Todellisuusraon metodit ja niiden kehitys

Turun yliopisto
Tietotekniikan laitos
TkK-tutkielma
Tietotekniikka
Marraskuu 2025
Anton Sebastian Teerioja

Turun yliopiston laatujärjestelmän mukaisesti tämän julkaisun alkuperäisyys on tarkastettu Turnitin OriginalityCheck -järjestelmällä.

TURUN YLIOPISTO
Tietotekniikan laitos

Anton Sebastian Teerioja: Todellisuusraon metodit ja niiden kehitys

TkK-tutkielma, 26 s.
Tietotekniikka
Marraskuu 2025

Simulaatioympäristöissä koulutetut vahvistusoppimisalgoritmit tarjoavat ratkaisun moneen käytännön ongelmaan. Näitä ovat monet robotiikan ongelmat kuten esineiden käsittely ja polunetsintä. Muita mahdollisia käyttökohteita ovat esimerkiksi sähköverkkojen hallinta ja kyberturva. Simulaatio on halvempi ja nopeampi tapa kouluttaa algoritmeja verrattuna perinteiseen todellisessa ympäristössä kouluttamiseen. Useita simulaatioita voidaan kouluttaa samanaikaisesti ja simulaatiossa robotit eivät vaurioidu tai kulu koulutuksen aikana. Koulutuksen siirtäminen simulaatiosta todellisuuteen ei kuitenkaan ole yksinkertaista. Siihen liittyy monia ongelmia, joista tärkein on todellisuusrako.
Todellisuusrako tarkoittaa eroa todellisuuden ja simulaation välillä, ja siitä aiheutuvia ongelmia koulutuksen siirtämiseen. Tässä tutkielmassa käsitellään todellisuusraon metodeja kuten toimintaympäristön satunnaistamista ja -mukauttamista sekä näiden metodien kehitystä. Kehityksen tavoitteena on parantaa näiden metodien toimintaa, nopeuttaa koulutusta ja madaltaa koulutuksen kustannuksia. Todellisuusraon metodit mahdollistavat monimutkaisten ongelmien ratkaisun moniulotteisissa ympäristöissä vahvistusoppimisen avulla. Näiden menetelmien todelliset hyödyt näkyvät erityisesti robotiikan käyttökohteissa. Robotit pystyvät suorittamaan monimutkaisia tehtäviä muuttuvissa ympäristöissä, ilman ihmisen ohjausta. Tämän lisäksi menetelmien kehitys mahdollistaa vahvistusoppimisen hyödyntämisen käyttökohteissa, joissa sitä ei ole vielä hyödynnetty.

Asiasanat: todellisuusrako, vahvistusoppiminen, simulaatioympäristöt

Sisällys

1 Johdanto **1**

2 Tausta **3**

2.1 Vahvistusoppiminen 3

2.1.1 Vahvistusoppimisen formalisointi 5

2.1.2 Vahvistusoppimisalgoritmit 6

2.2 Simulaatioympäristöt 9

2.3 Todellisuusraon historia 9

3 Todellisuusraon metodit **13**

3.1 Toimintaympäristön satunnaistaminen 13

3.2 Toimintaympäristön mukauttaminen 14

3.3 Simulaatioympäristöjen suunnittelu 15

4 Todellisuusraon metodien kehitys **17**

4.1 Vahvistusoppimisalgoritmien kehitys 17

4.1.1 Taktiili TD3 18

4.1.2 Hierarkkinen vahvistusoppiminen 19

4.2 Yleisten metodien kehitys 20

4.2.1 Bayesilainen satunnaistaminen 21

4.2.2 Kalman-suotimeen perustuva siirto-oppiminen 22

4.2.3 CAPTURE 23

5 Yhteenveto **24**

Lähdeluettelo **27**

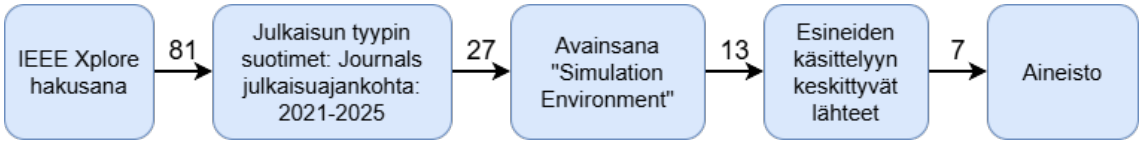
1 Johdanto

Tekoälyn käyttö on vahvassa nousussa. Yksi sen tärkeimmistä käyttökohteista on robotiikka. Siinä tekoälyä käytetään ratkaisemaan ongelmia, jotka ovat liian monimutkaisia perinteisille ohjelmointimenetelmille ja -algoritmeille. Vahvistusoppiminen on koneoppimisen metodi, joka keskittyy algoritmin oppimiseen sen ympäristöstä ilman ihmisen ohjausta. Robottien kouluttaminen todellisuudessa on kallista ja riskialtista erityisesti roboteille itselleen. Halvempi vaihtoehto on kouluttaa robotti simulaatiossa, ja siirtää oppiminen todelliseen robottiin.

Tämä tutkielma käsittelee todellisuusrakoa (engl. sim-to-real gap), joka on simulaatiossa koulutetun vahvistusoppimisen ongelma. Se on ajankohtainen ongelma, koska robotiikka on siirtymässä perinteisistä ennalta ohjelmoiduista roboteista dynaamisissa ympäristöissä toimiviin robotteihin. Näissä ympäristöissä ei voida luottaa muuttumattomiin algoritmeihin, vaan tarvitaan ympäristöihin mukautuvia algoritmeja.

Aihe valikoitui mielenkiintoiseksi ja ajankohtaiseksi aineistotutkimuksen aikana. Tutkielmassa on kaksi tutkimuskysymystä. *TK1 on mitä ovat todellisuusraon metodit?* Kysymykseen liittyen esitellään muutama nykyinen ja vakiintunut todellisuusraon metodi. *TK2 on miten todellisuuraon metodeja on kehitetty?* Metodeja käsitellään vahvistusoppimisen ja todellisuusraon näkökulmista. Metodeista esitellään niiden toimintaperiaatteet, hyödyt sekä ongelmat.

Aineisto on haettu IEEE Xplore tietokannasta, joka keskittyy: elektroniikan ja tekniikan julkaisuihin, konferenssiartikkeleihin ja muihin teknillisiin dokumentteihin. Hakulause rakennettiin aihepiirissä käytetystä teknisestä sanastosta, jotta aineisto olisi mahdollisimman tarkasti tähän tutkielmaan liittyvää. Käytetty hakulause on *("simulation based reinforcement learning" OR ("reinforcement learning" AND "sim-to-real") AND "parameters")*. Kuvassa 1.1 esitetään aineiston hakemisen ja rajauksen prosessi.



Kuva 1.1: Lähteiden rajaamisen prosessi.

Ensimmäisen haun jälkeen aineistoa on rajattu IEEE Xploren omilla suotimilla aineiston tyypin ja sen julkaisuvuoden sekä avainsanahaun mukaan. Jäljelle jääneistä aineistoista valittiin ne, jotka keskittyivät esineiden käsittelyyn. Lopullisten aineistojen vertailu toisiinsa on järkevää, koska ne pyrkivät löytämään ratkaisun samanlaiseen ongelmaan.

Tämän tutkielman rakenne on seuraavanlainen. Toisessa luvussa käsitellään vahvistusoppimista, simulaatioympäristöjä sekä todellisuusraon historiaa ja sen määrittelyä. Kolmannessa luvussa esitellään todellisuusraon nykyiset metodit. Neljännessä luvussa esitellään ja vertaillaan nykyisten metodien kehitystä. Viidennessä luvussa vastataan tutkimuskysymyksiin, ja pohditaan tutkielman sisältöä sekä esitellään havaintoja tutkimuksen teosta.

2 Tausta

Todellisuusraon metodit vaativat hyvää ymmärrystä vahvistusoppimisesta, simulaatioympäristöistä sekä ongelman todellisesta ympäristöstä. Vahvistusoppiminen on valikoitunut simulaatiokoulutukseen parhaiten soveltuvaksi koneoppimisen menetelmäksi, koska se oppii ympäristöstään ilman koulutuksen ohjausta. Simulaatioympäristöt ovat yleensä fysiikkamoottoriin perustuvia tietokoneella simuloituja ympäristöjä, jotka mallintavat todellisuuden rakennetta eri parametrien ja algoritmien avulla.

