Masteroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for elektroniske systemer

Harald Lønsethagen

Bruk av nevrale nettverk med minne for forbedring av sanntids lokasjonssystemer

Masteroppgave i elektronisk systemdesign og innovasjon
Veileder: Bjørn B. Larsen
Juli 2019

NTNU
Kunnskap for en bedre verden

Harald Lønsethagen

Bruk av nevrale nettverk med minne for forbedring av sanntids lokasjonssystemer

Masteroppgave i elektronisk systemdesign og innovasjon
Veileder: Bjørn B. Larsen
Juli 2019

Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for elektroniske systemer

NTNU
Kunnskap for en bedre verden

Sammendrag

Sanntids lokaliseringssystemer (SLS) har hatt en signifikant økning i anvendelser i det vi beveger oss inn i æraen av tingenes internett. Tradisjonelle lokaliseringsløsninger sliter med å møte de stadig økende nøyaktighetskrav fra moderne produkter og tjenester. Kunstig intelligens (KI) har de siste årene vist seg å være et særdeles nyttig verktøysett for å løse problemer av svært ulik art. Stadig flere KI-teknikker blir også anvendt innenfor lokaliseringssystemer.

Bransjen for delvis- eller helt selvkjørende biler har vist spesiell stor interesse for lokalisering da det er helt essensielt for selvkjørende biler å kjenne til sin posisjon i omgivelsene.

Formålet med denne oppgaven er å undersøke om bruk av et nevralt nettverk med minne, som tar inn sanntidsdata fra et SLS, kan lage et posisjonsestimat som har høyere nøyaktighet enn det originale posisjonsestimatet fra samme SLS. Videre ble det valgt å benytte blåtann fordi dette er en radioteknologi som lett lar seg integrere og fordi teknologien finnes i de fleste kjøretøy.

Quuppa Intelligent Locating System (QILS) er markedets ledende leverandør av lokaliseringssystem basert på blåtann. Med sine nyvinnende mottakere, hver besittende av et komplisert antenne-array, er det mulig å beregne estimat for ankomstvinkel og derigjennom posisjon. QILS tilbyr et API som tilgjengeliggjør sanntidsdata, som posisjonsestimat, filtrert posisjonsestimat, usikkerhets-kovarians matrise samt andre data. Med dette som bakgrunn ble QILS valgt som sanntids-lokaliseringssystemet for denne oppgaven.

Det er i denne anledning anskaffet og anvendt et eksemplar av Quuppa Demo Kit (QDK), som inkluderer sendere, mottakere og en prosesseringsenhet. Utstyret er satt opp på et lager med mange hindringer, med hensikt å etablere et testoppsett som kan representere et kaotisk veikryss for å gjøre undersøkelsene og eksperimentering med nevrale nett i omgivelser som er realistiske.

For å gjennomføre målinger og innhente data ble en mobil plattform besittende av flere blåtann-sendere, en laser og laptop realisert. Videre ble en metodikk utformet for å konstruere gode og store mengder treningdata.

Resultatene viser at det nevrale nettverket lærer seg å predikere posisjoner, men nøyaktig-

heten overstiger ikke estimatene fra QILS. Validering av nettverket over det fullstendige datasettet gir en score på 0.000508 til nettverket, sammenlignet med 0.000501 til posisjonsestimatene til Quuppa. Anbefalt videre arbeid er å gjøre ytterligere undersøkelser av ulike parametre til nettverket, og bruken av andre nettverksarkitekturer.

Abstract

Real-Time Locating Systems (RTLS) has had a significant increase in applications as we move into the era of the Internet of Things. Traditional localization solutions are struggling to meet the ever-increasing accuracy requirements of modern products and services. Artificial Intelligence (AI) has in recent years proved to be a particularly useful toolkit to solve very different problems. More and more AI-techniques are also being used in localization systems.

The industry for partly or completely driverless cars has shown particular interest in localization as it is absolutely essential for driverless cars to know their position in the surroundings.

The purpose of this task is to investigate whether using a neural network with memory, which captures real-time data from a RTLS, can create a position estimate that has higher accuracy than the original position estimate from the same RTLS. Furthermore, it was chosen to use bluetooth because this is a radio technology that can easily be integrated and because the technology is found in most vehicles.

QILS is the market leader in the location system based on on bluetooth. With its innovative receivers, each possessing a complex antenna array, it is possible to calculate the Angle of Arrival estimate and thereby the position. QILS provides an API that exposes real-time data, such as position estimate, filtered position estimate, uncertainty covariance matrix, and other data. With this as background, QILS was chosen as the RTLS for this task.

For this reason, a copy of QDK, which includes transmitters, receivers and a processing unit, has been acquired and used. The equipment is set up on a warehouse with various obstacles, intended to establish a test set-up that can represent a chaotic junction to make the investigations and experimentation with neural networks in realistic environments.

In order to carry out measurements and obtain data, a mobile platform possessing several bluetooth transmitters, a laser and a laptop was realized. Furthermore, a methodology was designed to construct high quality and large amounts of training data.

The results show that the neural network learns to predict positions, but the accuracy does not exceed the estimates of QILS. Validation of the network over the complete dataset

gives a score of 0.000508 to the network, compared to 0.000501 to the position estimates of Quuppa. Recommended further work is to further investigate various parameters of the network and the use of other network architectures.

Forord

Denne masteroppgaven avsluttes min mastergrad ved Norges teknisk-naturvitenskapelige universitet (NTNU) i løpet av våren 2019. Mitt fagområde er Embedded Systems ved Institutt for elektroniske systemer. Problemet som studeres i denne oppgaven, er satt sammen i samarbeid med Aventi Intelligent Communications (ACI), og undersøker hvordan man skal utforme et system som kan lokalisere biler i veikryss. Denne oppgaven er en videreføring av prosjektrapporten min skrevet høsten 2018.

Jeg vil gjerne gi en spesiell takk til min veileder, professor Bjørn B. Larsen ved Institutt for elektroniske systemer for gode råd, og med-veileder fra Aventi, Bjørn Magne Elnes, for hjelp med anskaffelse av utstyr og lokale.

Harald Lønsethagen

Trondheim, Juli 2019

Innhold

Forkortelser

AoA Angle of Attack. 5, 10

C-ITS Cooperative Intelligent Transport Systems. 3

GPS Global Positioning System. 3

KI kunstig intelligens. i, 4, 5

LSTM Long Short-Term Memory. 17

MSSI mottatt signalstyrke indikator. 8

QDK Quuppa Demo Kit. i, iii, 4, 10, 19

QILS Quuppa Intelligent Locating System. i–iii, 4, 10, 20, 27, 28, 32, 37–41, 43

RNN Recurrent Neural Network. 14

RSSI Received Signal Strength Indicator. 5, 8, 16

SLS sanntids lokaliseringssystemer. i, 4–7, 10

UWB Ultra Wide Band. 16

V2I Vehicle-to-infrastructure. 11

V2V Vehicle-to-vehicle. 11

Forkortelser

AoA Angle of Attack. 5, 10

C-ITS Cooperative Intelligent Transport Systems. 3

GPS Global Positioning System. 3

KI kunstig intelligens. i, 4, 5

LSTM Long Short-Term Memory. 17

MSSI mottatt signalstyrke indikator. 8

QDK Quuppa Demo Kit. i, iii, 4, 10, 19

QILS Quuppa Intelligent Locating System. i–iii, 4, 10, 20, 27, 28, 32, 37–41, 43

RNN Recurrent Neural Network. 14

RSSI Received Signal Strength Indicator. 5, 8, 16

SLS sanntids lokaliseringssystemer. i, 4–7, 10

UWB Ultra Wide Band. 16

V2I Vehicle-to-infrastructure. 11

V2V Vehicle-to-vehicle. 11

Kapittel 1

Introduksjon

Nåværende samfunn blir stadig mer digitalisert og automatisert. Mange nye tekniske utfordringer kommer som en følge. Selvkjørende biler er et område hvor automatisering ser ut til å ha gunstige fordeler i industriell og privat bruk. Selvkjørende biler vil kunne frakte både personer og gods, til en lavere kostnad enn nåværende transportløsninger. På grunn av de økonomiske insentivene for å tilrettelegge for selvkjørende kjøretøy, er dette et felt som er i rask utvikling. For både delvis- eller fullstendig autonome biler, så er det svært viktig å kunne bestemme sin posisjon, med høy presisjon og med lav tidsforsinkelse. Den tekniske utfordringen med å bestemme posisjon, også kalt lokalisering, har vist seg å være av høy vanskelighetsgrad.

Et sentralt tema er lokalisering av kjøretøy i situasjoner som krever mange interaksjoner mellom kjøretøy. Denne masteroppgaven tar for seg en slik situasjon, lokalisering av kjøretøy i et veikryss.

Det finnes allerede et vidt utbredt posisjonseringssystem i de fleste moderne biler, kjent som Global Positioning System (GPS). Teknologien er svært populær, da den har evnen til å navigere sjåfører til ønsket lokasjon, samt kunne oppgi posisjon med ca. 10 meters presisjon [10]. GPS mangler dog nøyaktigheten som kreves for moderne Cooperative Intelligent Transport Systems (C-ITS). GPS er basert på kommunikasjon med satelitter, som typisk befinner seg i en avstand på x kilometer fra ?. Dette fører til begrensninger av nåværende versjon av GPS til å kunne tilfredsstille de høye krav til dagens C-ITS systemer. Blant annet vil GPS-systemet bli skadet, eller fullstendig slutte å fungere i bysentrum med høye hus, eller tuneller. I områder hvor det er vanskelig å få en klar synslinje til en satelitt, vil et GPS-basert lokaliseringsystem generelt har store vanskeligheter. Det er derfor behov for å utvikle et lokaliseringssystem som både er raskt, presist, men hvor det økomomiske aspektet er også viktig. Derfor er kriterier som skalerbarhet, modenhet og pris svært viktig for et system som skal fungere som posisjonsestimat-systemet for moderne kjøretøy.

Denne masteroppgaven vil ta for seg lokalisering av biler i veikryss, ved bruk av Bluetooth. Bluetooth er en svært utbredt teknologi som finnes i mange elektroniske apparat og har mange applikasjoner. Viktigst av alt så er alle moderne kjøretøy utstyrt med en Bluetooth-sender, noe som vil si at utspredt adapsjon av det presenterte systemet vil kun kreve infrastrukturelle installasjoner langs veibanen, og ikke annet enn en software-oppdatering på kjøretøyene. Dette poenget er sentralt i denne masteroppgaven, da det legges vekt på at presentert system skal potensielt være en del av et kommersielt produkt.

QILS er et kommersielt lokaliseringssystem som baserer seg på Bluetooth-sendere, kalt tags, og strategisk plasserte noder, kalt locators, på høytliggende steder med godt synsfelt over ønsket lokasjonsområde. Quuppasystemer har blitt benyttet i mange bransjer, blant annet: helsetjenester, industrielt internett, lagerlogistikk, sikkerhet og sport [6]. I denne oppgaven er et eksemplar av 'QDK' brukt. Under lokalisering tilgjengeligjør QILS en mengde sanntidsdata gjennom et API; som posisjonsestimat, filtrert posisjonsestimat, usikkerhets-kovarians matrise og mye mer. Dette API'et brukes aktivt i oppgaven.

Målet med denne oppgaven er å undersøke om ett kunstig nevralt nettverk som tar inn data fra QILS sitt API, kan gi et mer nøyaktig posisjonsestimat enn det originale estimatet fra QILS.

1.1 Struktur i oppgaven

Stukturen av denne oppgaven er ment å bygge opp nødvendig kunnskap om SLS og relatert teknologi, samt vise til noen lignende prosjekter hvor nevrale nettverk har blitt brukt for lokalisering, slik at leseren er bedre i stand til å forstå metode og implementasjonskapittelet.

