seuraavaan valittuun pisteeseen. Tällä ehkäistään tilannetta, jossa algoritmi ei piirrä reittiä alkupisteestä loppupisteeseen vaan jää satunnaisesti pyörimään merialueelle. Algoritmin tapa valita reitin seuraava suunta on muutenkin perusteltua, sillä reitin mahdollisesti miinoitetulla merialueella ei tule olla liian pitkä johtuen siitä, että se kasvattaa aluksen riskiä joutua merimiinan vahingoittamaksi. Suunnan generoinnin jälkeen algoritmi generoi satunnaisen askelmäärän väliltä 1-30. Askelmäärän generoinnin jälkeen reittiä piirretään generoituun suuntaan generoidun askelmäärän verran. Mikäli kyseisessä suunnassa on rajoitettu alue generoitua askelmäärää pienemmän askelmäärän päässä, reittiä piirretään vain sallitun ja rajoitetun alueen rajalle. Tämän jälkeen algoritmi generoi uuden suunnan ja askelmäärän edellä kuvaillulla tavalla. Reitti pikselin sisällä on toteutettu suoraviivaisemmin. Pikselin sisäinen reitti koostuu noin metrin välein janalle sijoitetuista pisteistä. Jana on piirretty pikselin sisäisen alku- ja loppupisteen välille. Pikselin sisäinen alkupiste on pikselin ja aikaisemman pikselin yhteinen kulma tai yhteisen sivun keskipiste. Pikselin sisäinen loppupiste on pikselin ja seuraavan pikselin yhteinen kulma tai yhteisen sivun keskipiste. Pikselin sisäistä reittiä on kuvattu kuvassa 3. Työssä käytetyt MATLAB-koodit löytyvät työn liitteistä A, B, C ja D.




Kuva 3: Punaiset janat kuvaavat pikselien sisäistä reittiä. Siniset neliöt kuvaavat alku- ja loppupisteen välille generoitua reittiä, joka ei ole pikselien sisäisen reitin havainnoillistamisen vuoksi lyhin mahdollinen.

# 5 Esimerkki ohjelman käytöstä

Tässä osiossa kuvataan yhtä ohjelman käyttötavoista esimerkin avulla. Esimerkissä valittiin karttakuva jostakin saatavilla olleesta lähteestä. Esimerkissä käytettiin käsin piirrettyä mustavalkoista hahmotelmaa Pohjois-Koreaan sijoittuvasta Wonsanin

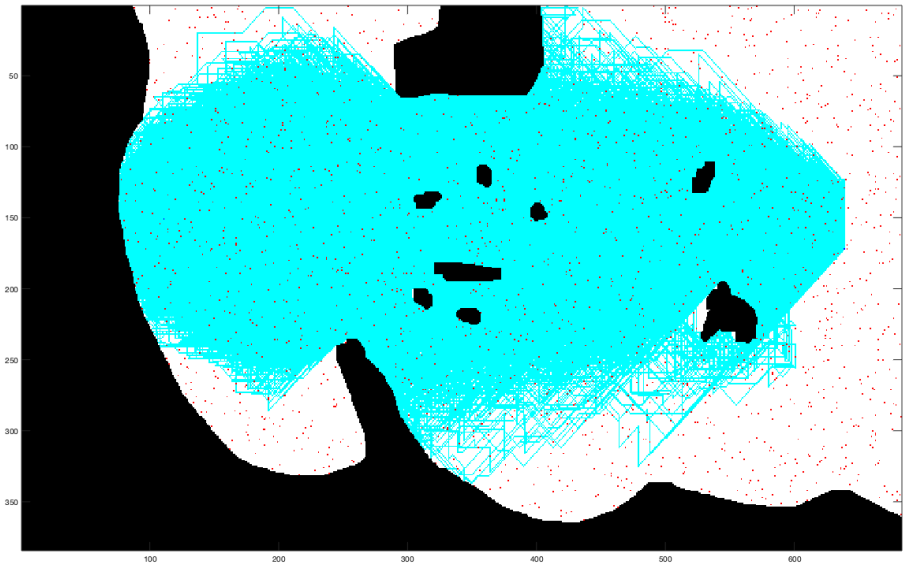
alueesta 4, jonka Korean sodan aikaista satama-alueen vesistön miinoittamista voidaan pitää onnistuneena, sillä se viivytti Yhdysvaltojen maihinnousua yli viikolla (Melia, 1991).