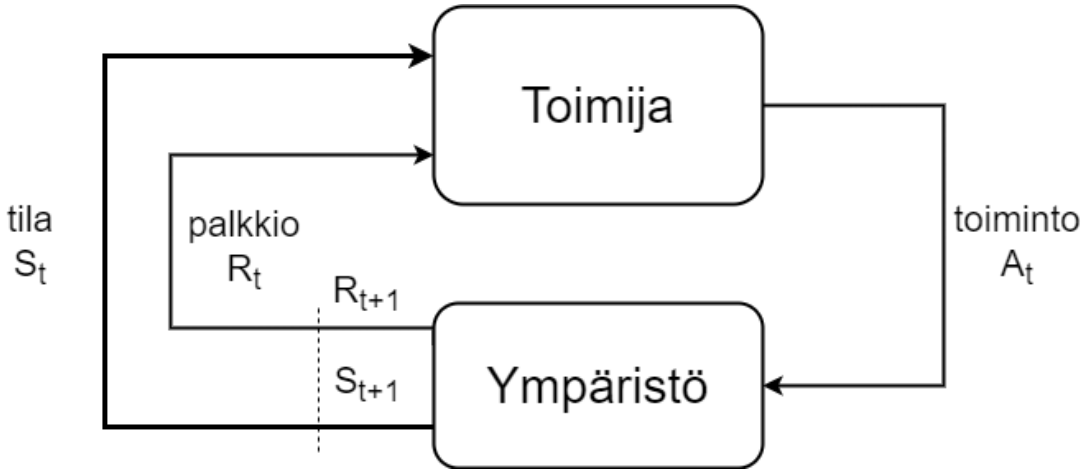
2.1 Vahvistusoppiminen

Vahvistusoppiminen on yksi kolmesta koneoppimisen alasta. Toiset kaksi ovat ohjattu oppiminen ja ohjaamaton oppiminen [1]. Vahvistusoppimiseen liittyy piirteitä kummastakin, koska sille voidaan antaa annotoitua aineistoa, jonka pohjalta se oppii kuten ohjatussa oppimisessa. Sitä koulutetaan myös ilman suoraa ohjausta kuten ohjaamattomassa oppimisessa.

Vahvistusoppiminen on luonteeltaan itsenäisesti oppivaa, eli kouluttaja ei kerro sille ongelman luonteesta, vaan sen pitää oppia hahmottamaan maailmaa, sekä ongelman ratkaisua itse. Sen koulutus toimii iteratiivisesti toistamalla ja muuttamalla algoritmin parametreja koulutuksen edetessä. Tämä mahdollistaa monimutkaisten ongelmien ratkaisun. Toisin kuin ohjatussa ja ohjaamattomassa oppimisessa, vahvistusoppiminen oppii vain ympäristöstään ja omista teoistaan ympäristöstään ja

omista teoistaan [1]. Sille ei kerrota mitään sen ympäristöstä ja sen käyttämä koulutusdata ei ole annotoitua. Kaikki edellä mainitut ominaisuudet tekevät vahvistusoppimisesta hyvän mallin ratkaisemaan ongelmia tuntemattomissa, muuttuvissa, tai monimutkaisissa ympäristöissä [1].

Kuvassa 2.1 esitetään toimijan ja ympäristön välistä suhdetta.



Kuva 2.1: toimijan ja ympäristön suhde.

Kuvassa käytettyjä tärkeitä käsitteitä ovat: *toimija* (engl. actor) päättää toiminnon perustuen ympäristön tilaan ja palkkioon, *toiminto (A)* (engl. action) toimijan tekemä toiminto, joka muuttaa ympäristön tilaa, *ympäristö* (engl. environment) ympäristö, jonka sisällä toimija toimii, *palkkio (R)* (engl. reward) signaali, joka ohjaa toimijan toimintoja perustuen niiden vaikutukseen ympäristössä, *tila (S)* (engl. state) ja *aika-askel (t)* (engl. time step), joka kertoo millä kierroksella koulutus on. Suhde on ketju, jossa toimija valitsee toiminnon. Se vaikuttaa ympäristöön, jonka muutoksen seurauksena valitaan seuraava tila. Tilan muutoksen seurauksena palkkio saa tietyn arvon. Ympäristön tila ja palkkion arvo ohjaavat toimijan seuraavaa toimintoa ja koulutus jatkuu.

Vahvistusoppimista hyödynnetään monessa käyttökohteessa. Sen kehitys on vaatinut laskentatehon kasvua ja kustannuksien laskua. Nykyään monet yksinkertaiset

vahvistusoppimisen tehtävät voidaan kouluttaa yleisesti saatavilla olevilla kaupallisilla prosessoreilla, joka ei ollut mahdollista vielä 2000-luvun alussa.

2.1.1 Vahvistusoppimisen formalisointi

Tämä tutkielma käsittelee vahvistusoppimisalgoritmeja, jotka ratkaisevat vahvistusoppimistehtäviä. Tehtävien tulee täyttää Markovin ominaisuuden (engl. Markov property) ehdot. Markovin ominaisuus tarkoittaa tilasignaalia, joka kertoo kaiken tarpeellisen tiedon ympäristön tilasta [1]. Tämän määritelmän mukaisia algoritmeja kutsutaan Markovin päätösprosesseiksi (engl. Markov Decision Process, MDP) [1]. Kaavassa 2.1 esitetään yleinen muoto Markovin päätösprosessista, joka tarkoittaa seuraavan tilan s' ja palkkion r todennäköisyyttä perustuen nykyiseen tilaan s ja toimintoon a.

$$p(s', r|s, a) = Pr\{S_{t+1} = s', R_{t+1} = r|S_t = s, A_t = a\}. \tag{2.1}$$

Markovin päätösprosessi on määritelty sen tila- ja toimintoavaruuden ja yhden aika-askeleen ympäristön dynamiikalla [1]. Tähän yleiseen muotoon lisätään termejä kuten γ eli alennustermi, jotta erilaisia algoritmeja ja metodeja voidaan käyttää. Kaavassa 2.2 esitetään tutkimuksissa yleisesti käytetty lyhennetty muoto käytetyistä termeistä.

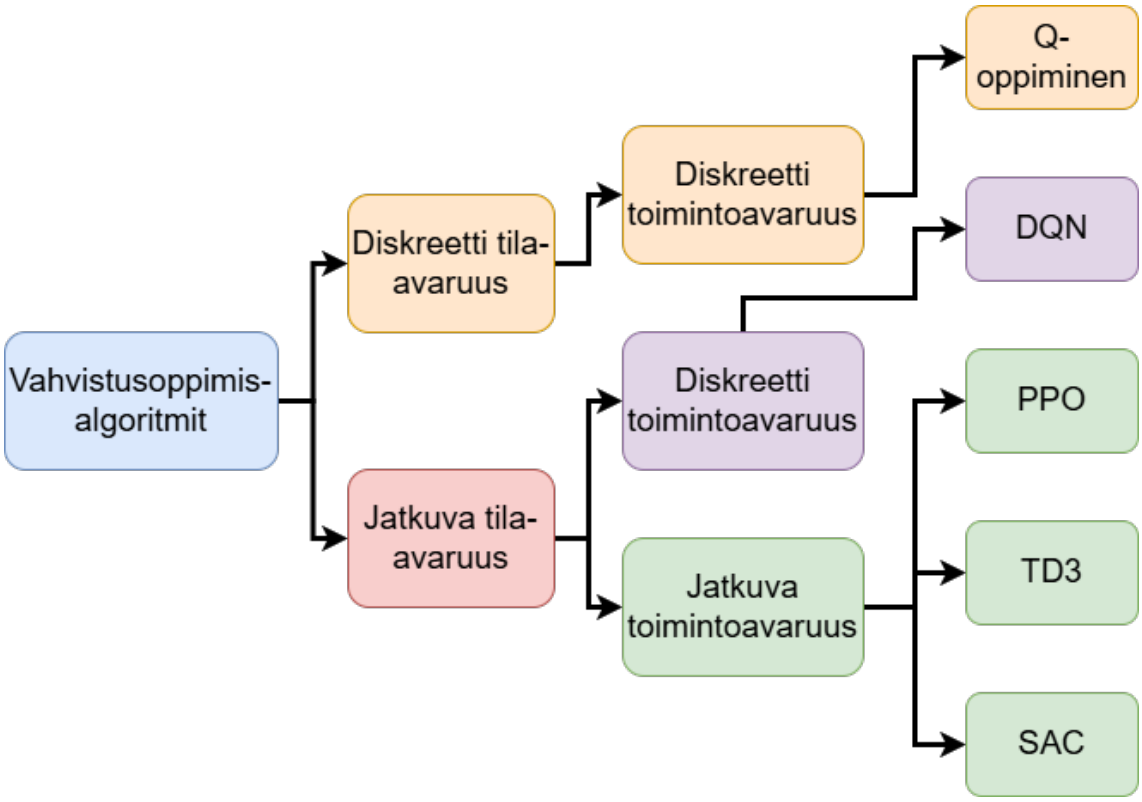
$$\{S, A, R, P, \gamma\} \tag{2.2}$$

tila-avaruus (S) (engl. state-space), *toiminto-avaruus (A)* (engl. action-space), *tilanmuutoksen todennäköisyysfunktio (P)*, *palkkiofunktio (R)* ja *alennustermi (γ)*. Näiden termien yleinen muoto on $s \in S$ kaikki tilat s kuuluvat tila-avaruuteen S. $a \in A$ = kaikki toiminnot a kuuluvat toiminto-avaruuteen A. $P(s'|s, a)$ = Todennäköisyys tilanmuutokselle perustuen nykyiseen tilaan ja toimintoon. $R(s, a, s')$ Palk-

kio määräytyy nykyisen- ja tulevan tilan sekä nykyisen toiminnon mukaan. γ Auttaa tasapainottamaan tutkimista (engl. exploration) ja hyödyntämistä (engl. exploitation). Näiden kaavojen ymmärtäminen mahdollistaa algoritmien tulkinnan.