Kapittel 2 innfører en rekke begreper og setter leseren inn i sentral teori. SLS er presentert, og deres evne til posisjonsering av biler i veikryss kommentert. Sentrale begreper innenfor KI er presentert, samt vanlige nevrale nettverksarkitekturer. Kapittel 3, Metode, beskriver avgrensningene og antagelsene gjort i oppgaven, for så å presentere overordnet testoppsett og planlagt nevralt nettverksarkitektur. Det som følger er en beskrivelse av hvordan det eksperimentelle oppsettet av utstyr på et lager har blitt gjennomført, hvor løsninger på praktiske problemer er inkludert. Underkapittel 3.4 tar leseren gjennom datainnsamlingemetodikken, for så å beskrive de steg som er gjort for å klargjøre eller preprossere dataen til det nevrale nettverket. En kort beskrivelse av implementasjonen av det foreslåtte nevrale nettverket vil bli beskrevet. Kapittel 5 presenterer først målt nøyaktighet til QILS for så å sammenligne dette med resultatene fra det utviklede nettverk. Kapittel 6 vil være en diskusjon av både det eksperimentelle oppsettet, datainnsamlingemetodikken og det endelige resultatet fra nettverket. Forslag til videre arbeid følger. En konklusjon av oppgavens arbeid og funn avslutter oppgaven.

Kapittel 2

Bakgrunn og teori

I dette kapitellet innføres konsepter, teknikker og definisjoner av teknologier som er relevant for sanntids lokaliseringssystemer (SLS) (eng: real-time locating systems) ettersom systemet utviklet i denne oppgaven befinner seg i denne kategorien. Fordi systemet utviklet i denne oppgaven bruker maskinlæring og nevrale nett vil en introduksjon av sentrale konsepter innenfor KI presenteres. Til slutt presenteres relatert arbeid om hvordan KI kan brukes for å lage gode SLS.

2.1 Intelligente Transportsystemer og radioteknologier

I denne seksjonen er først SLS (eng: real-time locating systems) med vanlige anvendelser og utfordringer introdusert. Det som følger er en utdyping av utbredte lokaliseringsteknikker som Received Signal Strength Indicator (RSSI) og Angle of Attack (AoA), før en kort introduksjon av intelligente transportsystemer. Et begrep som brukes flittig utover i oppgaven er ordet *tag*. Med *tag* så menes en sporbar enhet, typisk ikke større enn en fyrstikkeske, som sender ut radiosignaler. Enheten er ikke utstyrt med teknologi eller logikk til å finne sin egen posisjon, men noder i nærheten fanger opp signaler sendt av en tag og kan ved ulike teknikker estimere posisjonen. *Beacon* og tag brukes om hverandre.

2.1.1 Sanntids lokaliseringssystemer

Sanntids lokaliseringssystemer (SLS) er et samlebegrep av systemer som kontinuerlig identifiserer og bestemmer lokasjonen til et objekt eller menneske, vanligvis innenfor et avgrenset område. Normalt er en eller flere trådløse tags festet til et objekt eller menneske, og flere referansepunkter med kjent lokasjon mottar radiosignaler for å bestemme deres

lokasjon. Applikasjonene til SLS varierer stort og etter den økende populariteten til IoT-enheter, oppdages nye brukstilfeller. Følgende er et sammendrag av noen av de vanligste bruksområdene.

Anvendelser av sanntids lokaliseringssystemer

Selv om systemet presentert i denne oppgaven er designet for lokalisering av kjøretøy i veikryss, deler det mange likheter med SLS for andre domener. Metoder og teknikker utviklet i denne oppgaven kan også anvendes i de domenene, og det presenteres derfor ulike anvendelser for SLS.

Lokasjonsbeviste tjenester: Muséer inneholder vanligvis tusenvis av utstillinger, og spesielle utstillinger vil med all sannsynlighet være mer interessante for noen besøkende enn andre. Ved hjelp av lokasjonsbeviste tjenester kan muséet lede besøkende til bestemte utstillinger, og ta hensyn til personlige preferanser. Også informasjon om en utstilling kan gis til besøkende som nærmer seg en utstilling.

Navigasjonssytemer: Om en passasjer ønsker å finne veien til sin gaten på et flyplass, kan et navigasjonssystem, i likhet med GPS, navigere han eller henne på veien [2]. På samme måte kan Besøkende til et bilbiotek få hjelp med å finne en bok.

Personlig reklame: Det er vanlig praksis på nett å bruke personlig reklame, men det er vanskeligere å anvende i den fysiske verden. Med introduksjon av lokasjonsbeviste tjenester er det mulig å designe lignende funksjone, for eksempel reklame i et kjøpesenter basert på personlig preferanser.

Helsetjenester: Sykehus er utsatt for stress og tidskritiske situasjoner. Sporing av medisinsk personell vil hjelpe i situasjoner hvor pasienter trenger akutt medisinsk behandling, og det må informeres om hvilke personell som er i nærheten. Dette senker behovet for å sende ut melding til en større gruppe personell for å sikre å nå de nærmest.

Utstyrsforsvaltning: Mange bedrifter har et stort antall eiendeler, men har ofte utfordringer med å holde oversikt over plasseringen av disse. Nye teknologier har allerede dramatisk forbedret kostnadene knyttet til merking av eiendeler, hvor det er spesielt populært å benytte slike systemer i lager.

Sikkerhet: Å spore bevegelsene og interaksjonene av kunder kan hjelpe med å oppdage trusler som tyveri. Militæret kan også ha stor interesse i å spore lokasjonen av sine tropper og eiendeler, noe som kan gi verdifull informasjon til strategiske beslutninger.

Livskvalitet: Lokasjonsbeviste tjenester som samarbeider med IoT-enheter har stort potensial til å forbedre livskvalitet generelt. Noen eksempler er dører som åpnes automatisk når en person nærmer seg, et lys som slår på når en person går inn i rommet, datamaskiner som slås av når de forlates, og kaffemaskiner som automatisk lager kaffe basert på ankomst.

Utfordringer med SLS

SLS kan bygge på ulike teknologier, men svært mange benytter radiosignaler, med noen få unntak som f.eks. ultralyd. Dette medfører at de fleste SLS har sammenfallende utfordringer.

Flerveis propagasjon: Ettersom de fleste SLS bruker radiosignaler, arver de noen av deres grunnleggende problemer, der flerveis effekter kan være de mest alvorlige. Signaler som reflekteres fra omgivelsene, som vegger, skrivebord og elektronisk utstyr, gjør at mottakeren vanligvis mottar en rekke faseskiftede signaler. Det er vanligvis ekstremt vanskelig å skaffe seg klarlinje-signalet som avhenger av komplekse algoritmer. Store og komplekse algoritmer krever som regel betydelige prosesseringsressurser, noe som ofte er upraktisk [16].

Radiomiljø: Ytelsen av de fleste SLS avhenger sterkt av miljøet det brukes i. Hva veggene, bordene, stolene, etc. er laget av, er en viktig faktor da materialer reflekterer radiosignaler i ulike grad. Et stadig endrende miljø, som gående folk eller kjørende biler, vil også være vanskelig. Andre elektroniske enheter som sender radiosignaler kan være ødeleggende for SLS, da mottakeren kan ha problemer med å skille hvilke signaler som kommer fra hvilke sendere. En klarsiktlinje i et innemiljø er ofte umulig, noe som skaper et behov for å modellere miljøet. Moderne SLS leveres derfor ofte med programvare for å modellere miljø.

Kostnad: For mange SLS systemer er det nødvendig å installere noder i tak, eller i høytliggende posisjoner, og denne opprettelsen av infrastruktur kan være svært kostbar. SLS krever også utvikling av programvare og bruk av servere. Bruk av eksisterende infrastruktur kan derfor være en stor kostnadsbesparelse, og bør i alle tilfeller undersøkes om er mulig å ta i bruk.

Energieffektivitet: Det stilles høye krav til batterilevetiden til tags, som ofte er enhetene som skal lokaliseres i SLS. I flere SLS er senderen ofte innebygd i IoT-enheter, slik at de deler samme energikilde. Det er derfor svært viktig å designe et system slik at de ikke ødelegger kjernefunksjonaliteten til enheten de sitter på [11].

Personvern: De fleste brukere av SLS er ikke komfortable med å dele sin posisjon med ukjente tredjeparter. En innsats i å lage forståelige og kundevennlige kontrakter vil være viktig for stor adopsjon av SLS.

Standardisering: Per dags dato finnes det ingen standarder for design av lokaliseringsteknologier for å gjøre interkompabilitet mellom ulike enheter og merker mulig. Som mange næringer har opplevd, er slike standardiseringer viktig for stor skala adopsjon.

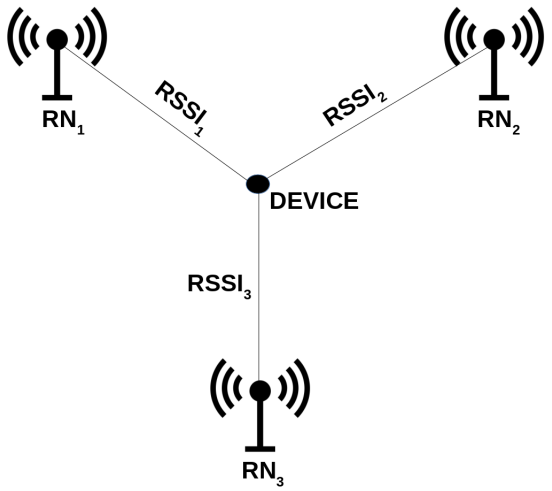
2.1.2 Mottatt signalstyrke indikator

Mottatt signalstyrke indikator (MSSI), kjent som Received Signal Strength Indicator (RSSI), er den vanligste og mest utbredte lokaliseringsteknikken. RSS, som ikke bør forveksles med RSSI, er mottatt signalstyrke målt ved mottakeren, vanligvis i decibel-milliwatt (dBm) eller milliWatt (mW). Gitt at effekten ved senderen (Tx) er kjent, kan RSS brukes til å estimere avstanden mellom senderen og mottakeren (Rx). Høyere RSS betyr mindre avstand mellom Tx og Rx, mens lavere RSS betyr større avstand. RSSI er RSS *indicator*, en relativ og abstrakt enhet. Tolkningen av RSSI-verdien er leverandørspesifikk, med forskjellige leverandører som bruker ulike typer RSSI-verdier. For eksempel bruker Atheros WiFi RSSI-verdier mellom 0 og 60, mens Cisco bruker mellom 0 og 100 [14].

Ved å bruke en enkel forplantningsmodell og RSSI-verdien, kan avstanden d mellom Tx og Rx beregnes fra (2.1) som

$$\mathrm{RSSI} = -10 * n \log_{10}(d) + A, \tag{2.1}$$

hvor n er propagasjonstapkoeffisient, mens A er RSSI verdien på en referansepunkt fra mottakeren. Det er dermed mulig å bestemme posisjonen til en sender, om man har minst tre mottakere, ved hjelp av trilaterasjon og grunnleggende geommetri/trigonometri som illustrert i figur 2.1.



Figur 2.1: RSSI-basert lokalisering

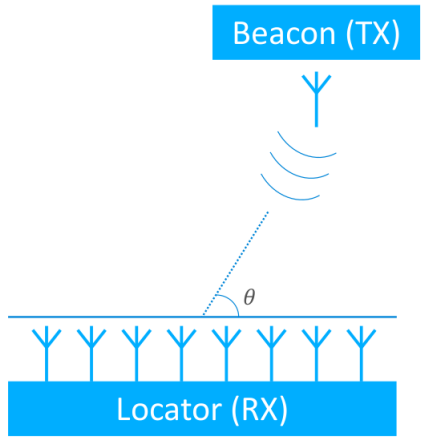
Pga. sin simplisitet er RSSI-metoden velkjent og svært utbredt, men kommer med noen betydelige ulemper.