Kuva 4: Mustavalkoinen hahmotelma Wonsanin alueesta (Google).

Kuva muutettiin matriisiksi ja kuvasta valittiin kaksi satunnaista pistettä, joista toinen sijoittui lähelle rantaa ja toinen avomerelle. Vesistöön generoitiin satunnainen miinoite, jonka tehokkuutta arvioitiin. Todellisuudessa arvioitava miinoite olisi esimerkiksi asiantuntijan suunnittelema satunnaisesti generoidun miinoitteen sijaan. Esimerkkitarkastelussa luotiin satunnainen miinoite MATLAB-funktiolla MiinaGen (liite A) asettamalla miinojen lukumääräksi 2000 kpl, jokaisen miinan räjähdeaineen massaksi 650 kg, herätteen kantamaksi satunnaisluku 20-40 m väliltä ja miinan syvyydeksi satunnaisluku 1-20 m väliltä. Esimerkkitarkastelussa asetettiin pääohjelmana toimivalle funktiolle (liite D) aluksen raivaussäteeksi 30 m, raivaussyvyydeksi 10 m, yhden pikselin skaalaksi 20 m, jonka jälkeen generoitiin 10 000 neljän reitin sarjaa miinoitteen läpi. Ohjelma toimii siten, että yhden sarjan aikana edellisillä reiteillä räjähtäneet ja raivatut miinat poistetaan miinoitteesta.

Tämän jälkeen ajettiin funktio, joka generoi reitit ja suorittaa reittien miinakontakteihin liittyvän laskennan. Suoritukseen kului aikaa noin 5 200 sekuntia eli lähes puolitoista tuntia. Liian suuri laskenta-aika voi rajoittaa ohjelman hyödyntämistä taktisessa toiminnassa. Esimerkissä käytetty kartta ja miinoite ovat kooltaan pieniä. Toisen maailmansodan aikana Suomenlahdelle laskettiin noin 70 000 miinaa, joista noin 12 000 miinaa oli suomalaisten laskemia (Reinbach, 2007). Tarkkojen tuloksien arviointi kyseiselle alueelle ja miinoitteelle vaatisi vielä suuremman laskenta-ajan. Funktio tulostaa kuvan 5 vesistöstä, joka sisältää reiteillä ylikuljetut pikselit ja miinojen sisältämät pikselit. Miinoitteen tehokkuutta arvioivina mittareina funktio