2.1.2 Vahvistusoppimisalgoritmit

Vahvistusoppimisalgoritmit voidaan jakaa kolmeen kategoriaan niiden toimintaympäristöjen ominaisuuksien mukaan [2]. Nämä kategoriat ovat rajoitettu määrä tiloja ja diskreetit toiminnot, rajoittamaton määrä tiloja ja diskreetit toiminnot sekä rajoittamaton määrä tiloja ja jatkuvat toiminnot. Kuvassa 2.2 esitetään tutkielmassa käsiteltävien vahvistusoppimisalgoritmien luokittelu.



Kuva 2.2: Vahvistusoppimisalgoritmien luokittelua.

Kuvan vahvistusoppimisalgoritmit ovat: Q-oppiminen, Deep Q-network (DQN), Proximal Policy Optimization (PPO), Twin Delayed Deep Deterministic Policy Gra-

dient Algorithm (TD3) ja Soft Actor Critic (SAC). Tämä ei ole kattava lista kaikista vahvistusoppimisalgoritmeista eikä niiden keskinäisistä suhteista.

Rajoitettu määrä tiloja ja diskreetit toiminnot ovat ympäristöinä yksinkertaisia. Niissä toimija voi valita ennalta määritetyistä toiminnoista ja päätyä ennalta tunnettuihin tiloihin. [2] Esimerkiksi ristinolla-pelissä pelaaja voi valita pelilaudan ruuduista yhden vapaan ruudun. Vaikka vaihtoehtoja on monta, kaikki mahdolliset pelitilanteet ovat tunnettuja tiloja, joten tila-avaruus on diskreetti. Näitä keinoja ei käsitellä tässä tutkielmassa, koska ne ovat riittämättömiä kuvaamaan todellisuuden ongelmia.

Rajoittamaton määrä tiloja ja diskreetit toiminnot ovat ympäristöinä monimutkaisempia, joten niitä voidaan hyödyntää todellisuuden käyttökohteissa [2]. Tämän kategorian tärkein algoritmi aineiston perusteella on Deep Q-network. DQN käyttää konvoluutio- ja kasauskerrosta (engl. convolution- and pooling-layer) yhdessä kahden täysin yhdistetyn kerroksen kanssa tuottaakseen Q-arvon, joka kuvaa parasta toimintoa, jonka toimija voi valita. Toimija valitsee sen, ja prosessoi tilan muutoksen sekä palkkiosignaalin. Tämän algoritmin suurin ongelma on sen taipumus yliarvioida Q-arvoa, joka johtaa toimijan valitsemaan huonoja toimintoja.

Rajoittamaton määrä tiloja ja jatkuvat toiminnot ovat kaikista monimutkaisimpia ympäristöjä, mutta kuvaavat yleensä parhaiten todellisia tilanteita. Näissä ympäristöissä toimija oppii parametrisoidun toimintatavan, joka pyrkii maksimoimaan odotetun arvon alennetuista palkkioista, koska on mahdotonta laskea toimintopalkkio arvo kaikille toiminnoille jatkuvassa tila-avaruudessa. [2] Tässä ympäristössä toimivat algoritmit käyttävät yleisesti toimintatapagradientin (engl. policy gradient, pg) metodeja. Ne toimivat laskemalla gradientille arvion toimintatapaa varten, jota optimoidaan stokastisilla gradientin nousumenetelmillä [3]. Tämän kategorian tärkeimmät algoritmit aineiston perusteella ovat: Proximal Policy Optimization (PPO),

Soft Actor Critic (SAC) ja Twin Delayed Deep Deterministic Policy Gradient Algorithm (TD3).

PPO perustuu toimintatapagradienttiin ja rakentaa sen päälle robustimman algoritmin. Se pystyy rajoittamaan arvojen suuruutta, joka johtaa gradientin nousun tuottavan vakaampia ja parempia ratkaisuja. Tämä on syy sille, miksi PPO on yksi eniten käytetyistä vahvistusoppimisalgoritmeista. [3]

SAC perustuu toimija-kriitikko-neuroverkkoihin, jossa toimija pyrkii optimoimaan toiminnot, mutta sen ei tarvitse itse päivittää arvofunktiota. Kriitikko keskittyy arvofunktion päivittämiseen ja ohjaamaan toimijaa matalan varianssin tiedolla. [4] SAC lisää toimija-kriitikko-verkkoon ympäristön entropian maksimoinnin samalla kun arvofunktiota maksimoidaan. Näin algoritmi ei jää jumiin ympäristön lokaaleihin maksimikohtiin vaan pystyy löytämään parhaan arvon. SAC on tehokas ja vakaa algoritmi, joka on myös hyvin laajasti käytetty monimutkaisten ongelmien ratkaisemisessa. [4]

TD3 rakentuu Double Q-learning-algoritmin päälle, ja sen tarkoituksena on parantaa Q-oppimisen suurinta heikkoutta eli arvojen yliarviointia. TD3 rajoittaa ääriarvoja ja tuottaa näin oppimista, joka aliarvioi arvoja. Tämä ei ole ongelma, koska toimintatapa pyrkii pois matala-arvoisista tuloksista. Tämä tekee TD3:sta yleisesti käytetyn, ja robustin algoritmin ratkaisemaan monimutkaisia ja moniulotteisia ongelmia vahvistusoppimisessa. [5]

Nämä algoritmit ovat laajasti käytettyjä, mutta paine löytää parempia algoritmeja on suuri ja uusia menetelmiä syntyy vuosittain useita. Näiden algoritmien ymmärtämisen kautta voidaan rakentaa entistä parempia vahvistusoppimisalgoritmeja, jotka voivat ratkaista entistä monikerroksisempia ja monimutkaisempia ongelmia.

2.2 Simulaatioympäristöt

Simulaatioympäristö on laaja käsite, josta tämän tutkielman kannalta tärkein osa-alue on robotiikan simulaatioympäristöt. Robotiikan simulaattori on sovellus, joka pyrkii emuloimaan oikeaa maailmaa ja luomaan simulaatioympäristön roboteille [6]. Robottien simulointi on tärkeää, koska se madaltaa niiden kehityksen kustannuksia ja mahdollistaa niiden samanaikaisen koulutuksen. [6].

Simulaattorin arkkitehtuuri vaihtelee suuresti sen tarkoituksen mukaan. Jotkut ympäristöt keskittyvät usean robotin dynamiikkaan, toiset robottien raajojen yksityiskohtaiseen liikutteluun. Simulaattori voi myös toimia erilaisilla ohjelmistoarkkitehtuureilla kuten asiakas-palvelin-arkkitehtuurilla, jossa simulaation prosessointi ja sen ohjaus tapahtuu eri laitteilla. Nämä tekijät pitää ottaa huomioon valittaessa simulaatioympäristöä robotin mallintamiseen. [6]

Robottien kouluttaminen vahvistusoppimisella simulaatioympäristössä vaatii hyvää ymmärrystä kummastakin aiheesta. Simulaation parametrien hallinnalla voidaan vaikuttaa algoritmin oppimiseen, sekä simulaatiokoulutuksen nopeuteen. Samanaikaisesti on syytä valita tilanteeseen sopiva vahvistusoppimisalgoritmi ja miettiä, mitä parametreja se tarvitsee ympäristöstä toimiakseen tehokkaasti. Simulaation ja algoritmin valintaan ei ole yhtä oikeaa vastausta, vaan se vaatii usein testaamista monella eri keinolla. Nämä tekijät ovat tärkeitä kun ratkaistaan todellisuusraon ongelmaa.