I situasjoner hvor en direktelinje mellom sender og mottaker er umulig, vil dette føre til store fall i nøyaktigheten. Flere reflekterte signaler vil være eneste mulighet for lokasjon og på grunn av signaldemping gjennom vegger eller andre hindringer, vil gjøre det vanskelig å bestemme avstand. RSSI er også spesielt følsom overfor andre signalkilder, da det kan være vanskelig å skille mellom ønsket signal og bakgrunnsstøy.

2.1.3 Angle of Arrival

Hovedprinsippet bak Angle of Arrival er at den sporede enhet sender ut et signal som kun krever én antenne, mens mottakeren har en spesiell antenne-array, hvor hver antenne har muligheten til å finne nøyaktig ankomsttidspunkt til et signal.

Hver antenne kan sample både amplituden og fasen til et signal, og siden det er forskjellig vinkel og avstand mellom senderen på enheten, og hver spesifikke antenne på mottakeren, så er små forskjeller i fase og amplitude mulig å detektere. Ved å bruke denne amplitude- og faseinformasjonen, samt den fysiske posisjonen til hver antenne, er det mulig å beregne et estimat for ankomstvinkelen. Figur 2.2 illustrerer dette.



Figur 2.2: Angle of Arrival

I praksis har denne teknikken lignende utfordringer som andre teknikker, da flerveis-propagasjon og antenne-konfigurasjonen påvirker nøyaktigheten. Den grunnleggende fordelen med AoA er at det kun kreves én antenne på den sporede enhet. Hardware og software på tag kan være enkel, mens kompleksiteten er plassert på mottakeren, som typisk er koblet til strøm og holder betydelige prosesseringsressurser. I teorien er det

kun nødvendig med ett antenne-array for å beregne posisjon, men dette gir typisk dårlig posisjonsestimat. Derfor er det ikke uvanlig å ha minst tre antenna-array med klarsyn til tag-en.

2.2 Quuppa Intelligent Locating System

Quuppa er et finsk teknologiselskap med kjernekompetanse innenfor sanntidslokalisering av blåtann-enheter. Deres flaggskipprodukt, Quuppa Intelligent Locating System (QILS), er en av de mest utbredte kommersielle SLS som har anvendelser sammenfallende med dem presentert i 2.1.1.

QILS har mulighet til å finne posisjon til blåtann-enheter med centimeterpresisjon, ved å bruke noder, kalt locators, plassert strategiske steder i lokaliseringsmiljøet. Hver locator er utstyrt med en sirkulær antenna-matrise, hvor hver antenne kan finne ankomsttiden til et blåtann-signal med høy oppløsning. Som beskrevet i forrige delkapittel, vil det med denne informasjonen være mulig å beregne et estimat for ankomstvinkelen. QILS baserer seg hovedsakelig på AoA for posisjonsestimat, og erfaring viser at lokalisering basert på AoA er spesielt nøyaktig, og robust mot fleirveispropagasjon.

Siden QILS bruker blåtann, og det finnes mange eksisterende blåtann-enheter, finnes det mange potensielle områder hvor QILS kan integreres med lav kostnad. Dette er fordi at QILS støtter tredjepartproduserte blåtann-enheter, i tillegg til Quuppa-produserte tags. Kostnaden av produksjon av blåtann-enheter er også lav, noe som gjør det overkommelig å produsere mange tags.

Hver locator i QILS er koblet med ethernet til en sentral prosesseringsenhet som kjører posisjonseringsmotoren kalt Quuppa Positioning Engine (QPE). QPE tilgjengeligjør en mengde sanntidsdata gjennom et API. Dette inkluderer råposisjonen, filtrert posisjon, usikkerhets-covarians matrise, RSSI fra nærmeste locator og mye mer.

Som en del av denne oppgaven ble et eksemplar av Quuppa Demo Kit (QDK) anskaffet, og data fra API'et ble brukt aktivt. Ytterligere beskrivelse og anvendelse av QDK presenteres i Kapittel 3.

2.2.1 Intelligent Transport Systems

Intelligente Transportsystemer (ITS) er et samlebegrep for avanserte applikasjoner med mål om å tilby innovative tjenester relatert til ulike transportformer og trafikkstyring ved å forsyne brukere med bedre informasjon slik at de kan gjøre tryggere, mer koordinert og smartere"valg i transportnettverket [12].

ITS systemer blir stadig mer populære og den norske regjeringen presenterte ITS som et fokusområdet i fremtidens transport [7].

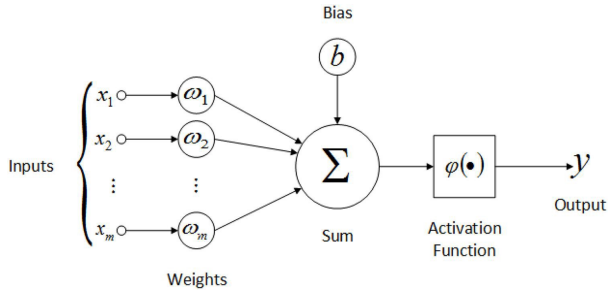
Cooperative Intelligent Transport Systems (C-ITS) er en grein av ITS som sikter på å koble traffikk sammen ved bruk av trådløse kommunikasjonsnettverk, med mål om å øke sikkerheten og effektiviteten gjennom informasjonsdeling og samarbeid mellom trafikanter. Ved Vehicle-to-vehicle (V2V) kan biler dele sin posisjon og hastighet, og med Vehicle-to-infrastructure (V2I) kan kjøretøy få informasjon om friksjonskoeffisienten til veibanen, fartsgrensen, når det blir rødt lys i et lyskryss og mye mer.

Systemet utviklet i denne oppgaven har potensialet til å være en modul av C-ITS.

2.3 Kunstige nevrale nettverk

I denne seksjonen introduseres grunnleggende idéer og konsepter ved kunstige nevrale nettverk og hvordan disse kan relateres til lokalisering.

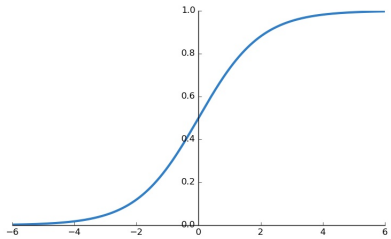
Kunstige nevrale nettverk er en tilnærming for å kunne utføre komplekse beregninger av en prosess, ved bruk av et stort antall enkle sammenkoblede enheter. Denne tilnærmingen er inspirert av aktiviteten til nevroner i hjernen. Den *kunstige nevronen* er modellert som en aktiveringsfunksjon, som gir ut en verdi som en funksjon av den vektede sum av dens innganger. Historisk har det vært mest vanlig å bruke en versjon av en sigmoidfunksjon. En illustrasjon av en kunstig nevron er vist i figur 2.3.



Figur 2.3: Kunstig Nevron

Alle innganger x_i blir multiplisert med henholdsvis vektene ω_i, og summert. En bias b blir lagt til, for å produsere summen Σ. Summen er så argumentet til en aktiveringsfunksjon $\varphi(\Sigma)$. Den mest vanlige aktiveringsfunksjonen er en versjon sigmoidfunksjon, med ligning 2.2 og graf vist i figur 2.4.

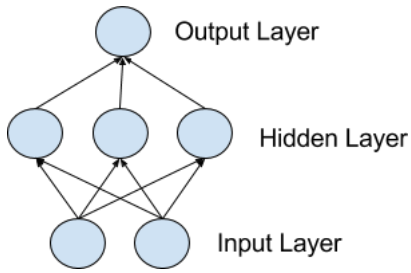
$$\varphi(x) = \frac{x}{1 + e^{-x}} \tag{2.2}$$



Figur 2.4: Sigmoid aktiveringsfunksjon

Denne kunstige nevronen, eller noden, er organisert i lag av noder som er koblet til nodene av det neste og forrige laget. Verdien av signalet som kommer inn i en node er en funksjon av signalene som kommer inn fra det forrige laget, og vektet, som forklart tidligere og vist i figur 2.3. Et nevralt nettverk med tilstrekkelig antall nevroner og en kontinuerlig, avgrenset og ikke-konstant aktiveringsfunksjon, er i stand til å approksimere enhver matematisk funksjon [4][5].

Vektene i nettverket bestemmes ved å minimere en kostfunksjon, gjennom en prosess kalt *gradient descent*, ofte omtalt som *trening* av nettverket. Et eksempel på et nevralt nettverk med ét skjult fullt koblet lag vises i figur 2.5



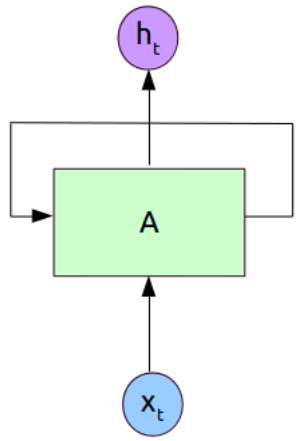
Figur 2.5: Multilayer perceptron network. Et skjult lag med tre noder, hvor deres inngang er de to foregående nodene fra inngangslaget vektet.

Det finnes svært mange konfigurasjoner av nettverk, hvor de spesialiseres til hver sin anvendelse. Konvolusjonelle nevrale nettverk er eksempelvis gode på å finne kjennetegn i et 2D bilde, og kan dermed gjøre gode gjett om hva som befinner seg i bildet. *Recurrent Neural Networks*, tilbakevendende nevrale nettverk, er gode på sekvens prediksjonsproblemer og brukes ofte for å predikere aksjekurser, forstå tale- og tekstdata og ikke minst lokaliseringsproblemer. I denne oppgaven er nettopp et slikt, tilbakevendende nevrale nett-

verk, anvendt og er derfor bli beskrevet ytterlige.

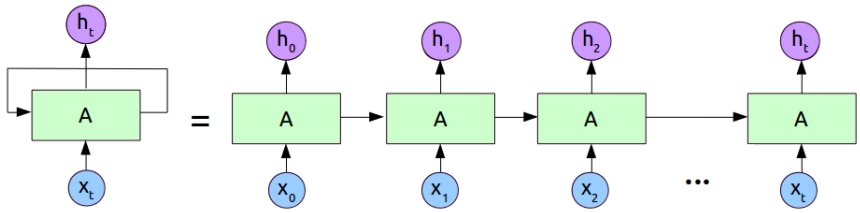
2.3.1 Tilbakevennende Nevrale Nettverk (RNN)

Når mennesker tenker starter vi ikke på en ny tanke hvert sekund. Når vi leser forstår vi hvert ord basert på forståelsen av tidligere ord. Vi bygger våre tanker på tidligere tanker og forstår noe utifra en kontekst. Tradisjonelle nevrale nettverk har ikke denne egenskapen noe som er en stor ulempe. Recurrent Neural Networks adresserer nettopp dette problemet ved å innføre sløyfer slik at informasjon kan vedvare. I neste figur kan vi se ett nevralt nettverk A som tar inn data x_t, og gir ut en output h_t. Men i tillegg blir noe av den invendige tilstanden til A sendt tilbake til inngangen til A. Den mest vanlige konfigurasjonen er å la inngangen til A være sammensetningen av et datapunkt x_t og den forrige outputen til nettverket h_{t-1}.



Figur 2.6: Recurrent Neural Network med sløyfe

Man kan tenke på et tilbakevennende nevrale nettverk som flere kopier av det samme nettverket, hvor hver sender informasjon til etterfølgeren. Her er en figur som viser hva som skjer hvis vi ruller ut nettverket.