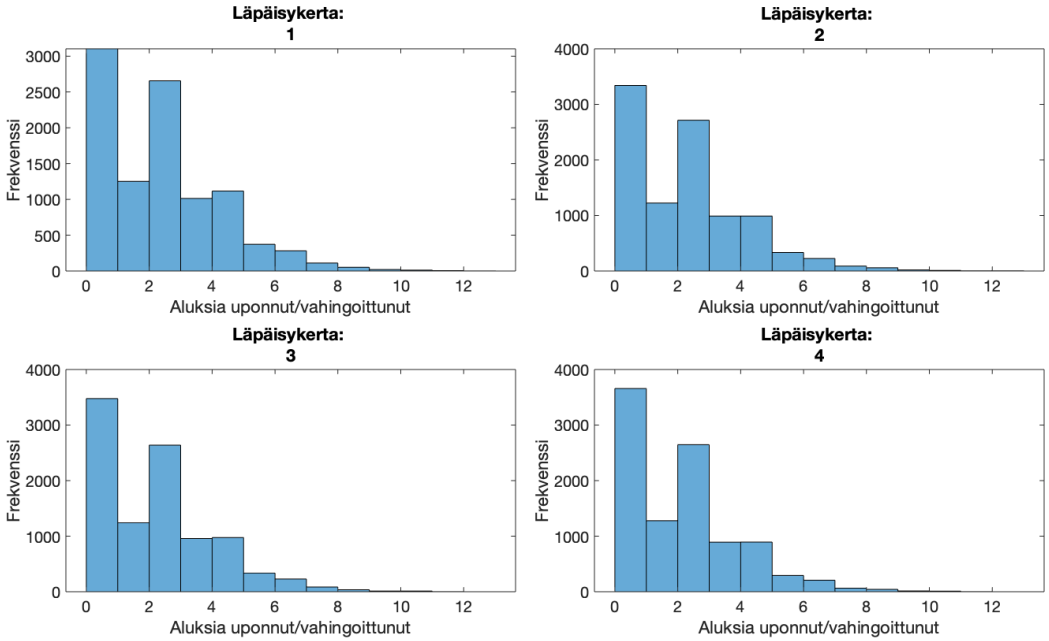
tulostaa kullakin miinoitteen läpäisykerralla koettujen vahinkokategorioiden esiintymien keskiarvon, raivattujen miinojen keskiarvon ja Simple Initial Threatin (SIT) eli todennäköisyyden sille, että ensimmäinen miinoitteen läpäisevä alus vahingoittuu tai uppoaa. Laskennan tulokset ovat taulukossa 3. Lisäksi funktio tulostaa uhrijakauman kullekin läpäisykerralle, joita on esitelty kuvassa 6. Taulukosta 3 havaitaan, että SIT, raivattujen miinojen lukumäärä ja kaikki vahinkokategorioiden esiintyvyydet laskevat läpäisykertojen kasvaessa, mikä on ymmärrettävää, sillä miinoitteesta poistetaan aiemmilla reiteillä räjähtäneet ja raivatut miinat. Reittien väliset erot ovat kuitenkin suhteellisen pieniä johtuen miinoitteen miinojen lukumäärästä, joka on verrattain suuri reiteillä keskimäärin esiintyviin miinakontakteihin nähden. Lisäksi reiteillä ei esiinny vahinkokategorioiden 0 ja 0.25 räjähdyksiä, mikä johtuu siitä, että miinojen herätteen kantama oli lyhyt verrattuna räjähdeaineen massaan. Tämä tulos voidaan tulkita miinoitteelle positiiviseksi, sillä jokainen räjähtänyt miina aiheutti vähintään alusta merkittävästi vahingoittaneen paineiskun.




Kuva 5: 10 000 generoitua reittiä Wonsanin satama-alueen vesistössä. Ylikuljetut pikselit on värjätty turkoosilla ja miinan sisältävät pikselit punaisella. Yhden pikselin leveys kuvassa vastaa 20 metriä.

Taulukko 3: Neljän reitin sarjan miinakontaktien keskiarvo jaoteltuna vahinkokategorioihin ja raivauksiin.

| **Läpäisykerta** | $D = 0$ | $D = 0.25$ | $D = 0.5$ | $D = 0.75$ | $D = 1$ | *raivatut* | SIT |
|---|---|---|---|---|---|---|---|
| **1** | 0 | 0 | 0.7674 | 1.0342 | 0.1197 | 11.4016 | 0.6897 |
| **2** | 0 | 0 | 0.7199 | 0.969 | 0.1156 | 10.7733 | 0.6659 |
| **3** | 0 | 0 | 0.6961 | 0.94255 | 0.1075 | 10.2862 | 0.6523 |
| **4** | 0 | 0 | 0.6623 | 0.8954 | 0.0979 | 9.6848 | 0.6343 |




Kuva 6: Miinoitteen uhrijakauma neljällä reitillä kun aikaisemmilla reiteillä räjähtäneet ja raivatut miinat ovat poistettu miinoitteesta. Kussakin histogrammissa on 10 000 simulaatiota.

# 6 Yhteenveto

Työssä tutustuttiin erilaisiin merimiinatyyppeihin, miinanraivaukseen ja merimiinoitteiden mallintamisen historiaan. Lisäksi työn tavoitteena oli luoda ohjelmisto, jolla pystytään arvioimaan miinoituksen tehokkuutta käyttäjän valitsemalla kartalla. Ohjelmiston miinojen ja alusten välisiä vuorovaikutuksia varten luotiin malli, jonka pohjana käytettiin Uncountered Minefield Planning Model (UMPM) -mallia, jossa miinan ja aluksen välinen etäisyys otetaan huomioon miinan räjähdyksen aiheuttamaa vahinkoa arvioitaessa. Arviointimenetelmänä käytettiin Monte Carlo -simulaatiota, jossa generoitiin suuri määrä reittejä miinoitetussa vesistössä. Ohjelmisto generoi reitit yksinkertaisella algoritmilla, joka valitsee satunnaisen suunnan ja askelmäärän suosien suuntia, jotka lyhentävät etäisyyttä määränpäähän. Miinan

räjähtäessä aluksen kokemaa vahinkoa arvioitiin räjähdyksen tuottaman paineiskun ja kaasukuplan avulla.