2.3 Todellisuusraon historia

Todellisuusrako (engl. sim-to-real, sim2real, sim-to-real gap) tarkoittaa todellisuuden ja simulaatioympäristön välistä eroa [7]. Se on suurin ongelma simulaatiopohjaisessa vahvistusoppimisessa. Tämä johtuu koulutuksen siirtämisestä lähdeympäris-

töstä (engl. source domain) kohdeympäristöön (engl. target domain), jota kutsutaan koulutuksen yleistämiseksi.

Todellisuusraon historian ja simulaatiopohjaisen robottien kouluttamisen historian voidaan katsoa alkaneen vuonna 1995. Silloin julkaistiin artikkeli [8] simulaatioympäristön ja simulaatiopohjaisen oppimisen käytöstä robotin kouluttamisessa. Työssä [8] käytettiin realistista robotin mallinnusta simulaatioympäristössä, jonka koulutukseen lisättiin häiriöitä parantamaan koulutuksen tulosta. Vahvistusoppimisen sijaan käytettiin geneettistä algoritmia, joka pyrkii myös kehittämään toimijan ominaisuuksia haluttuun suuntaan. Simulaatioympäristö oli yksinkertaistettu kaksiulotteinen versio robotista. [8]

Tässä ensimmäisessä tutkimuksessa käydään läpi monia todellisuusraon ongelmia kuten simulaatioympäristön rakentamista, todellisuusraon metodeja sekä oppimisalgoritmien suunnittelua. Tämän tutkimuksen vaikutukset näkyvät nykyisissä tutkimuksissa ja tämä tutkimus oivalsi jo varhain, että toimintaympäristön satunnaistaminen on tärkeä menetelmä simulaatioraon ongelman ratkaisussa.

Simulaatioympäristön suunnittelussa on tärkeää tuntea ympäristö ja sen toimija. Ennen nykyisiä menetelmiä, simulaatioiden suurin rajoitus oli tietokoneiden rajallinen laskentateho. Tämä vaati tarkempaa kartoitusta tarpeellisista ominaisuuksista, jotta simulaatioympäristön ei tarvinnut simuloida koko kolmiulotteista tilaa. [8]

Tutkimuksessa [8] käytettiin kaksipyöräistä *Khepera-robottia*, joka vältti esteitä sen ympäristössä. Simulaatioympäristö sovitettiin kaksiulotteiseksi, koska kolmannen ulottuvuuden lisääminen simulaatioon olisi ollut laskentatehollisesti kallista. Tutkijat arvioivat, että esteiden korkeus ei vaikuttanut merkittävästi robotin toimintaan. Simulaatioympäristön parametrien arvot saatiin mittaamalla fyysisen robotin arvoja. [8]

Nykyään simulaatioita pyritään suunnittelemaan mahdollisimman tarkoiksi kuvauksiksi todellisuudesta. Simulaatioympäristöjä on moni erilaisia eri käyttökohtei-

siin. Niitä on eri ohjelmointikielille ja erilaisten robottien simulointiin. Myös nykyään rakennetaan omia räätälöityjä simulaatioympäristöjä, jos paras mahdollinen koulutustulos vaatii sitä. [9] Nykyisten simulaatioympäristöjen kehityksen taustalla on tietokoneiden laskentatehon kasvu ja robottien simulaatioympäristöjen kasvanut kysyntä.

Tutkimuksessa [8] käytetään geneettistä algoritmia tuottamaan koulutusta robotille. Geneettinen algoritmi perustuu biologiseen evoluutioon, jossa populaation jäsenet lisääntyvät naapureidensa kanssa tuottaen uusia jäseniä, joilla on sekoitus ominaisuuksia sen vanhemmilta sekä tietty määrä satunnaista muutosta. [8] Geneettisen algoritmin suurin ero vahvistusoppimiseen, on sen riippuvaisuus ihmisen asettamiin ja määrittämiin arvoihin. Se ei opi itsenäisesti ympäristöstä, vaan se tarvitsee ennalta määritetyt ominaisuudet ja painot, jotta se voi oppia.

Geneettiset algoritmit eivät ole vahvistusoppimisen menetelmiä, vaan ne perustuvat evoluutioon. Vahvistusoppimisen historia perustuu kolmeen pääsuuntaukseen. Ensimmäinen on yrityksen ja erehdyksen oppimisen menetelmä (engl. trial-and-error learning), joka perustuu eläinten opetuksen psykologiaan. Toinen suuntaus on optimiohjaus (engl. optimal control), joka ratkaisee ongelmia dynaamisen ohjelmoinnin ja arvofunktioiden avulla. Kolmas suuntaus on aikaero-oppiminen (engl. temporal-difference learning), joka perustuu ajallisesti peräkkäisten todennäköisyysarvioiden eroon. aika-ero oppiminen ja optimiohjaus tuotiin yhteen Q-oppimisessa. Q-oppiminen on kasvanut nykyiseen vahvistusoppimisalgoritmien kirjoon, joka kattaa kaikki kolme historiallista pääsuuntausta. [1]

Vahvistusoppimisen teoria on huomattavasti vanhempaa kuin sen nykyiset käyttökohteet. Monet nykyään käytetyistä menetelmistä perustuvat näihin vahoihin algoritmeihin. Erityisesti tietokoneiden laskentatehon kasvu on mahdollistanut algoritmien hyödyntämisen todellisten ongelmien ratkaisussa.

Työssä [8] oivallettiin todellisuusraon keskeisiä ongelmia. Näistä tärkein oli ympäristön satunnaistaminen. Robotin koulutus täysin ideaalissa ympäristössä tuotti huonomman oppimistuloksen kuin satunnaistetussa ympäristössä koulutettu robotti. Ympäristön satunnaistaminen suoritettiin lisäämällä häiriöitä robotin sensoreiden tuloksiin, jotta jokainen koulutus oppi hieman erilaisessa ympäristössä. Työssä huomattiin myös, että häiriön suuruus ei saa olla liian suuri, koska muuten koulutus heikkenee huomattavasti. [8]

Toimintaympäristön satunnaistaminen on ensimmäinen todellisuusraon metodi, mutta sen lisäksi löytyy monia muita metodeja. Yksinkertaisin metodi on suunnitella niin tarkka simulaatio tai kouluttaa niin suurella määrällä todellista dataa, että koulutus voidaan siirtää suoraan ilman muutoksia robottiin. Tätä kutsutaan nollapisteoppimiseksi (engl. Zero-shot Transfer). Toinen yksinkertainen metodi on mallintaa todellisuus matemaattisesti mahdollisimman tarkasti. Tätä kutsutaan toimintaympäristön tunnistamiseksi (engl. System Identification). [9]

Näitä metodeja käytetään vielä ja monet monimutkaiset metodit perustuvat yksityiskohtaiseen simulaatioympäristöjen suunnitteluun. Nämä monimutkaiset metodit ovat laajasti käytettyjä ja ne ovat monipuolisempia kuin toimintaympäristön tunnistaminen ja nollapisteoppiminen.

3 Todellisuusraon metodit

Todellisuusraon ongelman ratkaisuksi on kehitetty monia metodeja, joista käsitellään kolme tutkielman kannalta keskeisintä metodia. Ne ovat toimintaympäristön satunnaistaminen (engl. domain randomization), toimintaympäristön mukauttaminen (engl. domain adaptation) ja simulaatioympäristöjen suunnittelu. Näiden metodien tarkoituksena on pienentää todellisuuden ja simulaation välistä eroa ja ne keskittyvät eri osa-alueisiin todellisuusraon ongelmassa.

3.1 Toimintaympäristön satunnaistaminen

Toimintaympäristön satunnaistaminen on metodi, joka perustuu vankkaan säätöön (engl. robust control) epävarmassa ympäristössä. Se tarkoittaa käytännössä satunnaistettuja häiriöitä simulaatioympäristön mittaustuloksissa ja dynamiikassa. Dynamiikan muutokset vaikuttavat tilan muutoksiin ja mittaustulokset vaikuttavat objektien tilojen havaintoihin.[7] Satunnaistaminen on tärkeää, jotta vavhistusoppimisalgoritmi ei opi hyväksikäyttämään simulaatioympäristön rajoituksista johtuvia virheitä. Satunnaistaminen ei kuitenkaan ole yksinkertainen metodi käyttää tehokkaasti. Se vaatii syvää ymmärrystä kohdeympäristöstä, jotta satunnaistettujen arvojen parametrit voidaan asettaa tarkasti [10].