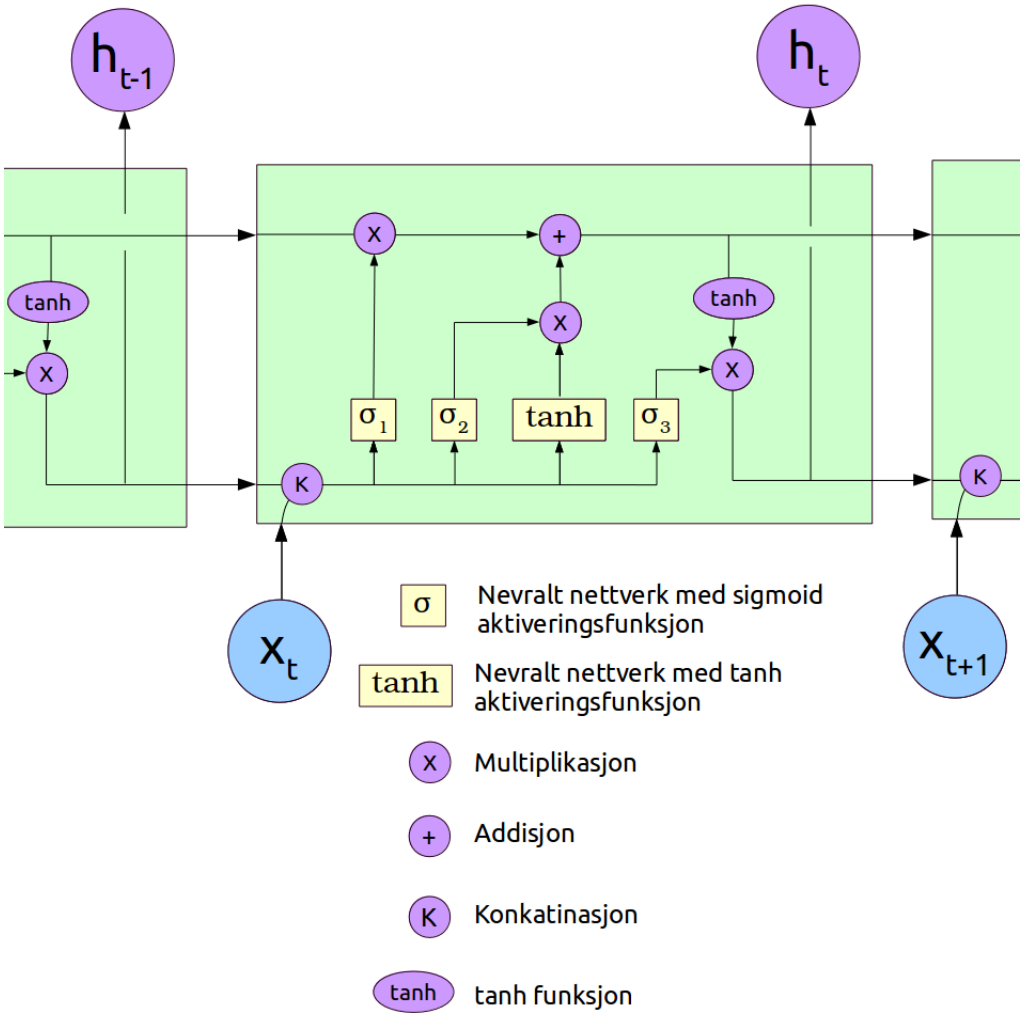
Figur 2.7: Et utrullet recurrent neural network.

Denne lenke-strukturen viser at recurrent neural network er nært beslektet sekvenser og lister. RNN er dermed det naturlige valget av nettverksarkitektur når man jobber med data hvor tolkningen av et datapunkt avhenger av tidligere data.

Utfordringen med den klassiske tilbakevendende nevrale nettverket er at i noen situasjoner vil avhengigheten mellom datapunkter være avskilt med stor avstand. For eksempel i en tekst hvor du skal prøve å predikere neste ord i setningen: '*Jeg snakker flytende ...*'. For å kunne predikere hvilket språk neste ord skal være, trenger vi konteksten fra tidligere i teksten. Om setningen '*Jeg vokste opp i Norge*' kommer mye tidligere enn '*Jeg snakker flytende ...*' vil det være vanskelig for nettverket å huske så langt tilbake. Det er teoretisk mulig at RNN kan håndtere slike langsiktige avhengighetsforhold, dog har det vist seg i praksis å være vanskelig å gjennomføre([3]). Long-short-term memory er en underkategori av RNN som addresserer nettopp dette problemet, og er også nettverksarkitekturen som er benyttet i denne oppgaven.

2.3.2 LSTM

Lang kortsiktig hukommelse, på engelsk 'long short-term memory', er en verjon av Recurrent Neural Network (RNN) som er i stand til å lære langsiktige avhengighetsforhold. Da et ledd i et klassisk tilbakevendende nevralt nett kun inneholder ett nevralt nettverk, vil hvert ledd i et LSTM nettverk inneholde flere. En slik LSTM-celle vises i neste figur.



Figur 2.8: Den repeterende cellen i et LSTM nettverk inneholder fire nevrale nettverk.

Inngangene til cellen kommer fra tre steder. Den første, som er nederst x_t, er neste datapunkt fra datasettet. Inngang to er nede til venstre og er utgangen til forrige LSTM-celle, mens inngang tre er celle-tilstanden og representeres ved den øverste horisontale linjen. Inngang en og to konkateneres og blir sendt videre til fire ulike nevrale nettverk, hver med

sin spesifikke funksjon.

Det første nettverket, sett lengst til venstre, har som oppgave å bestemme hva slags informasjon som kommer til å kaste bort fra celle-tilstanden. Celle-tilstanden representeres som en vektor av verdier, og for hver verdi vil det første nettverket gi ut et tall mellom 0 og 1. 1 representerer 'fullstendig behold denne', mens 0 betyr 'fullstendig kast bort' denne verdien.

Neste steg er å bestemme hvilken informasjon som skal lagres i celle-tilstanden. Dette er en to-stegs operasjon. Første nettverk, kalt 'input gate layer' bestemmer hvilke verdier som skal oppdateres. Det andre nettverket, et $\tanh$ lag, lager en vektor av nye kandidatverdier som kan legges til i celle-tilstanden. I neste steg multipliseres disse to vektorene og adderes til celle-tilstanden.

Så til bestemmes det hva cellen skal gi ut. Utgangen vil være basert på cell-tilstanden, men en filtrert versjon. Først kjøres et nevralt nettverk som bestemmer hvilken del av celletilstanden som skal gis ut. Så, puttes celle-tilstanden gjennom en **tanh**, for å begrense verdiene mellom -1 og 1, og multipliserer så utgangen med sigmoid laget, så det gis ut bare de delene som modellen mener er fornuftig.

Nettverket har nå mulighet til å raskt lære seg ny informasjon, samtidig som det tar i bruk informasjon som ble lært for lenge siden.

2.4 Relatert arbeid

Ettersom presise og raske lokaliseringssystemer er svært etterspurte, er det gjort mye forskning og utvikling på dette området. De aller fleste systemene bruker enten WiFi, blåtann eller Ultra Wide Band (UWB), da disse er de mest utbredte radioprotokollene. Samtidig bruker majoriten av systemer klassiske lokaliseringsteknikker som RSSI, AoA og fingerprinting. De siste årene har det dog vært flere forsøk på å bruke teknikker fra maskinlærings-verden, da disse teknikkene har vist seg å gi gode resultater innenfor andre fagfelt. De mest relevante forskningsarbeid gjennomført på lokalisering ved bruk av maskinlæring presenteres.

I [1] presentes en nyvinnende metode for lokalisering ved bruk av flere nevrale nettverk. De har gjort funn at ved bruk av beacons plassert på mennesker, at orientering av personen har stor påvirkning på RSSI målingene. De utvikler derfor en modell som trener flere nettverk, hvor hvert nettverk er spesifikt for en orientasjon av brukeren. De bruker et kompass under datainnsamling for å måle hvilken orientasjon brukeren har. Nettverket de benytter er et simpelt MLP-nettverk, med ett skjult lag, hvor input er RSSI verdiene som fem ulike basestasjoner måler, og output-vektoren har lik lengde som antall posisjoner systemet kan differansiere. Dette nyutviklede systemet klarte å lokalisere en person til korrekt posisjon med 90% nøyaktighet. Dog er da den store mangelen til denne tilnærmingen at lokaliseringen er bundet til disse posisjonene.

I [13] presenteres 'DeepML', et nyutviklet lokaliseringssystem basert på Deep Long Short-Term Memory (LSTM) (Deep LSTM betyr at i hvert av de nevrale nettverkene, er det mer enn et skjult lag). De bruker magnetisk feltstyrke og lysstyrke som input til systemet. Dette er på bakgrunn av deres observasjoner om at det magnetiske feltet og lysstyrken på hver posisjon er stabilt og robust over tid. For det andre så er magnetisk felt og lyssterke komplementert til hverandre på mange steder. Altså at på noen posisjoner er ikke det magnetiske feltet så ulikt, men da er lysstyrken forskjellig. De skriver at DeepML har 82% av alle testlokasjonen har mindre feil enn 2 m, mens ved bruk av kun magnetisk felt er dette tallet på 50%. De kommenterer at bakgrunnen for deres bruk av LSTM er nettopp nettverkets egenskap til å lære seg long-range avhengigheter. Dette skrivet er en motivasjon for å bruke LSTM for lokalisering.

I [8] er et nytt forskningsprosjekt hvor de bruker en smarttelefons akselerometer som rådata. De bruker et LSTM nettverk til å identifisere ulike handlingsenheter, som venstre sving, høyre sving, normalt steg, kort steg eller langt steg. Rådata fra et akselerometer ifra en gående person vil gi bestemte mønstre. De ulike mønstrene ved å gå rett fram, til venstre, til høyre, langt og kort, er unike, og det er nettopp disse mønstre LSTM nettverket i [14] klarer å detektere. De skriver at grunnen til at de valgte LSTM nettverket er at det klarer automatisk å velge ut effektive kjennetegn fra de forskjellige mønstrene, som muliggjør klassifisering av handlingsenheter.

I [9] benytter de både den klassiske MLP- men også et convolutional neural network for sproring av pasienter på et sykehus. De plasserer ut RSSI noder i hvert rom, og under innhenting av data plasserer de en beacon i et rom og over et sekund måler RSSI verdier. De mapper så verdiene over til et bilde, hvor den fysiske geommetrien til sykehuset, samt målte de RSSI-verdiene bestemte hvilke pixler som skulle lyse hvitt (aktive) og hvilke som skulle forbli svarte (inaktive). Convolutional neural network er spesielt godt til å finne kjennetegn i bilder, og er grunnen til at de bruker det i denne sammenheng. For hvert sekund lager de et bilde, og kjører bildet gjennom en CNN. Så bruker de resultatet av de siste 30 bildene som inngang til et vanlig MLP nettverk, som igjen gir ut en sannsynlighetsvektor for hvert rom. Ulempen med denne metoden er at pasienten må være i samme rom i over 30 sekunder for at nettverket kan gjøre en prediksjon. De addresserer dette problemet med å vise til at et LSTM nettverk muligens kan hjelpe. Merk at i denne konteksten brukes ANN og MLP som begrep for samme nettverksarkitektur.