Ohjelmiston käyttäjä voi arvioida vapaasti valittavalle alueelle suunniteltua miinoitetta syöttämällä ohjelmistolle listan parametreja, johon kuuluu haluttu karttakuva, lista reitin alku-, väli- ja loppupisteistä, lista miinoista, aluksen raivaussäde ja -syvyys sekä generoitavien reittien lukumäärä. Ohjelmisto tulostaa karttakuvan, johon on piirretty generoidut reitit ja miinojen sijainnit pikselin tarkkuudella. Karttakuva mahdollistaa silmämääräisen tarkastelun, jolla voidaan varmistaa, onko miinojen sijainti haluttu ja onko generoitujen reittien hajonta halutun suuruinen. Lisäksi ohjelmisto tulostaa kullakin miinoitteen läpäisykerralla koettujen vahinkokategorioiden esiintymien keskiarvon, raivattujen miinojen keskiarvon, Simple Initial Threat -tunnusluvun ja uhrijakaumaa kuvaavan histogrammin. Kyseiset tulosteet tarjoavat monipuolista tietoa miinoitteen tehokkuudesta.

Miinoitteen tehokkuuden arviointi ei ole yksiselitteistä ja siksi käytettäviä tehokkuusmittareita on useita. Lisäksi Monte Carlo -simulaatiossa on punnittava, mikä on riittävä iteraatioiden lukumäärä. Iteraatioiden lukumäärän kasvattaminen parantaa tehokkuusmittarien tarkkuutta, mutta samalla laskenta-aika kasvaa. Testauksessa miinoituksen tehokkuuden arviointi osoittautuikin laskennallisesti raskaaksi, kun arviointimenetelmänä käytettiin Monte Carlo -simulaatiota. Ohjelmaa voisi parantaa muodostamalla miinojen herätekäyrät, jotka noudattavat fysikaalisia ilmiöitä, kuten esimerkiksi ääniaallon etenemistä vedessä. Tällöin voitaisiin määritellä aluksille ominaiset herätejäljet, jolloin simulaatio voitaisiin suorittaa useilla alustyypeillä.

# Viitteet

[Reinbach, 2007] Indrek Reinbach, Miinantorjunnan nykytila ja kehityssuunnat Itämerellä, Pro gradu -tutkielma, Maanpuolustuskorkeakoulu, Helsinki, 2007

[Romberger, 1996] David D Romberger, Optimization methods for mixed minefield clearance, Master's Thesis, Naval Postgraduate School, Monterey California, 1996

[Levie, 1992] Howard S. Levie, Mine Warfare at Sea, Martinus Nijhoff Publishers, Netherlands, 1992

[Washburn ja Kress, 2009] A. Washburn ja M. Kress, Combat Modeling, Springer Science & Business Media, United States, 2009

[Washburn, 1995] A. Washburn, MIXER: A TDA for Mixed Minefield Clearance, Naval Postgraduate School, 1995

[Timothy ja Floore, 2011] T. E. Floore ja G. H. Gilman, Design and Capabilities of an Enhanced Naval Mine Warfare Simulation Framework., Proceedings of the 2011 Winter Simulation Conference

[Redmayne, 1996] J.C.J Redmayne, Evaluation of Mine Threat, SACLANT Undersea Research Centre, 1996

[Szturomski, 2015] B. Szturomski, The Effect of an Underwater Explosion on a Ship, Scientific Journal of Polish Naval Academy, volyymi 201, numero 2, sivut 57-73, 2015

[Merisotilaan käsikirja, 2010] Merivoimien esikunta, Merisotilaan käsikirja, Edita Prima Oy, Helsinki, 2010

[Kiciński ja Szturomski, 2020] R. Kiciński ja B. Szturomski, Pressure Wave Caused by Trinitrotoluene (TNT) Underwater Explosion—Short Review, Applied Sciences, volyymi 10, numero 10, sivu 3433, 2020
`https://doi.org/10.3390/app10103433`

[Keil, 1961] A.H. Keil, The Response of Ships to Underwater Explosions, Reprint of Paper Presented at The Annual Meeting, Society of Naval Architects and Marine Engineers, 1961