Satunnaistamiseen on yleisesti ottaen kaksi tekniikkaa staattinen ja dynaaminen satunnaistaminen. Staattinen satunnaistaminen tarkoittaa muuttumatonta ala- ja ylärajaa arvolle, jota halutaan satunnaistaa [11]. Jos arvojen ääripäät eivät ole rea-

listisia, se heikentää oppimisen nopeutta ja sen yleistettävyyttä [10]. Staattinen satunnaistaminen on kuitenkin yleisesti käytettyä, koska se on yksinkertainen metodi toteuttaa ja sillä voi saada aikaan hyviä oppimistuloksia. Dynaaminen satunnaistaminen tarkoittaa vahvistusoppimisalgoritmin päivittävän ääriarvoja automaattisesti perustuen oppimisen tuloksiin. Tämä johtaa siihen, että satunnaistaminen ohjaa koulutusta oikeaan suuntaan. [11] Dynaaminen satunnaistaminen on tehokas metodi, mutta sen toteuttaminen käytännössä ei ole yksinkertaista ja vaatii hyvää ymmärrystä vahvistusoppimisesta sekä lähde- ja kohdeympäristöstä.

3.2 Toimintaympäristön mukauttaminen

Toimintaympäristön mukauttaminen on siirto-oppimisen metodi, jota sovelletaan vahvistusoppimisessa. Mukauttaminen pyrkii yhtenäistämään lähde- ja kohdeympäristöt, jotta kouluttamisen yleistäminen olisi tehokkaampaa. [7] Tämä metodi on paljon hankalampi toteuttaa kuin toimintaympäristön satunnaistaminen, koska esimerkiksi ympäristön ja toimijan parametrit voivat saada täysin yllättäviä arvoja perustuen koulutuksen tuloksiin. Kouluttaja ei voi tietää etukäteen minkälaisia arvoja mukauttaminen tuottaa, joka lisää kouluttajan ammattitaidon tarvetta ongelman ratkaisemiseksi.

Mukauttamisen tekniikat ovat yksilöllisempiä kuin satunnaistamisessa. Esimerkiksi mukauttamista voidaan tehdä käyttäen kohdeympäristöstä kerättyä dataa, jota käytetään lähdeympäristössä. Tämän metodin ongelmana on jakauman siirtymisen ongelma (engl. distributional shift problem). Tämä tarkoittaa todellisuuden ja simulaation välisistä eroista aiheutuvaa Q-arvojen väärin arviointia, joka johtaa huonoihin koulutustuloksiin. [7] Toinen ongelma liittyy koulutusaineiston keräämiseen kohdeympäristöstä, koska sen tulee olla mahdollisimman samankaltaista koulutusympäristön datan kanssa. Siinä tulee siis ottaa huomioon, mitä kaikkia ominai-

suuksia ja dynamiikkoja simulaatio simuloi ja mitkä niistä ovat tärkeimpiä hyvän koulutustuloksen kanssa.

Toimintaympäristön satunnaistaminen ja mukauttaminen ovat metodeja, joita voidaan käyttää yhdessä tuottamaan parempia koulutustuloksia. Esimerkiksi kohdeympäristön koulutusdataa voidaan lisätä toimijalle, joka on koulutettu alunperin satunnaistetussa simulaatioympäristössä. Tällä menetelmällä pyritään ratkaisemaan jakauman siirtymisen ongelma, koska satunnaistaminen auttaa Q-arvojen paremmassa arvioinnissa ja mukauttaminen auttaa simulaatioraon sulkemisessa. [7]

3.3 Simulaatioympäristöjen suunnittelu

Simulaatioympäristöjen suunnittelu on tärkeää jokaisen koulutuksen kannalta, koska se on yhteinen vaikuttava tekijä kaikissa ympäristöissä. Niiden suunnittelussa on yleisesti ottaen kaksi koulukuntaa räätälöity- ja yleinen simulaatioympäristö. Jotkut tehtävät ovat niin monimutkaisia, että ainoa ratkaisu niiden kouluttamiselle on simulaatioympäristön yksityiskohtainen suunnittelu [12]. Räätälöidyt ympäristöt ovat täysin tai osittain itse rakennettuja, jotta niissä voidaan ottaa huomioon niiden omat koulutustarpeet. Tämä tuottaa yleisesti parempia koulutustuloksia nopeammin, mutta koulutuksen yleistettävyys on heikkoa. Räätälöityjen ympäristöjen luonti on myös haastavaa ja monimutkaisten ongelmien ratkaisun kannalta erittäin haastavaa. Yleiset ympäristöt ovat nopeampia ottaa käyttöön koulutuksessa, mutta niiden avulla kouluttaminen on hitaampaa tietyn ongelman ratkaisun kannalta.

Simulaatioympäristöjen suunnittelu alkaa ongelman määrittämisellä. Yleensä monimutkainen ongelma pilkotaan pienempiin osiin, ja simulaatioympäristöä lähdetään rakentamaan näiden pienten ongelmien ehdoilla [12]. Yleinen ongelma on fysiikkasimulaatiot, jotka ovat tärkeitä robottien kouluttamisessa. Fysiikan tarkka mallintaminen on ongelma, joka vaatii tarkentuessaan lisää laskentatehoa. Simulaation yksityiskohtaisuus on tärkeää suunnitella oman käyttökohteen kannalta tar-

peeksi yksityiskohtaiseksi, mutta ei liian yksityiskohtaiseksi. Tämä johtuu siitä, että nopeampi simulaatio nopeuttaa myös kouluttamista ja se voi johtaa hyviin koulutustuloksiin vaikka käytettävä fysiikan simulointi ei ole kaikista tarkin [10]. Lähdeympäristössä käytetään yleensä monia erilaisia tehtäviä ja ympäristöjä, jotta koulutuksen tulos on mahdollisimman yleistettävä ja jotta algoritmi ei opi liikaa simulaattorin yksinkertaisuudesta johtuvia virheitä [7] [12].

On myös tärkeää tietää minkälaiseen todelliseen ympäristöön koulutusta ollaan siirtämässä. Tuntemalla robotin tarkat yksityiskohdat ja sen käyttämät sensorit, voidaan ne ottaa huomioon ympäristön suunnittelussa. Erilaisten sensoreiden, ohjauslaitteiden ja ympäristöjen vaikutus simulaation parametreihin on tärkeää koulutuksen yleistettävyyden kannalta [12]. Virtuaalisten sensoreiden suunnittelu on tärkeä osa koulutuksen tarkkuutta. Todellisissa sensoreissa on aina tarkkuuteen vaikuttavia virheitä [12]. Näiden virheiden simulointi voi parantaa koulutustuloksia [7]. Niitä pyritään myös ratkaisemaan uusilla tekniikoilla, jotka perustuvat virheiden korjaamiseen ohjelmoinnin tasolla [12].

Todellisuusraon vakiintuneet metodit ovat toimintaympäristön satunnaistaminen ja -mukauttaminen. Ne eivät kuitenkaan tarkoita yhtä tiettyä prosessia, vaan ne kuvaavat minkälaisilla keinoilla koulutusta pyritään parantamaan. Tämän takia ne toimivat yläkäsitteinä, mutta niiden käytännön toteutus riippuu paljon käytetystä metodista. Näiden metodien kehitys ja toimintaperiaatteet ovat tärkeitä ymmärtää, todellisuusraon ratkaisemisen kannalta.

4 Todellisuusraon metodien kehitys

Todellisuusrako on monimutkainen ongelma, jonka ratkaisuun ei ole yhtä oikeaa menetelmää. Seuraavissa luvuissa käsitellään kahta aluetta, joihin todellisuusraon ongelman voi jakaa eli vahvistusoppimisalgoritmeihin ja yleisiin metodeihin. Vahvistusoppimisalgoritmeihin keskittyvässä aliluvussa käydään läpi uusia kehityksiä vanhoihin algoritmeihin, niiden toteutuksia ja toteutuksien hyötyjä ja haittoja. Yleiset metodit tarkoittavat kaikkia ei suoraan vahvistusoppimisalgoritmeihin liittyviä metodeja. Ne pyrkivät kehittämään satunnaistamista ja mukauttamista käyttäen uusia menetelmiä kuten todellisuudesta tuotetun tiedon hyödyntämistä. Niitä käsitellään toisessa aliluvussa.