> "*In addition, the long short-term memory (LSTM) network may be capable of learning better than the ANN used in this study to identify the transitions between each room.*"

Både [13] og [8] bruker LSTM nettverk og argumenterer for styrkene til arkitekturen. De bruker ulike sensorer for lokalisering men klarer likevel å vise gode resultater ved bruk av LSTM. Styrkene som argumenteres for nettverket er noe av bakgrunnen for at det er valgt å bruke et LSTM-nettverk i denne oppgaven. En gjenganger i rapporter om lokalisering er diskretiseringen av posisjonene det er mulig å lokalisere. Det er også vanlig at beacon må være i ro i en betydelig tid for at systemet skal kunne predikere posisjonen. Disse to

typiske manglene forsøkes å adresseres i denne oppgaven.

———————-

Kapittel 3

Metode

I dette kapittelet gis det først en introduksjon av hva som ønskes å oppnå ved kommende eksperimentell oppsett, samt grunnlag for valg av nevral nettverksarkitektur. Deretter vil et design for overordnet testoppsett presenteres, etterfulgt av det eksperimentelle oppsettet. Neste delkapittel vil være datainnsamling hvor realisering av designet datainnsamlingsplattform beskrives. Manipulering og prosessering av data for å skape treningssett begrunnes, før implementasjonen av nettverket utdypes.

Grunnlaget for metoden er at QDK skal brukes, samt at det er ønskelig å samle data i et miljø som ligner et veikryss for å simulere et realistisk scenario av lokalisering av kjøretøy.

I miljøet skal det være enkelte områder med dårlig dekning for Quuppa sine locators, og andre områder med god dekning. Lokaliseringen skal foregå mens tags er i en bevegelse som ligner bevegelsene til et kjøretøy. Dette er fordi systemet er tiltenkt å kunne være en del av et lokaliseringssystem av kjøretøy langs veibanen. Lokalisering av tags som beveger seg på kryss av veibanen vil derfor ikke være en del av oppgaven.

Formålet med å lokalisere tags i områder med varierende dekning av locators, er å simulere et realistisk scenario. Det er ofte slik at i nærheten av veikryss finnes det høye bygninger, som gir uønskede refleksjoner, eller store objekter som gir blindsoner. Formålet er derfor å lage et system som er robust mot slike forstyrrelser i radiomiljøet.

I et reelt scenario vil det være dynamiske forstyrrelser som kjørende biler. Dog vil denne oppgaven kun undersøke lokalisering i et statisk miljø.

For valg av nevralt nettverk er det lagt vekt at nettverksarkitekturen har vist gode resultat på andre tidsserie-prediksjonsproblemer. På bakgrunn av fordelene ved LSTM-arkitekturen presentert i seksjon 2.3.2 og underbygget av artiklene i seksjon 2.4, vil det designes et LSTM nettverk som tar inn data fra Quuppa sitt API, og gir ut et posisjonsestimat.

I tillegg til resultatene fra 2.4, så gis det her flere grunner til at LSTM-nettverket har potensial til å gi bedre posisjonsestimater enn QILS:

1. Siden tags vil bli ført i bevegelser lignende kjøretøy, vil nettverket muligens lære seg slike typiske bevegelsesmønstre. Eksempelvis vil det ikke være noen brå akselerasjoner, og bevegelsene vil for det meste være en-dimensjonale.

2. Det er varierende dekning av locators, noe som resulterer i at posisjonsestimatet fra QILS vil variere i nøyaktighet. I områder med dårlig dekning vil dataen fra API'et være unøyaktig, og mest sannsynlig sprikende. Muligens vil nettverket lære seg kjennetegn for unøyaktige data, og klare å knytte dette opp mot en posisjon. F.eks. hvis API-dataen i posisjon A er slik aHva den målert posisjonen oscillerer mellom to koordinater, vil nettverket muligens lære seg at når dette er tilfellet, så er taggen i posisjon A.

3. Nettverket kan bruke mange tidligere datapunkter, og dermed kunne gjenkjenne en trend i posisjonsendringen, mens QILS estimerer hver posisjon uavhengig.

4. Tags vil ikke forlate veibanen, noe netterverket muligens vil lære seg.

QILS sitt API tilgjengeliggjør en mengde data. Den data som er bestemt skal lagres og føres inn til nettverket er følgende verdier.

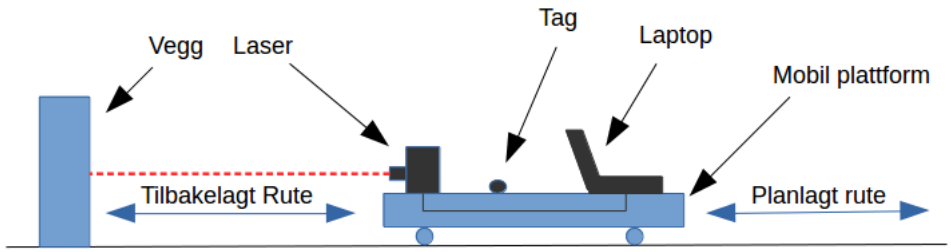
Dataverdi	Datalengde (Antall tall)	Beskrivelse
position	3	x, y, z koordinater
positionAccuracy	1	Estimert nøyaktighet av posisjon. Beskriver radius av usikkerhetssirkel.
covarianceMatrix	4	Usikkerhetselipse i x, y planet
smoothedPosition	3	x, y, z koordinater
smoothedPositionAccuracy	1	Estimert nøyaktighet av smoothedPosisjon. Beskriver radius av usikkerhetssirkel.
kalmanLocation	3	x, y, z koordinater
kalmanVelocityComponent	3	kalman-filtrert hastighet i x, y, z retning

Tabell 3.1: Dataverdier som brukes som inngang til nevralt nettverk

En sample fra QILS gir verdier for alle ovenforbeskrevne dataverdier, og danner det som fra nå av vil bli kalt en *datavektor*.

3.1 Overordnet testoppsett

Hvordan man samler inn, lagrer, og organiserer data er av stor betydning for ethvert prosjekt som bruker 'veiledet læring' (eng: supervised learning) [15]. For hver datavektor som samples fra API'et er det også nødvendig med en 'sann' posisjon, dvs. den virkelige fysiske posisjonen i rommet av taggen. Dette oppnåes ved å lage en mobil plattform, hvor taggen befinner seg, og en laser koblet til en laptop er montert. Det skal så defineres et start- og sluttpunkt hvor den mobile plattform skal bevege seg mellom. En vegg plasseres på startpunktet og plattformen, med laseren på enden, rettes mot veggen. Avstanden til veggen måles så, og siden start- og sluttpunkt er definert, kan man beregne den eksakte fysiske posisjonen til plattformen, og dermed også taggen. Det vil kun bli målt posisjoner i det plattformen er i bevegelse,for å fokusere på lokalisering av bevegende objekter. En illustrasjon av oppsettet vises i figur 3.1.



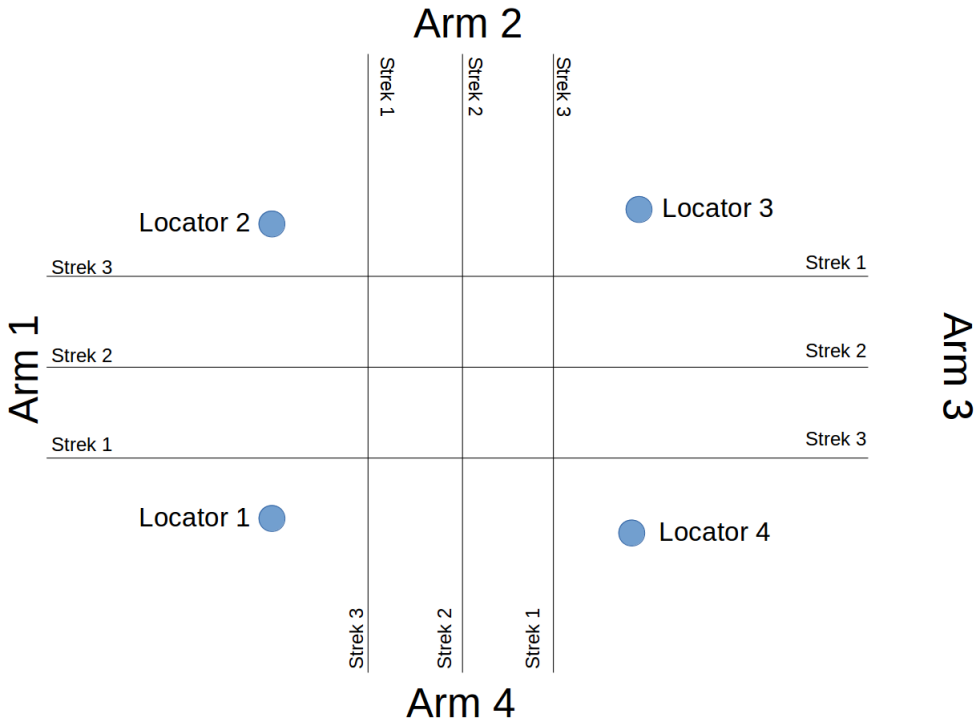
Figur 3.1: Mobil plattform sett fra siden

Denne plattformen skal så bli ført mellom mange start- og sluttpunkter, hvor det i hvert forsøk vil bli plassert en vegg på startpunkt og data samples mens plattformen er i bevegelse. Linjene plattformen skal beveges mellom bør dekke store deler av lokaliseringsområdet for at nettverket skal lære seg de unike forskjellene i miljøet.

Miljøet som brukes er et lager, avbildet i figur 3.2

Figur 3.2: Bilde av lageret, med plasseringen av QILS Locators markert med piler.

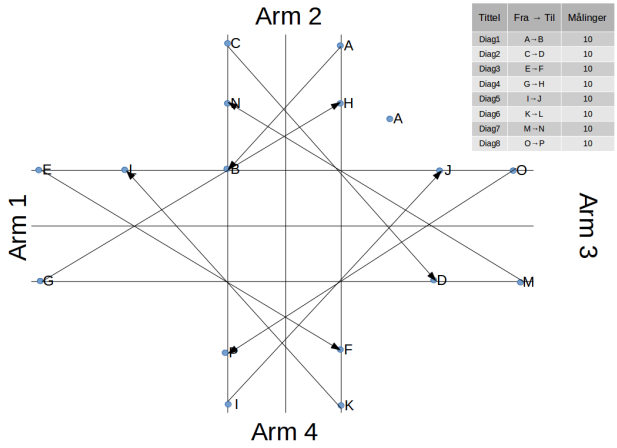
En modell av lageret og oppsettet vises i figur 3.3



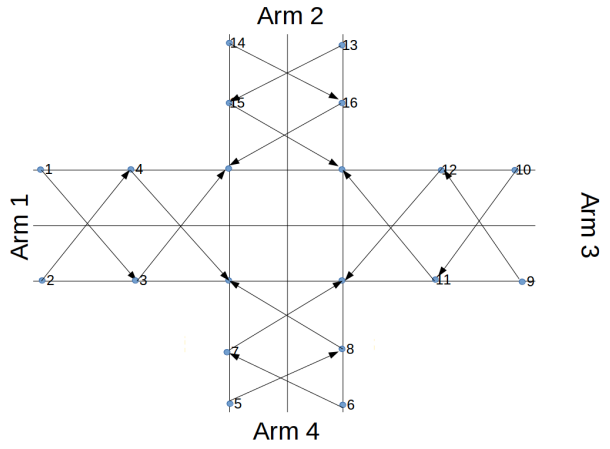
Figur 3.3: Modell av omgivelse. De tre stekene i hver retning representerer grensene i en tiltenkt kjørebane.

Arm 1 i figur 3.3 tilsvarer området mellom hyllene nærmest locator 1 og 2 i figur 3.2. Dvs. at Arm 1 strekker seg til venstre vegg i figur 3.2. Strekene 1-3 er tiltenkt å representere en grensene i en veibane. Strek 2 markerer skille mellom kjørefeltene mens strek 1 og 3 markerer veibanens ytterpunkt.