[Geers ja Hunter, 2001] T. L. Geers ja K. S. Hunter, An integrated wave-effects model for an underwater explosion bubble, Department of Mechanical Engineering, University of Colorado, Colorado, 2001

[Kumar ja Verma, 2010] T. Kumar ja K. Verma, A Theory Based on Conversion of RGB image to Gray image, International Journal of Computer Applications, volyymi 7, numero 2, sivut 7-10, 2010

[Google] Hahmotelman luomisessa käytetty kuva Wonsanista, Google Maps, https://www.google.fi/maps/place/Wŏnsan,+Kangwŏn,+Pohjois-Korea/@39.2354021,127.4603398,12.31z/data=!4m5!3m4!1s0x5fd7056d015e34fd:0x547c2226d8a09bd5!8m2!3d39.1538529!4d127.4460001

[Melia, 1991] T. Melia, Damn the Torpedoes: A Short History of U.S. Naval Mine Countermeasures, 1777-1991, Contributions To Naval History, 4. painos, Naval Historial Center, Department of the Navy, 1991

# Liitteet

## A) MiinaGen

```
function generator = miinaGen(lkm, vesisto, rajahdekg,
    herateRange, ...
     maxSyvyys)
     maxY=size(vesisto,1);
     maxX=size(vesisto,2);
     % Muutetaan kuva mustavalkoiseksi
     for i = 1:maxY
          for j = 1:maxX
               if max(vesisto(i,j,:)) < 128 % 0 on musta, 255
                  valkoinen
                    vesisto(i,j,:) = [0,0,0]; %musta
               else
                    vesisto(i,j,:) = [255,255,255]; %valkoinen
               end
          end
     end
     i = 1;
     output = zeros(0,5);
     while i <= lkm
          randY = randi([1,maxY],1)+rand;
          randX = randi([1,maxX],1)+rand;
          randZ = randi([1, maxSyvyys],1)+rand;
          if vesisto(floor(randY),floor(randX)) ~= 0
               herate = randi([herateRange(1),herateRange(2)
                  ],1);
               output(i,:) = [randY,randX,randZ, rajahdekg,
                  herate];
               i = i + 1;
          end
     end
     generator = output;
end
```