4.1 Vahvistusoppimisalgoritmien kehitys

Taulukossa 4.1 esitellään vahvistusoppimisen algoritmeja, joita käsitellään tutkielman aineistoissa. Selitteet tarkoittavat järjestyksessä vasemmalta oikealle: Mallipohjainen, Malliton, Toimintatapapohjainen, Arvopohjainen, Toimija-Kriitikko, On-toimintatapa, Off-toimintatapa. Tämä luokittelu eroaa toisessa kappaleessa esite-

Taulukko 4.1: Aineistoissa kiinnostuksen kohteena olevat vahvistusoppimistekniikat

Nimi	Algoritmi	Mallivapaa	Toimintatapa	Arvo	Tk	On-toimintatapa	Off-toimintatapa
Shakerimov [7]	DQN	X		X			X
Shakerimov [7]	DDPG	X			X		X
Shakerimov [7]	PPO	X	X			X	
Ou [10]	SAC	X					X
Zhou [12]	TD3	X			X		X
Liu [13]	PGPE	X	X			X	

tystä luokittelusta, koska tämä on yleisempi tapa järjestää vahvistusoppimisalgoritmeja. Mallipohjainen ja -vapaa tarkoittavat onko algoritmilla käytössä maailman mallinnus funktio vai ei. Tässä aineistossa käsitellään vain mallivapaita algoritmeja, koska ne soveltuvat hyvin ympäristöihin, joita ei tunneta etukäteen ja ympäristöihin, jotka ovat erittäin monimutkaisia. Toimintatapa, Arvo ja toimija-kriitikko kertovat, mitä funktiota algoritmi yrittää maksimoida. Toimija-kriitikon tapauksessa toimija yrittää maksimoida arvofunktiota ja kriitikko toimintatapafunktiota. On- ja Off-toimintatapa kertoo muuttaako algoritmi koulutuksen käytössä olevaa toimintatapaa koulutuksen aikana, vai muuttaako se ulkoista toimintatapaa.

4.1.1 Taktiili TD3

Yksi tärkeimmistä tehtävistä roboteille on esineiden käsittely. Se mahdollistaa erilaisia teollisuuden ja automaation käyttökohteita, jotka ovat tärkeitä mahdollisuuksia robotiikalle. Esineiden manipulaatio on kuitenkin ongelma, johon ei ole yksinkertaista ratkaisua. Lähes kaikki nykyiset tekniikat perustuvat konenäköön tunnistamaan esineiden oikeaa käsittelyä. Nämä tekniikat tuottavat heikkoja tuloksia, koska konenäköön vaikuttavat ympäristön valaistus ja kohteen pinnan muodot ja tekstuurit. Taktiilit sensorit voivat ratkaista tämän ongelman. [12]

Taktiilien sensorien kouluttaminen oikean maailman ympäristössä on kallista ja voi vaurioittaa robottia ja esineitä, joten sitä suoritetaan simulaatioympäristössä. TD3-algoritmia ei voida käyttää suoraan tämän ongelman ratkaisussa, koska sen käyttämä kaksoiskriitikkoverkko perustuu yksinkertaiseen konvoluutioneuroverkkoon, joka tuottaa heikkoja oppimistuloksia. Tähän on kehitetty ratkaisuna Taktiili-TD3 algoritmi, jota käytetään simulaatiossa ja todellisuudessa ja joka tuottaa parempia koulutustuloksia verrattuna TD3-algoritmiin. [12]

Ensimmäinen osa T-TD3 algoritmia on tuottaa sille sen tarvitsema kuusiulotteinen tarttumakartta (engl. tactile prior map), jonka se koostaa neljästä eri kar-

tasta [12]. Nämä neljä karttaa tuotetaan sensoripareilla, jotka on sijoitettu kappaleen molemmille puolille ja paritekniikan tarkoituksena on vähentää sensorien sisäisiä virheitä [12]. T-TD3 algoritmi käyttää MSF-Net [14] neuroverkkorakennetta sen toimija-kriitikko neuroverkkona, joka mahdollistaa robotin omien sensorien ja tarttumakartan yhdistämisen koulutusta varten [12]. Jotta kouluttaminen olisi helpompaa, tarttumistehtävä on jaettu kolmeen osaan liukkauden tunnistaminen, tarttumisen arviointi ja tarttumisen minimivoiman seuranta [12]. Jokaiseen osaan on määritetty oma palkkiofunktio ja koulutuksen tuloksena toimija hyödyntää jokaisen osan maksimointia, löytääkseen parhaan mahdollisen tarttumisalgoritmin [12].

T-TD3 algoritmi on erittäin monimutkainen toteuttaa ja vaatii vahvaa ympäristön osaamista sen toteuttajilta. Se saavuttaa myös erittäin hyviä oppimistuloksia ongelmassa, joka ei ole yksinkertainen ratkaista perinteisillä robotin koulutusmenetelmillä. Algoritmin ja koko strategian vahvuus perustuu laajaan määrään todellisuutta hahmottavaan datan keräykseen ja robusteihin neuroverkkoihin, jotka voivat oppia useasta lähteestä samanaikaisesti.

4.1.2 Hierarkkinen vahvistusoppiminen

Esineiden manipulaatio on monimutkainen prosessi, joka koostuu useasta toisiaan seuraavasta osasta. Perinteiset vahvistusoppimisalgoritmit eivät toimi tämänlaisessa ympäristössä, koska tilamuutokset ovat vaikeita kouluttaa algoritmille. [13] Tämä johtuu algoritmien pyrkimyksestä löytämään yhden oikean ratkaisun, vaikka ongelmia on useita ja ne ovat eri ympäristöissä. Yksi vastaus tähän ongelmaan on hierarkkinen vahvistusoppiminen (engl. hierarchical reinforcement learning, HRL), joka käyttää abstraktiotasoja ongelmien mallintamiseen korkeammassa tila-avaruudessa.

Hierarkkinen vahvistusoppiminen toimii käyttämällä ihmisen suunnittelemaa tai automaattisesti opittua tehtävien hierarkiaa. Tehtävien hierarkia vähentää ongel-

man monimutkaisuutta. Tähän yhdistetään oppiminen aiemmasta tiedosta, jotta saavutetaan parempia koulutustuloksia. [13]

Yksinkertaista mallia voidaan tarkastella kahdessa kerroksessa. Korkeamman kerroksen toimintatapa pyrkii valitsemaan oikean tila-avaruuden matalan tason kerrokselle. Matalan tason kerros valitsee toiminnon ja päivittää parametreja. [13] Tämä rakenne mahdollistaa usean erilaisen matalan tason kerroksen, jotka ratkaisevat omia tehtäviään. Jokaiseen ympäristöön tulee kouluttaa oma algoritminsa, joka tekee kokonaisuudesta hitaan kouluttaa, mutta se pystyy toimimaan toisistaan täysin eroavissa ympäristöissä, joka ei ole mahdollista muilla nykyisillä menetelmillä.

4.2 Yleisten metodien kehitys

Yleiset metodit tarkoittavat keinoja, joita ei voi luokitella vain vahvistusoppimisalgoritmeiksi. Nämä metodit hyödyntävät laajempaa ympäristöä eli robotin fyysisiä ominaisuuksia, metadataa ja muita keinoja parantaakseen koulutustuloksia. Taulukossa 4.2 esitellään todellisuusrakoon keskittyviä yleisiä metodeja, joita käsitellään kyseisissä aineistoissa.

Taulukko 4.2: Aineistossa käsiteltävät todellisuusraon metodit

Nimi	Metodi	Toimintaympäristön satunnaistaminen	-mukauttaminen
Muratore [11]	BO		X
Muratore [11]	BayRn	X	
Dong [15]	KFOTL		X
Zhang [16]	SysID		X
Zhang [16]	CAPTURE		X

Taulukossa esitetyt metodit ovat järjestyksessä ylhäältä alas: Bayesian Optimization, Bayesian Randomization, Kalman Filter-Based One-Shot Sim-to-Real Transfer Learning, System Identification, in-Context Adaptation Module for Sim-to-Real System Identification. Nämä metodit pyrkivät parantamaan kokonaisuutta laajemmin kuin vain vahvistusoppimisalgoritmien kautta.