Ved bruk av den samme modellen vises hvor plattformen er dratt for å samle data.



Figur 3.4: Diagonale linjer hvor plattform er dratt



Figur 3.5: Ytterligere diagonale linjer hvor plattform er dratt

3.2 Eksperimentelt oppsett

Fire locators ble montert i taket, og ethernet-kabler ble routet til et sentralt punkt. I figur 3.2 er det mulig å se at lageret har mye åpent rom på høyre side, mens venstre side er dekket

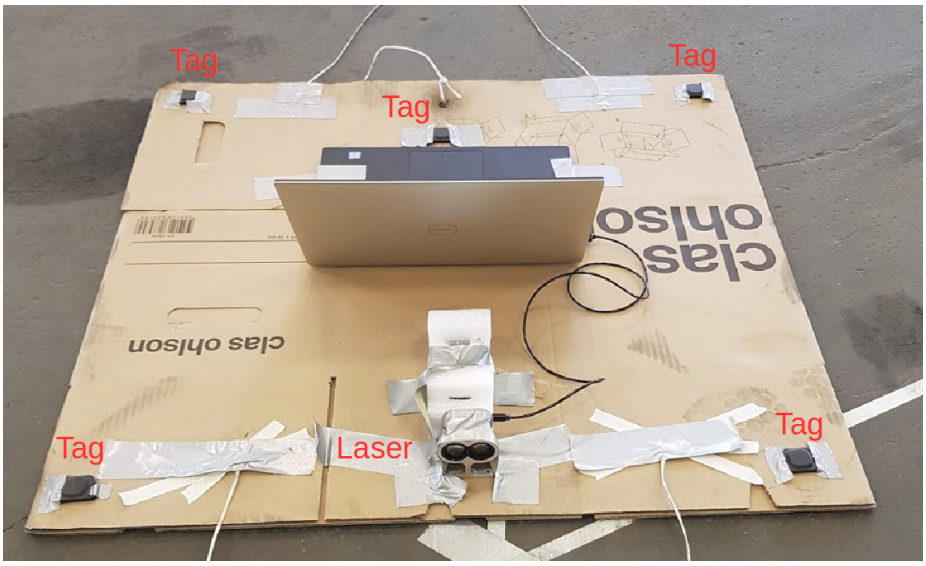
av hyller i metall. Dette er som ønsket, da det var planlagt med områder med varierende grad av forstyrrelser.

De samme linjene som vist i figur 3.3 ble realisert med teip på gulvet på lageret som vist i følgende figur.

Figur 3.6: Fysisk koordinatsystem laget med teip. Representerer de samme linjer som vist i figur 3.3

Linjen Strek 2 fra Arm 1 til Arm 3 og Strek 2 fra Arm 2 til Arm 4 fungerer som aksene i et fysisk koordinatsystem. Disse er av stor betydning da alle fysiske posisjoner, måles utifra dette koordinatsystemet med origo i krysningen mellom de to Strek 2 strekene. Det er utdypet ytterligere hvordan dette koordinatsystemet er utnyttet senere i oppgaven.

Den mobile datainnsamlingsplattformen ble realisert med en papp-plate. På denne er det festet fem tags, laser og en laptop, alle i kjente posisjoner. Grunnen til at det ble brukt fem tags på plattformen er fordi at for at nevrale nettverk skal yte godt er det nødvendig med god og store mengder data. Ved å bruke fem tags femdobles generert treningsdata for hvert forsøk, noe som både sparer mye tid, men også vil føre til økt robusthet mot uregulariteter ved enkelte tags. F.eks. hvis én tag gir svært upresis posisjonsestimat ved et område, mens de andre gir godt, så vil nettverket kunne dempe påvirkningen av denne uregulariteten. På bakgrunn av funnet til [1] at orientasjonen av blåtann-tag har stor påvirkning, ble alle tags orientert likt. Bilde av plattformen vises i figur 3.7.



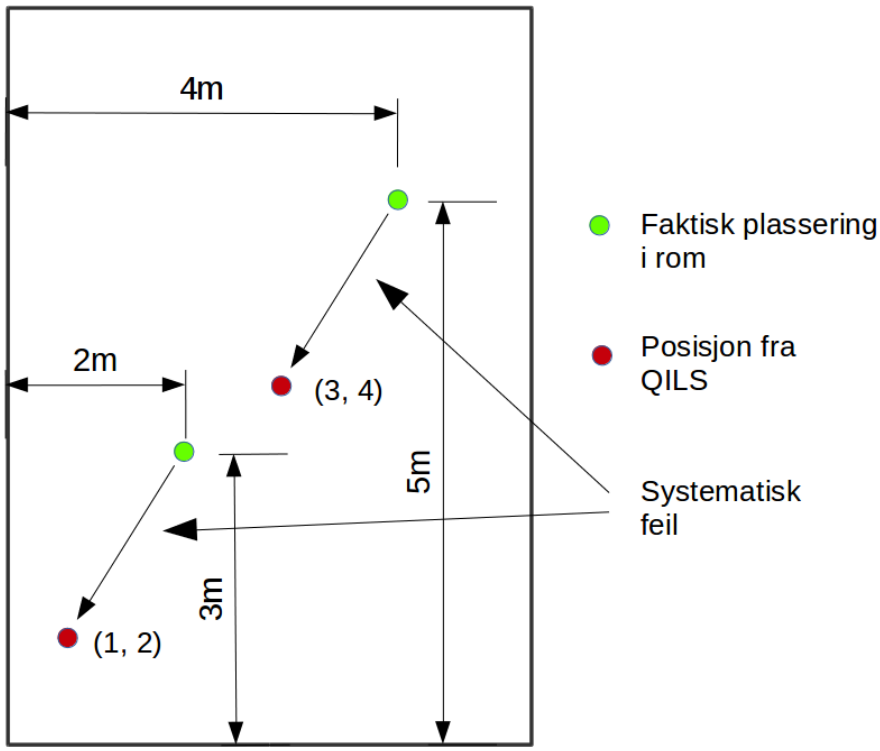
Figur 3.7: Mobil plattform for datainnsamling

Det er festet tau på to sider av plattformen slik at det er mulig å dra plattformen langs gulvet. Denne ble så satt til et startpunkt, pekene mot en perpendikulær vegg. Denne perpendikulære veggen ble realisert ved å bruke en papp-plate festet til siden av en boks, og posisjonen ble målt med laser i det fysiske koordinatsystemet som nevnt tidligere. Boksen vises i figur 3.8.

Figur 3.8: Perpendikulær boks.

Plattformen ble så dratt i en rett linje, markert med tape på gulvet. Da plattformen dras i en rett linje, og både start- og endepunktet er kjent, samt avstanden til den perpendikulære veggen, så kan posisjonen til plattformen beregnes nøyaktig. Som nevnt tidligere vil denne posisjonen fungere som 'sann' posisjon under trening av nevralt nettverk. Under draging av plattformen ble både avstanden fra den perpendikulære veggen samplet, og posisjonsinformasjon fra QILS sitt API. På denne måten genereres treningsdata ved at data fra API'et samples, altså posisjonen QILS estimerer tagsene på plattformen til å være, men også den 'sanne' posisjonen målt med laseren.

En av de større utfordringer med denne oppgaven er å lage en metode for å bedømme nøyaktigheten til QILS og det nyutviklede systemet. QILS sitt API gir ut koordinater, men hvilket punkt representere dette i den virkelige verden? I Quuppa sin software lager man en modell av omgivelsen med plasseringen av vegger, og det er i denne modellen koordinatene gis. I en perfekt modell vil et koordinat (1, 2) representere punktet 1 m fra veggen langs x-aksen og 2 m fra veggen langs y-aksen, gitt at disse veggene står vinkelrett på hverandre. Dog viser det seg i praksis at det kan oppstå unøyaktigheter i denne modellen, noe som innfører feil. Dette vil være et problem da det under datainnsamling er brukt posisjoner(som start og sluttpunkt) definert utifra det det fysiske koordinatsystemet. Denne utfordringen illustreres i figur 3.9

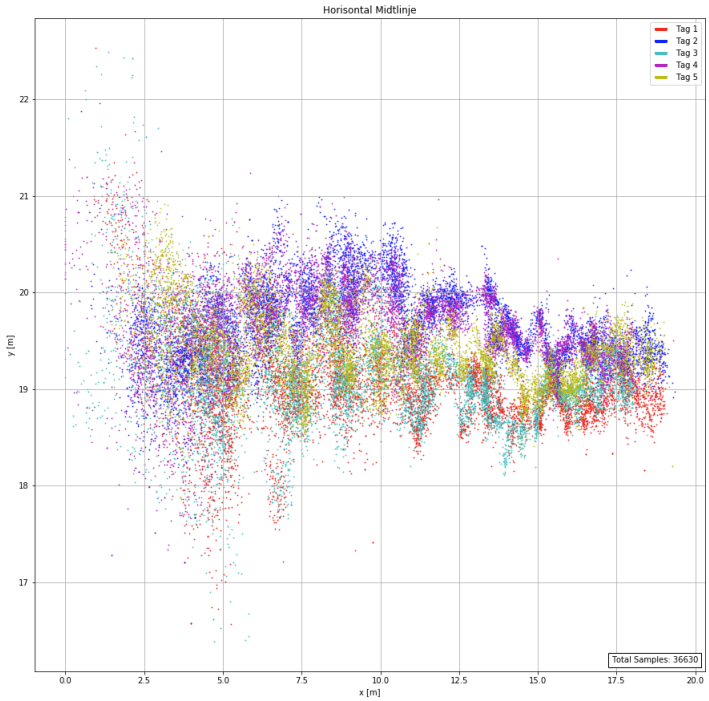


Figur 3.9: Systematisk feil introdusert av unøyaktigheter i model

Utfordringen er at det er ønskelig å ha et felles koordinatsystem for posisjonene fra QILS og posisjonene målt med laseren. Om det oppstår en uoverenstemmelse i disse representasjonene vil det kunne oppstå en systematisk feil. F.eks. i tilfellet ovenfor vil koordinater gitt fra QILS alltid gi en posisjon med lavere x- og y-koordinat enn i et koordinatsystem definert utifra veggene i figuren. Det som kan være litt skummelt er at mest sannsynlig vil det nevrale nettverket kunne lære seg denne systematiske feilen, og under evaluering av nøyaktighet til QILS samenlignet med nettverket, vil nettverket alltid vise til bedre resultater, selv om den ikke tar hensyn til noe temporal data som var hele poenget med nettverket.

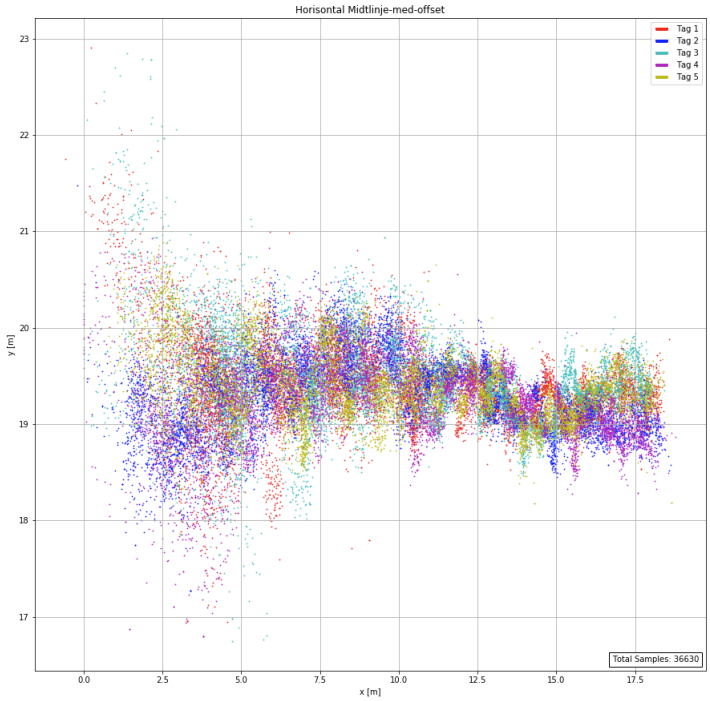
Måten dette problemer er løst på er å definere det fysiske koordinatsystemet utifra en stor mengde målinger fra QILS hvor målingene er i gjort i posisjoner kjent i det fysiske koordinatsystemet. Helt konkret er dette gjennomført ved å føre den mobile plattformen 10 ganger langs Strek 2 fra Arm 2 til Arm 4(horisontal midtlinje), og 10 ganger langs Strek 2 fra Arm 1 til Arm 3(vertikal midtlinje). Posisjonene målt fra de 10 målingene for den

horisontale midtlinjen vises i figur 3.10.