## B) reittiGen

```
function path = reittiGen(coordinate_list, map)
%neighbors
maxY=size(map,1);
maxX=size(map,2);
list = zeros(0,2);
listLength = size(coordinate_list,1);
stepper = 1;
```

```
8  index = 1;
9  while stepper < listLength
10     currenty = coordinate_list(stepper, 1);
11     currentx = coordinate_list(stepper, 2);
12     endy = coordinate_list(stepper+1, 1);
13     endx = coordinate_list(stepper+1, 2);
14     while pdist([currenty,currentx;endy,endx], 'euclidean')
          ~= 0
15 %         ydifference = endy-currenty;
16 %         xdifference = endx-currentx;
17
18         dirMat = zeros(8,2);
19         dirMat(1,:) = [currenty-1,currentx];%north
20         dirMat(2,:) = [currenty-1,currentx+1];%northeast
21         dirMat(3,:) = [currenty,currentx+1];%east
22         dirMat(4,:) = [currenty+1,currentx+1];%southeast
23         dirMat(5,:) = [currenty+1,currentx];%south
24         dirMat(6,:) = [currenty+1,currentx-1];%southwest
25         dirMat(7,:) = [currenty,currentx-1];%west
26         dirMat(8,:) = [currenty-1,currentx-1];%northwest
27         %check suitable directions
28         i = 1;
29         %valitaan feasiblet
30         while i <= size(dirMat,1)
31             if dirMat(i,1)> maxY || dirMat(i,1) < 1 ||
                  dirMat(i,2) > maxX ...
32                 || dirMat(i,2) < 1 || map(dirMat(i,1),dirMat(
                      i,2)) == 0
33                 dirMat(i,:) = [];
34                 i = i - 1;
35             end
36             i = i + 1;
37         end
38
39         %jaotellaan feasiblet matkaa lyhentaviksi ja matkaa
              ei lyhetaviksi
40         dirMat2 = zeros(0,2);
41         i = 1;
42         j = 1;
43         while i <= size(dirMat,1)
44             if pdist([dirMat(i,1),dirMat(i,2);endy,endx]) >=
                   ...
45                     pdist([currenty,currentx;endy,endx])
46                 dirMat2(j,:) = dirMat(i,:);
47                 j = j + 1;
```

```
48                      dirMat(i,:) = [];
49                      i = i - 1;
50                  end
51                  i = i + 1;
52              end
53          % Suositaan matkaa lyhentavia suuntia
54          if size(dirMat,1) > 0
55              nS = dirMat(randi([1,size(dirMat,1)]),:);
56          else
57              nS = dirMat2(randi([1,size(dirMat2,1)]),:);
58          end
59          % Valtetaan siirtymista suuntaan, josta ollaan saavuttu
60          if index > 2
61              t = 1;
62
63              while (pdist([nS(1),nS(2);list(index-2,1),list(index
                    -2,2)]) == 0)...
64                  && t < 20
65                  if size(dirMat,1) > 1
66                      nS = dirMat(randi([1,size(dirMat,1)]),:);
67                  else
68                      nS = dirMat2(randi([1,size(dirMat2,1)]),:);
69                  end
70                  t = t + 1;
71              end
72          end
73
74      d = nS - [currenty,currentx];
75      sameDirection = 0;
76      maxSameDirection = randi([1,30]);
77      clutch = 0;
78      while ( sameDirection < maxSameDirection && (currenty + d(1)
            )...
79            < maxY && (currenty + d(1)) > 1 && (currentx + d(2))
                ...
80            < maxX && (currentx + d(2)) > 1 && ...
81            map(currenty + d(1),currentx + d(2)) ~= 0 ...
82            && ((pdist([endy,endx;currenty,currentx]) >= ...
83            (maxSameDirection-sameDirection))) )
84          currenty = currenty + d(1);
85          currentx = currentx + d(2);
86          list(index,:) = [currenty,currentx];
87          index = index + 1;
88          sameDirection = sameDirection + 1;
89          clutch = 1;
```

```
90  end
91  if clutch == 0
92      currenty = currenty + d(1);
93      currentx = currentx + d(2);
94      list(index,:) = [currenty,currentx];
95      index = index + 1;
96      sameDirection = sameDirection + 1;
97  end
98
99
100     end
101
102
103 stepper = stepper + 1;
104 end
105 path = list;
106 end
```