4.2.1 Bayesilainen satunnaistaminen

Toimintaympäristön satunnaistaminen staattisilla muuttujilla on toimiva, mutta epätehokas tapa käyttää satunnaistamista metodina vahvistusoppimisalgoritmin toimintasuunnitelman optimoinnissa. Satunnaistamisen hyödyt ovat kuitenkin niin suuret, että metodia kehittämällä saadaan aikaan laskentatehollisesti optimoitua koulutusta. Yksi ratkaisu tähän on käyttää BayRn-metodia (engl. Bayesian Domain Randomization), joka käyttää BO-metodia (engl. Bayesian Optimization) sen perustana. [11]

Bayesialainen optimointi optimoi tuntemattoman funktion $f : X \to \mathbb{R}$ pienessä sarjassa X[10]. Se käyttää todennäköisyysmalleja, jotka ovat Gaussisia prosesseja. Gaussiset prosessit ovat jakaumia, jotka määritetään funktiolle sen historiallisen keskiarvon ja historiallisen positiivisen kovarianssifunktion avulla. Tämän todennäköisyysmallin avulla Bayesialainen optimointi optimoi miten funktion arvoja tulee parantaa. Historiallinen keskiarvo ja historiallinen positiivinen kovarianssifunktio saadaan aikaisemmista mittaustuloksista. [11]

Bayesialainen satunnaistaminen vastaa ongelmaan lähdeympäristön mukauttamisesta perustuen kohdeympäristöön. Se käyttää kahta funktiota optimoimaan kohdeympäristön maksimointia. Ensimmäinen funktio toimii löytämällä parhaan lokaalin arvon kyseisessä stokastisessa ympäristössä. Tätä toimintasuunnitelmaa arvioidaan kohdeympäristössä. Toinen funktio käyttää BO-metodia löytääkseen uuden kohdeympäristön arvojen jakauman, joita käytetään simulaation arvojen satunnaistamiseen. Samaan aikaan vanhojen arvojen ja ensimmäisen funktion tuloksen arviointiin käytetään Gaussista prosessia, joka tuottaa keskiarvon ja kovarianssin. Nämä arvot päivitetään perustuen koko historiaan eri arvoja, joita on käytetty ympäristössä. Tätä iteratiivista prosessia käydään läpi, kunnes saavutetaan ennalta määritetty hyvyyden arvo. [11]

Bayesialainen satunnaistaminen on metodi, joka hyödyntää tehokkaasti rajallista määrää todellisen maailman koulutusdataa ja se oppii keräämällä tietoa jokaisen iteraation yli. Nämä ovat tärkeitä ominaisuuksia, koska simulaation tarkkuuden kasvattaminen on usein liian vaikeaa, kallista tai epätehokasta.

4.2.2 Kalman-suotimeen perustuva siirto-oppiminen

Kohde- ja lähdeympäristön välinen ero on suurin ongelma simulaatioraossa. Yksi keino sen sulkemiseen on kuvata eroa virheenä. Sillä on keskiarvo ja kovarianssi, jotka kuvaavat havaintojen ja ympäristön välistä eroa. Tämänlaisen ongelman ratkaisuun on kehitetty suodinmenetelmiä, jotka ottavat virheen huomioon ja korjaavat sitä dynaamisesti. [15]

Simulaation ja todellisuuden välinen ero on algoritmin alussa 0. Se kasvaa jokaisella aika-askeleella, kunnes toimija ei pysty enää suorittamaan tehtäväänsä. Jos jokaisen aika-askeleen lopussa tehdään virheenkorjaus, voidaan saada toimija suorittamaan tehtävän loppuun. Tähän ongelmanratkaisuun perustuu KFOTL (engl. Kalman Filter-Based One-Shot Sim-to-Real Transfer Learning) metodi. Siinä käytetään lähdeympäristön likimääräisesti optimaalista toimintasuunnitelmaa, jota päivitetään olemaan sama kohdeympäristön toimintasuunnitelman kanssa, käyttäen Kalman-suodinta. [15]

Tämän metodin suurin ongelma on, että se ei pyri suoraan ratkaisemaan ongelmaa. Se pyrkii sopeuttamaan epätäydellisen simulaation ja todellisuuden välisiä eroja, joka on mahdollista lyhyellä aikavälillä. Kun simulaation ja todellisuuden erot kasvavat tarpeeksi suuriksi, ei suodinmenetelmät saavuta enää hyvää tulosta todellisuudessa.

4.2.3 CAPTURE

Toimintaympäristön satunnaistaminen on metodi, joka ei kuvaa suoraan kohdeympäristöä. Se saavuttaa vain keskimääräisen tilan, joka ei tarjoa ääriarvojen hienosäätöä. Ympäristön identifiointi (engl. System Identification, SysID) metodit päivittävät simulaation arvoja perustuen todellisuuteen. [16]

CAPTURE (engl. in-Context Adaptation module for sim-to-real system identification) pyrkii dynaamisesti yhtenäistämään simulaation ja todellisuuden parametrien suuntaa käyttäen historiallista vuorovaikutusdataa. Historiallinen oppiminen tarkoittaa kontekstiin perustuvaa oppimista, joka on yleinen metodi luonnollisen kielen prosessoinnissa. [16]

Käytännössä tämä prosessi toimii käyttäen simulaatioympäristön parametria ϵ kuvaamaan simulaation parametrien muutosta. Tätä optimoidaan kuten palkkiosignaalia, tuottamaan parhaan mahdollisen arvon. Tätä parametria käsitellään yhdessä tilan, toiminnon ja palkkion kanssa. [16]

Tämän metodin suurin heikkous on sen vaikeus oppia monia erilaisia tehtäviä. Se siis oppii vain yhden tehtävän kerrallaan, joten sen soveltaminen monimutkaisiin ympäristöihin on haastavaa. [16]

On selvää, että simulaatioraon ratkaisemiseen kehitetään paljon erilaisia ratkaisuja. Näiden ratkaisujen prosessointi ja konkreettisten hyötyjen löytäminen on kuitenkin vielä haastavaa. Se johtuu jokaisen uuden metodin yksityiskohtaisuudesta, jota on vaikea lähteä sovittamaan uusiin käyttökohteisiin. Näiden ja uusien metodien hyödyntäminen suuressa mittakaavassa vaatii vielä paljon tutkimusta ja vertailua.

5 Yhteenveto

Simulaatiopohjainen vahvistusoppiminen on ratkaisu robotiikan yleisempään käyttöön. Se on kustannustehokas tapa kouluttaa robotteja toimimaan monimutkaisissa ympäristöissä, joita ei voida ohjelmoida etukäteen, kuten perinteiset robotit ohjelmoidaan. Simulaatiorako on kuitenkin suuri ongelma, jota ei ole vielä pystytty ratkaisemaan riittävän hyvin. Siihen kehitetään ja on kehitetty ratkaisuja, metodeja ja algoritmeja. Nämä eivät vielä ole sillä tasolla, että robotit pystyisivät esimerkiksi autonomiseen lentämiseen metsässä tai erilaisten esineiden käsittelyyn automaattisesti. Tämä ala kuitenkin kehittyy nopeasti ja on mielenkiintoista nähdä mihin tämä kehitys lopulta johtaa.

Tutkimuksen ensimmäinen tutkimuskysymys on mitä ovat todellisuusraon metodit? Todellisuusrakoon on kehitetty vakiintuneita metodeja kuten toimintaympäristön satunnaistaminen, -mukauttaminen ja simulaation suunnittelu. Satunnaistaminen auttaa algoritmia ymmärtämään erilaisia tilanteita, joten sen oppimisen hyödyllisyys kasvaa. Tämän ongelmana on satunnaistamisesta johtuva koulutuksen määrän kasvu. Mukauttaminen pyrkii käyttämään kohdeympäristöä hyödyksi lähdeympäristön toimintasuunnitelman parantamisessa. Tämä johtaa koulutuksen tehokkuuden kasvuun ja myös parempiin oppimistuloksiin. Tällä metodilla ei ole yhtä paljon heikkouksia kuin satunnaistamisella, mutta sitä käytetään yleensä yhdessä satunnaistamisen kanssa. Simulaation suunnittelussa tasapainotellaan simulaation tarkkuutta ja sen laskentatehokkuutta. Korkeampi tarkkuus parantaa oppimistuloksia, mutta

johtaa laskennan määrän kasvuun. Matalampi tarkkuus voi myös johtaa hyviin oppimistuloksiin, koska koulutusta voidaan tehdä enemmän samassa ajassa. Simulaation suunnittelu on yksi tärkeimmistä, mutta vähiten tutkituista kokonaisuuksista simulaatioraon näkökulmasta.

Tutkimuksen toinen tutkimuskysymys on miten todellisuuraon metodeja on kehitetty Todellisuusraon metodeja kehitetään niin vahvistusoppimisen algoritmeihin kuin tarkempiin metodeihin. Algoritmien kehitys pyrkii yleisesti ottaen tehostamaan niiden toimintaa korkeaulotteisissa toimintaympäristöissä ja tila-avaruuksissa. Monet nykyiset algoritmit eivät sovellu esimerkiksi robotiikan ongelmien ratkaisemiseen suoraan, vaan algoritmeihin pitää lisätä robusteja ominaisuuksia, jotta ne voivat oppia monimutkaisia ongelmia. Tarkemmat metodit hyödyntävät satunnaistamista ja mukauttamista, mutta tekevät niin omilla tekniikoillaan. Yleisin käytetty menetelmä on satunnaistaa eri arvoja ja tehdä niiden pohjalta tilastollisesti parempia ratkaisuja. Toinen tärkeä menetelmä on hyödyntää koko oppimisen historiaa koulutuksen aikana, jotta koulutuksen tulos olisi vakaampi. Nämä metodit eivät ole sidottuja mihinkään yksittäiseen vahvistusoppimisalgoritmiin. Tämä helpottaa niiden käyttöön ottoa uusissa kohteissa, verrattuna monimutkaisiin algoritmeihin.