Figur 3.10: Rådata punkter fra QILS ved 10 måler av horisontal midtlinje

Merk at siden hver tag har sin unike posisjon på plattformen vil også posisjonene målt være litt forskjellig. Disse punktene skal brukes til å lage en linje som definerer x-aksen, og det er da viktig å ta hensyn til denne forskjellen. Dette løses ved å legge til den negative offsetten til hver tag i forhold til posisjonen til laseren. Merk at denne offsetten avhenger av orientasjonen av plattformen, som igjen avhenger av start- og sluttpunkt av hvor plattformen har blitt dratt. Alle disse verdiene er kjent, og en offsett er mulig å beregne. Neste figur viser alle punktene transformert.

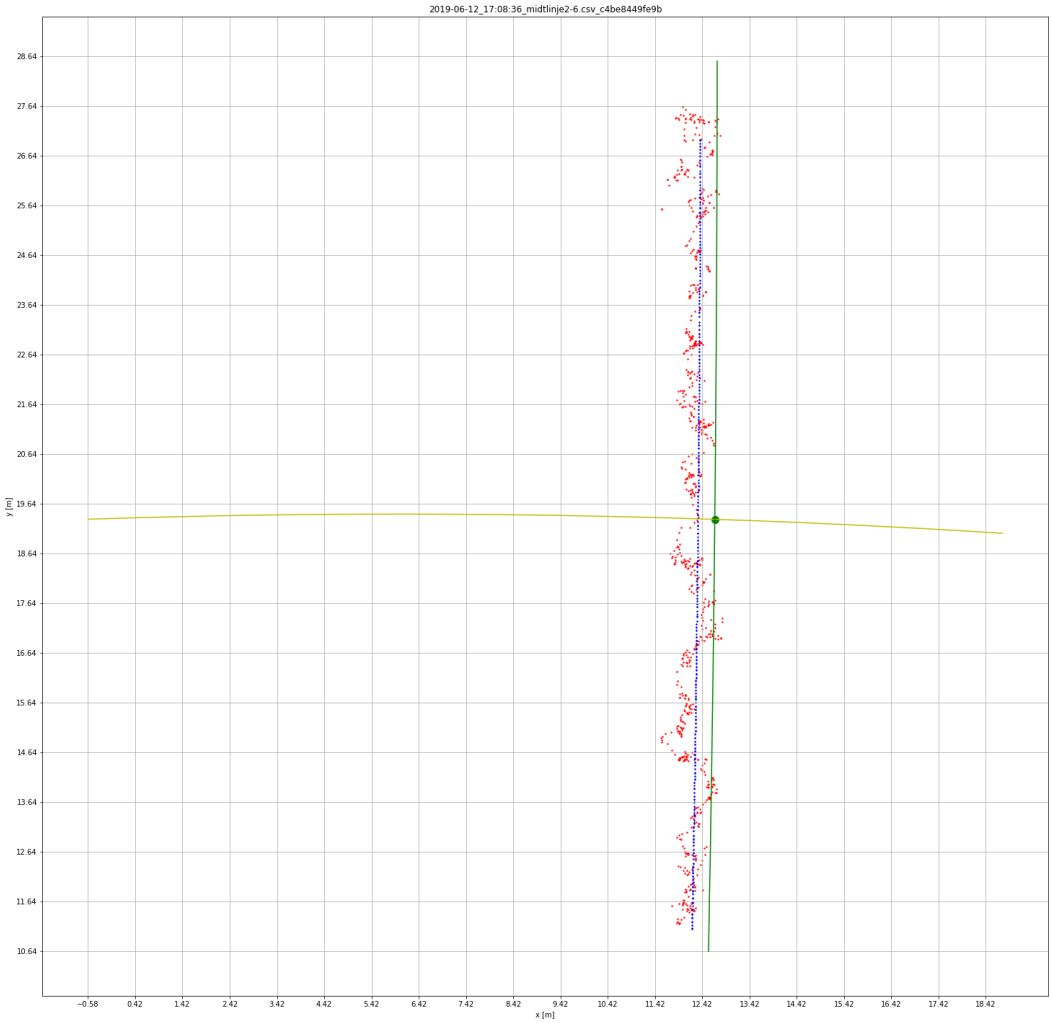


Figur 3.11: Rådata punkter fra QILS ved 10 måler av horisontal midtlinje

I figuren ovenfor kan vi se at punktene ligger mer samlet. I tillegg observerer vi at selv om plattformen er dratt i en rett linje så ser det ikke ut som punktene ligger på en rett strek. Spesielt punktene til venstre, med lav x-verdi ser ut som er spesielt spredt. Dette kommer av at dette området er lengst inn i Arm 1 er mellom to hyller, og dårlig dekning av locators. Vi observerer altså at det er varierende grad av nøyaktighet til posisjonsestimatene til QILS, en antagelse vi tidligere har gjort.

Det er så regnet ut en andregrads regresjonslinje som best passer punktene, og dette andregradspolynomet definerer x-aksen. Samme fremgangsmetode er brukt for den vertikale midtlinjen. Krysningspunktet mellom polynomene definerer origo, og det er dermed mulig å definere enhver posisjon i den fysiske verden i et koordinatsystem som er konsistent med koordinatsystemet QILS benytter.

Et eksempel som viser funksjonen til arbeidet med å definere et konsistent koordinatsystem vises i figur 3.13



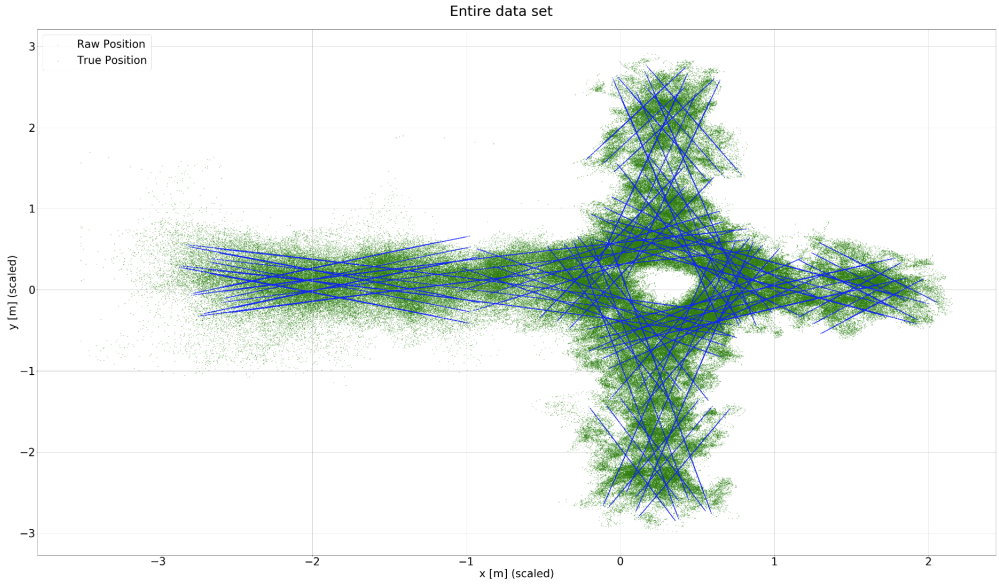
Figur 3.12: Rådata punkter fra QILS, 'sanne' punkter fra lasermålinger i konsistent koordinatsystem

Denne figuren viser målinger fra én tag i det plattformen er dratt langs den vertikale linjen. De røde punktene er posisjoner fra QILS, mens de blå punktene er 'sann' posisjon fra laseren. Den gule og grønne linjen er henholdvis den horisontale- og vertikale aksen laget fra polynomene tidligere beskrevet. De røde punktene ligger noe til venstre for y-aksen, noe som kommer av at taggen er plassert på siden av plattformen. Vi kjenner start- og sluttpunktene ettersom disse er målt utifra teip-koordinatsystemet. F.eks. i dette tilfellet er startpunktet 7.5m langs y-aksen, og 0m langs x-aksen. Posisjonen finnes i det konsistente koordinatsystemet ved å finne punktet langs den vertikale polynomet 7.5m ifra origo og så påføre offsetten til tag'en.

Vi observerer at de røde punktene er noe spredt rundt den sanne posisjonen, med enkelte punkter sterkt avvikende, og en trend i samme retning som den sanne posisjonen.

Merk dog at det er kun presentert posisjonsestimater fra QILS mens det egentlig er samlet en rekke andre datapunkter også, som måler andre variable som f.eks. usikkerhet i posisjonene. Det er altså ikke kun posisjonsestimatene som føres til inngangen av nettverket.

For å samle treningsdata som sikrer at de egenskaper ved ulike områder blir fanget opp, ble plattformen dratt langs 24 start- og sluttpunkt par, som vist i figur 3.4 og 3.5. For hvert par, ble plattformen dratt fra start- til sluttpunkt 10 ganger for å kunne utelukke effekter som inntreffer et fåtall ganger. Følgende figur viser posisjonsestimatene i grønt fra QILS og sann posisjon i blått.



Figur 3.13: Hele treningssettet med posisjonser fra QILS i grønt og sanne posisjoner i blått. X- og y-aksen er skalert.

Dette er et stort treningsett med over 1 000 000 datapunkter. Igjen observerer vi at det er større spredning av posisjoner ved lave x-verdier. Verdiene i figuren er skalert, noe som kommer av behovet for et nevralt nettverk for å få normaliserte og skalerte verdier. Det er samplet langt flere posisjoner enn vist ovenfor, men kun punkter samplet i det plattformen er i bevegelse er brukt. Dette er for å fokusere på posisjonering av et bevegende objekt.

3.3 Preprosseringer av data og implementasjon av nevralt nett

Formatet av dataen som føres inn i et netterk er av stor betydning for ytelsen. Det er derfor gjort betydelig innsats i å organisere dataen på en et format som lar nettverket trene enkelt.

Alle laser-målinger ble samplet til en .csv fil, hvor hver fil inneholder data fra der plattformen har blitt dratt fra et start- til et sluttpunkt. Underveis av dragingen av plattformen ble data fra QILS sitt API samplet til en separat fil. Dette utgjør rådataen, og etterfølgende manipulering av dataen følger i kronologisk rekkefølge.