### C) mineCalculation

```
1  function [vahinko, raivaukset] = mineCalculation(reitti,
       miina, raivaus, ...
2      skaala)
3      % miinan params: x,y,z-koords,rajahdekg, heratteen
           kantama
4      tiheys = 1520;
5      vahinko = [0,0,0,0,0]; %0, 0.25, 0.5, 0.75, 1;
6      raivaukset = 0;
7      reittiS = reitti;
8      reittiS(:,3) = 1; %reitin syvyys = 1
9      %reitin ja miinan i pienin etaisyys
10     [k, dist] = dsearchn(reittiS, miina(1:3));
11     [s, dist2d] = dsearchn(reitti, miina(1:2));
12     if k > 1 && k < size(reitti,1)
13             ts = reitti(k,:)-reitti(k-1,:);
14             ms = reitti(k+1,:)-reitti(k,:);
15             ap = reitti(k,:);
16             % Maaritetaan pikselin ylityksen alkupiste
17             if ts(1) == -1
18                 if ts(2) == -1
19                     ap = reitti(k,:) + [1,1];
20                 end
21                 if ts(2) == 0
22                     ap = reitti(k,:) + [1,1/2];
23                 end
```

```
24              if ts(2) == 1
25                  ap = reitti(k,:) + [1,0];
26              end
27          end
28
29          if ts(2) == 0
30              if ts(2) == -1
31                  ap = reitti(k,:) + [1/2, 1];
32              end
33              if ts(2) == 1
34                  ap = reitti(k,:) + [1/2, 0];
35              end
36          end
37
38          if ts(1) == 1
39              if ts(2) == -1
40                  ap = reitti(k,:) + [0, 1];
41              end
42              if ts(2) == 0
43                  ap = reitti(k,:) + [0, 1/2];
44              end
45              if ts(2) == 1
46                  ap = reitti(k,:) + [0, 0];
47              end
48          end
49
50          % Maaritetaan pikselin ylityksen loppupiste
51          lp = reitti(k,:);
52
53          if ms(1) == -1
54              if ms(2) == -1
55                  lp = reitti(k,:) + [0,0];
56              end
57              if ms(2) == 0
58                  lp = reitti(k,:) + [0,1/2];
59              end
60              if ms(2) == 1
61                  lp = reitti(k,:) + [0,1];
62              end
63          end
64
65          if ms(2) == 0
66              if ms(2) == -1
67                  lp = reitti(k,:) + [1/2, 0];
68              end
```

```
69                  if ms(2) == 1
70                      lp = reitti(k,:) + [1/2, 1];
71                  end
72              end
73
74              if ms(1) == 1
75                  if ms(2) == -1
76                      lp = reitti(k,:) + [1, 1];
77                  end
78                  if ms(2) == 0
79                      lp = reitti(k,:) + [1, 1/2];
80                  end
81                  if ms(2) == 1
82                      lp = reitti(k,:) + [1, 0];
83                  end
84              end
85
86              suunta = lp-ap;
87              reittiPikselissa = zeros(skaala+1,2);
88              j = 1;
89
90              for step = 0:1/skaala:1
91                  reittiPikselissa(j,:) = ap + step*suunta;
92                  j = j+1;
93              end
94
95              reittiPikselissa(:,3) = 1;
96              [k, dist] = dsearchn(reittiPikselissa, miina
                    (1:3));
97              [s, dist2d] = dsearchn(reittiPikselissa(:,1:2),
                    miina(1:2));
98      end
99                  paine = 0;
100                 if (skaala*dist) <= miina(5) %miina rajahtaa
101                     Rpmax = 1.53*(miina(4)/(1+0.1*miina(3)))
                            ^(1/3);
102                     if (skaala*miina(5)) <= Rpmax
103                         vahinko(5) = 1;
104
105                         a = (3*miina(4)/(4*pi*tiheys))^(1/3)
                                ;
106                         paine = 10*1420*(a/miina(5))^1.13;
107                     else
108                         a = (3*miina(4)/(4*pi*tiheys))^(1/3);
109                         paine = 1420*(a/miina(5))^1.13;
```

```
110                         end
111                         if paine >= 16
112                             vahinko(5) = 1;
113                         end
114                         if  paine < 16
115                             vahinko(4) = 1;
116                         end
117                         if paine < 12 && paine >= 8
118                             vahinko(3) = 1;
119                         end
120                         if paine < 8 && paine >= 4
121                             vahinko(2) = 1;
122                         end
123                         if paine < 4
124                             vahinko(1) = 1;
125                         end
126                     end
127 
128                     if paine == 0
129                         if(skaala*dist2d) <= raivaus(1) && miina
                               (3) < raivaus(2)
130                             raivaukset = 1;
131                         end
132                     end
133 
134 end
```