Todellisuusrako on aiheena uusi ja siihen liittyvää tutkimusta tuotetaan kiihtyvässä määrin lisää. Tämä johtaa tiedon vakauden heikkenemiseen ja uudelta tuntuvien lähteiden nopeaan vanhentumiseen. Esimerkiksi lähteessä [11] sanotaan simulaation ja todellisuuden välisen raon ratkaisuun olevan nykyisillä metodeilla kaksi yleistä vastausta. Ensimmäinen on simulaation tarkkuuden parantaminen tarkemmin valituilla muuttujilla ja paremmalla ympäristön simuloinnilla. Toinen on simulaatioympäristön muuttujien satunnaistaminen, jotta algoritmi ei ylisovita oppimista simulaation heikkouksiin [11]. Nykyisen aineiston valossa ehdotan kolmatta yleistä vastausta eli simulaatioympäristön simulaation nopeuttaminen, tarkkuuden kustannuksella. Tämä voi johtaa parempiin koulutustuloksiin, mutta sen suurin hyöty

on koulutuksen todellisen ajan määrän pienentyminen. Tämä perustuu lähteen [10] tuloksiin tehokkaan, mutta vähemmän tarkan nesteen simuloinnin käyttöön ja sen tuottamiin hyviin koulutustuloksiin.

Tämän tutkielman tekeminen on tuonut esiin monia asioita, joita olisi voinut tehdä toisin tai paremmin. Tutkielman aihealueeseen tutustuminen paremmin, ennen tutkielman tekoa, olisi vastannut moneen kysymykseen ja auttanut rakenteen suunnittelussa. Tässä ehkä tärkeimpänä se, miten tarkasti mistäkin aiheesta voi tai kannattaa kertoa. Aineistossa on hyvin yksityiskohtaisesti selitetty, miten niiden käyttämät ratkaisut toimii. Tätä ei tietenkään voi suoraan esittää tutkimuksessa, tai sen pituus kasvaa liian pitkäksi ja tiedon tiivistäminen vaikeutuu. Tämä tiivistäminen oli kuitenkin todella hankalaa ja osittain tulokset, joihin tässä tutkimuksessa päädyttiin, eivät varmasti kata jokaisen ratkaisun täyttä syvyyttä. Tutkielma on kuitenkin onnistunut tiivistämään pääpiirteet aineistosta, joka riittää tämän tutkielman laajuuden kannalta.

Tämän tutkielman perusteella voidaan ehdottaa jatkotutkimuskohteita. Niitä ovat simulaatioympäristöjen suunnittelu eli simulaation parametrien, algoritmien ja muiden ominaisuuksien suunnittelun periaatteet. Toinen alue on todellisuusraon metodien luokittelu, koska nykyiset metodit ja algoritmit ovat liian monimutkaisia, jotta niitä voitaisiin soveltaa suoraan monissa eri käyttökohteissa. Lisäksi tarvitaan näiden metodien välisiä tutkimuksia, joissa todetaan näiden metodien hyödyt ja haitat suhteessa toisiinsa. Tutkimalla tätä aihetta, saadaan kehitettyä parempia robotteja ja autonomisia järjestelmiä tulevaisuudessa.

Lähdeluettelo

[1] R. S. Sutton ja A. G. Barto, *Reinforcement Learning* (Adaptive Computation and Machine Learning series), 2. painos. Cambridge, MA: Bradford Books, marraskuu 2018.

[2] F. AlMahamid ja K. Grolinger, ”Reinforcement Learning Algorithms: An Overview and Classification”, teoksessa *2021 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2021, s. 1–7. DOI: 10.1109/CCECE53047.2021.9569056.

[3] L. Jia, B. Su, D. Xu ja Y. Wang, ”Proximal Policy Optimization with an Activated Policy Clipping”, teoksessa *2024 International Conference on Energy and Electrical Engineering (EEE)*, 2024, s. 1–5. DOI: 10.1109/EEE59956.2024.10709748.

[4] M. Hlavatý ja A. Kozáková, ”Development of Advanced Control Strategy Based on Soft Actor-Critic Algorithm”, teoksessa *2023 International Scientific Conference on Computer Science (COMSCI)*, 2023, s. 1–5. DOI: 10.1109/COMSCI59259.2023.10315824.

[5] S. Fujimoto, H. van Hoof ja D. Meger, ”Addressing Function Approximation Error in Actor-Critic Methods”, teoksessa *Proceedings of the 35th International Conference on Machine Learning*, J. Dy ja A. Krause, toim., sarja Proceedings of Machine Learning Research, vol. 80, PMLR, 2018, s. 1587–1596. url: https://proceedings.mlr.press/v80/fujimoto18a.html.

[6] K. Kumar ja P. S. Reel, ”Analysis of contemporary robotics simulators”, teoksessa *2011 International Conference on Emerging Trends in Electrical and Computer Technology*, 2011, s. 661–665. DOI: 10.1109/ICETECT.2011.5760200.

[7] A. Shakerimov, T. Alizadeh ja H. A. Varol, ”Efficient Sim-to-Real Transfer in Reinforcement Learning Through Domain Randomization and Domain Adaptation”, *IEEE Access*, vol. 11, s. 136 809–136 824, 2023. DOI: 10.1109/ACCESS.2023.3339568.

[8] N. Jakobi, P. Husbands ja I. Harvey, ”Noise and the reality gap: The use of simulation in evolutionary robotics”, teoksessa *Advances in Artificial Life*, F. Morán, A. Moreno, J. J. Merelo ja P. Chacón, toim., Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, s. 704–720, ISBN: 978-3-540-49286-3.

[9] W. Zhao, J. P. Queralta ja T. Westerlund, ”Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey”, teoksessa *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2020, s. 737–744. DOI: 10.1109/SSCI47803.2020.9308468.

[10] Y. Ou, A. Soleymani, X. Li ja M. Tavakoli, ”Autonomous Blood Suction for Robot-Assisted Surgery: A Sim-to-Real Reinforcement Learning Approach”, *IEEE Robotics and Automation Letters*, vol. 9, nro 8, s. 7246–7253, 2024. DOI: 10.1109/LRA.2024.3421191.

[11] F. Muratore, C. Eilers, M. Gienger ja J. Peters, ”Data-Efficient Domain Randomization With Bayesian Optimization”, *IEEE Robotics and Automation Letters*, vol. 6, nro 2, s. 911–918, 2021. DOI: 10.1109/LRA.2021.3052391.

[12] Y. Zhou, Y. Jin, P. Lu, S. Jiang, Z. Wang ja B. He, ”T-TD3: A Reinforcement Learning Framework for Stable Grasping of Deformable Objects Using Tactile

Prior", *IEEE Transactions on Automation Science and Engineering*, vol. 22, s. 6208–6222, 2025. DOI: 10.1109/TASE.2024.3440047.

[13] X. Liu, G. Wang, Z. Liu, Y. Liu, Z. Liu ja P. Huang, "Hierarchical Reinforcement Learning Integrating With Human Knowledge for Practical Robot Skill Learning in Complex Multi-Stage Manipulation", *IEEE Transactions on Automation Science and Engineering*, vol. 21, nro 3, s. 3852–3862, 2024. DOI: 10.1109/TASE.2023.3288037.

[14] J. Chen, J. Fan, M. Zhang, Y. Zhou ja C. Shen, "MSF-Net: A Multiscale Supervised Fusion Network for Building Change Detection in High-Resolution Remote Sensing Images", *IEEE Access*, vol. 10, s. 30 925–30 938, 2022. DOI: 10.1109/ACCESS.2022.3160163.

[15] Q. Dong, P. Zeng, G. Wan, Y. He ja X. Dong, "Kalman Filter-Based One-Shot Sim-to-Real Transfer Learning", *IEEE Robotics and Automation Letters*, vol. 9, nro 1, s. 311–318, 2024. DOI: 10.1109/LRA.2023.3333661.

[16] X. Zhang, S. Liu, P. Huang, W. J. Han, Y. Lyu, M. Xu ja D. Zhao, "Dynamics as Prompts: In-Context Learning for Sim-to-Real System Identifications", *IEEE Robotics and Automation Letters*, vol. 10, nro 4, s. 3190–3197, 2025. DOI: 10.1109/LRA.2025.3540391.