## D) minefieldSim

```
1  function kApp = minefieldSim(lista, miinat, raivaus, kuva,
       lapaisykerrat, ...
2      reititlkm,skaala)
3      vesisto = imread(kuva);
4      maxY=size(vesisto,1);
5      maxX=size(vesisto,2);
6      % Muutetaan kuva mustavalkoiseksi
7      for i = 1:maxY
8          for j = 1:maxX
9              if max(vesisto(i,j,:)) < 128 % 0 on musta, 255
                   valkoinen
10                 vesisto(i,j,:) = [0,0,0]; %musta
11             else
12                 vesisto(i,j,:) = [255,255,255]; %valkoinen
13             end
14         end
```

```
15      end
16
17      kytkin = 0;
18      %varmistetaan etta pisteet ja miinat kuvassa
19      if min(lista(:,1)) < 1 || max(lista(:,1)) > maxY...
20          || min(lista(:,2)) < 1 || max(lista(:,2)) > maxX...
21          || floor(min(miinat(:,1))) < 1 || floor(max(miinat
                (:,1))) > maxY...
22          || floor(min(miinat(:,2))) < 1 || floor(max(miinat
                (:,2))) > maxX...
23          kytkin = 1;
24          disp("Valitut pisteet tai miinat ovat koordinaatiston
                ulkopuolella")
25      end
26
27      if kytkin == 0
28          %varmistetaan etta pisteet vesistossa
29          for i = 1:size(lista,1)
30              if vesisto(lista(i,1),lista(i,2)) == 0
31                  kytkin = 1;
32                  disp("Jokin valituista pisteist? on
                        rajoitetulla alueella")
33              end
34          end
35      end
36
37      if kytkin == 0
38          %varmistetaan etta miinat vesistossa
39          for i = 1:size(miinat,1)
40              if vesisto(floor(miinat(i,1)),floor(miinat(i,2))
                    ) == 0
41                  [miinat(i,1),miinat(i,2)]
42                  kytkin = 1;
43                  disp("Jokin miinoista on rajoitetulla
                        alueella")
44              end
45          end
46      end
47
48      % Jos pisteet ja miinat ok, voidaan generoida reitit ja
            laskea
49      % miinakontaktit
50      if kytkin == 0
51          vesistoOut = vesisto;
52
```

```
53 %         % Lasketaan reittien vaarallisuus ottaen huomiion
       aikaisemmat
54 %         % lapaistut reitit ja niilla putsatut miinat
55           laskelmatReps = zeros(lapaisykerrat,6);
56           SIT = zeros(lapaisykerrat,1);
57           CD = zeros(reititlkm,lapaisykerrat);
58           for s = 1:reititlkm
59            miinatMod = miinat;
60            laskelmat = zeros(lapaisykerrat,6);
61            for i = 1:lapaisykerrat
62                 reitti = reittiGen(lista,vesisto);
63                 if i == 1
64                      for j = 1:length(reitti)%Merkitaan reitit
                           kuvaan turkoosilla
65                           vesistoOut(reitti(j,1),reitti(j,2)) = 1;
66                      end
67                 end
68
69                 % lasketaan reitin keskiarvoistetut
                      miinakontaktit ja
70                 % poistetaan
71                 % rajahtaneet/raivatut
72                 j = 1;
73                 while j <= size(miinatMod,1)
74                      [vahinko, raivatut] =...
75                        mineCalculation(reitti, miinatMod(j,:),
                             raivaus,skaala);
76                      if max(vahinko) ~= 0 || raivatut ~= 0
77                         miinatMod(j,:) = [];
78                         laskelmat(i,:) = laskelmat(i,:) + [
                              vahinko, raivatut];
79                         j = j - 1;
80                      end
81                      j = j + 1;
82                 end
83                 if sum(laskelmat(i,2:5)) > 0
84                        SIT(i) = SIT(i) + 1;
85                 end
86                 CD(s,i) = sum(laskelmat(i,2:5));
87            end
88            laskelmatReps = laskelmatReps + laskelmat;
89 %           if i == 1
90 %               CD(s,i) = sum(laskelmat(i,2:5));
91 %           end
92           end
```

```
93          %SIT
94          SIT = SIT/reititlkm;
95          reitti = [1:lapaisykerrat]';
96          laskelmatReps = laskelmatReps/reititlkm;
97          tulokset = [reitti, laskelmatReps];
98          %disp(["SIT: ", SIT])
99          disp(["L p isykerta", "ei vahinkoa KA","vahinko
               0.25 KA",...
100              "vahinko (0.5) KA","vahinko (0.75) KA","vahinko
                    (1) KA",...
101              "raivatut KA","SIT";tulokset,SIT])
102         for j = 1:length(miinat) % Merkitaan miinat kuvaan
               punaisella
103             vesistoOut(floor(miinat(j,1)),floor(miinat(j,2))
                    ,:) = [255,0,0];
104         end
105         for j = 1:length(lista) % Merkitaan pisteet kuvaan
               sinisella
106             vesistoOut(lista(j,1),lista(j,2),:) = [0,0,255];
107
108         end
109         image(vesistoOut);
110         disp(["Alku- ja loppupisteen valinen etaisyys (m)
                ",...
111             skaala*pdist([lista(1,1),lista(1,2);lista(end,1),
                    lista(end,2)])])
112         disp(["Miinojen lkm: ", length(miinat)])
113         figure;
114         t = tiledlayout(ceil(lapaisykerrat/2),2); % Requires
               R2019b or late
115         for i = 1:lapaisykerrat
116             nexttile
117             histogram(CD(:,i), [0:1:max(CD, [], 'all')] );
118             xlabel('Aluksia uponnut/vahingoittunut','
                    FontSize',30)
119             ylabel('Frekvenssi','FontSize',30)
120             title(["L p isykerta: ", i],'FontSize',30);
121             set(gca,'FontSize',20)
122         end
123     end
124 end
```