1. Finne tidsintervaller hvor plattformen er i bevegelse, og beholde laser-målinger fra denne perioden
2. For hvert laser-datapunkt, finne datavektoren fra QILS som er nærmest i tid til da laser-datapunktet ble samplet
3. Sammenslå data fra laser og datavektor fra QILS til én datavektor
4. Bruke alle datapunkter i det plattformen er dratt langs horisontal midtstrek, og lage andregrads polynomligning, som tidligere beskrevet
5. Gjenta forrige punkt for punktene langs den vertikale midtstrek
6. Bruke de to forrige polynomiligningene til å definere et nytt koordinatsystem
7. For hver datavektor i den sammenslåtte datafilen fra punkt 3, blir den sanne posisjon beregnet vhp. kjent start- og sluttpunkt, avstanden målt med laseren, samt polynomfunksjonene beregnet i punkt 3. og 4. Start- og sluttpunkt assosiert til hver lasermåling er lagret i en separat fil

Resultatet av disse stegene er for hver laser-.csv fil en assosiert ny fil inneholdene både laser-data og data fra QILS for hvert tidssteg, og sann posisjon beregnet utifra kjent start- og sluttpunkt. Da det er 20 start- og sluttpunkt par, og for hvert par er plattformen dratt 10 ganger, og det er fem tags på plattformen resulterer dette i $20 * 10 * 5 = 1200$ filer med data.

Siden programmeringsspråket Python er det mest populære innenfor datavitenskap, ble dette språket benyttet. En av grunnene for språkets populæritet er at det finnes en rekke ulike maskinlæringsbibliotek. Tensorflow ble valgt på grunn av at det finnes store mengder dokumentasjon og utviklingsmiljø. Det betyr at det finnes mange gode ressurser for teknikker og fremgangsmetoder for implementasjon av nevrale nettverk. Tensorflow muliggjør også lavnivå sammenkobling av kunstige nevroner. Dette gjør at man som utvikler har god kontroll på den eksakte dataflyt i nettverket.

Et klassisk LSTM nettverk, som vist i figur 2.8, ble implementert uten noen betydelige avvik i dataflyten fra figuren og den forutliggende beskrivelsen. Under trening av nettverket er det dog en rekke parametre som må bestemmes, og har stor betydning på resultet.

Siden datasette var svært stort, ble kun deler av det brukt under trening. Først ble mesteparten av datasettet kuttet bort. Av det gjenværende datasettet ble 70% definert som treningssett, og 30% testsett. Under trening, trenes nettverket kun på treningssettet, mens under evaluering tester man også ytelsen på testsettet. Dette er for å unngå *overfitting*, som er et tilfelle hvor nettet blir for spesialisert på akkurat treningssettet, og unnlater å generalise.

Da et LSTM-nettverk er basert på tidsseriedata, er en av de viktigste parametre det som kalles *window size*. Window size er antall tidligere datavektorer nettverket skal fores med før det gir ut en prediksjon. Ved å velge en høy verdi av window size har nettverket

mulighet til å lære seg langsiktige avhengighetsforhold. Ulike verdier for window size ble testet.

Batch Size er antall vinduer av data som fores til nettverket før oppdatering av gradienten. Det er ikke noen konkrete rettningslinjer på hvordan verdien settes, så igjen ble ulike verder testet.

Som tidligere nevnt består en LSTM-celle av fire ulike nevrale nettverk. Hver av disse nettverkene er et klassisk MLP nettverk, med ett skjult lag. Antall skjulte noder i det skjulte laget er også en parametre som kan settes. Antall skjulte noder har stor betydning på kompleksitet og treningstid til nettverket. Flere ulike antall skjulte noder ble prøvd, dog ble det raskt observert at selv om antallet skjulte noder doblet, så ble ikke prediksjonene nevneverdige bedre. Det ble derfor prioritert å trene nettverket for det meste med kun 512 skjulte noder. Med få noder var det ikke lenger nødvendig med store mengder treningsdata. I de aller fleste tilfeller, ble kun 1% av den totale treningsdataen brukt, noe som var uventet.

En 'epoch' betyr at det nevrale nattverket blir trent ved å gå gjennom treningssettet én gang. Det ble observert at trening med 25 til 30 epochs var tilstrekkelig, da det raskt ble observert at prediksjonsfeilen avtok. Ytterligere trening kan medføre 'overfitting' hvor nettverket blir for spesialisert på datasettet den trener på, og vil yte dårligere på et testsett.

Følgende loss function ble brukt.

$$\text{loss} = \sum_{i=1}^{n}((x_{p_i} - x_{l_i})^2 + (y_{p_i} - y_{l_i})^2) \tag{3.1}$$

For å evaluere ytelsen til ovenforpresentert system, ble en simpel algoritme utviklet. Denne regner ut kvadratisk gjennomsnitt.

$$\text{KG} = \frac{1}{N}\sum_{i=1}^{n}((x_i - x_{l_i})^2 + (y_i - y_{l_i})^2) \tag{3.2}$$

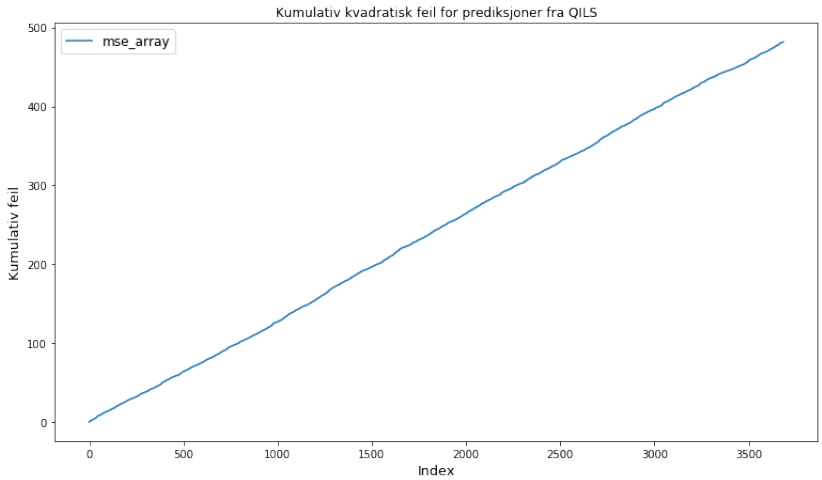
Kapittel 4

Resultater

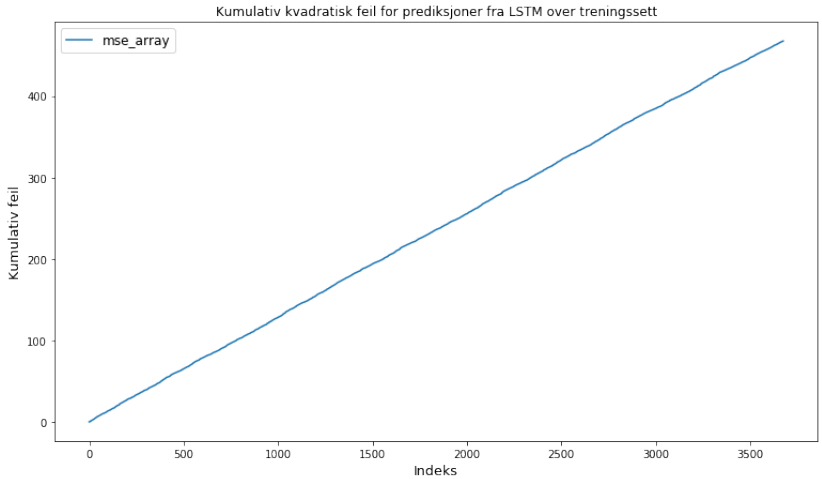
Flere nevrale nettverk har blitt trent med ulike hyperparametre for å undersøke hvilke kombinasjoner som gir best resultat. For hver trening har nettverket produsert en liste med prediksjoner, og det kvadratiske gjennomsnitt mellom prediksjon og sann posisjon har blitt beregnet. Enkelte nettverk yter bedre enn andre, og her er noen utvalgte bli presentert for å formidle sensitiviteten til ulike hyperparametre. Det nettverket som har gitt best resultater er presentert til slutt.

I etterfølgende figurer og tabeller er ulike enheter for ytelse presentert. Kvadratisk gjennomsnitt, forkortet RMS (eng: Root-squared-error), er den viktigste måleenheten for ytelse. Lavere RMS betyr lavere gjennomsnittlig avstand mellom estimert posisjon og sann posisjon. Kumulativ kvadratisk feil er summen av den kvadratiske avstand mellom estimert posisjon og sann posisjon. Denne vil øke med antall estimat som blir undersøkt.

Det vil følgelig først bli presentert en evaluering av nøyaktigheten til posisjonsestimatene til QILS. I figur 4.1 vises kumulativ kvadratisk feil for posisjonsestimatene fra QILS. Tilsvarende figur for prediksjonsnøyaktigheten til en konfigurasjon av LSTM nettverket vises i figur 4.2



Figur 4.1: Kumulativ kvadratisk feil for posisjonsestimatene fra QILS over treningsettet.



Figur 4.2: Kumulativ feil for posisjonsprediksjoner fra LSTM-nettverket over treningssettet.

Vi observerer at den kumulative feil for prediksjonsestimatene fra QILS og LSTM nettverket over det samme treningssettet, er nærmest identisk. Algoritmen som regner ut kvadratisk gjennomsnitt, til henholdsvis Quuppa og LSTM nettverket gir følgende resultater:

Kvadratiske gjennomsnitt mellom posisjonsestimatene fra QILS og sann posisjon:

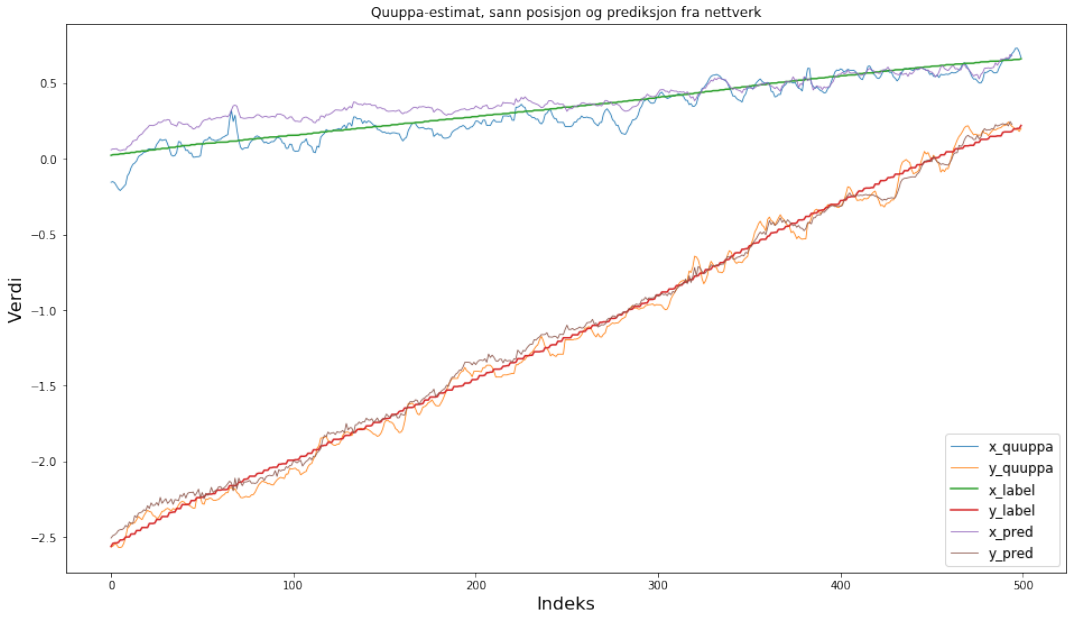
$$\text{RMS}_{Quuppa} = 0.00596 \tag{4.1}$$

Kvadratiske gjennomsnitt mellom posisjonsestimatene fra LSTM nettverket og sann posisjon:

$$\text{RMS}_{LSTM_1} = 0.00588 \tag{4.2}$$

RMS_{LSTM_1} er kvadratisk gjennomsnitt for én av konfigurasjonene av LSTM nettverket. Igjen observerer vi at verdiene er svært like.

For å få en bedre intuitiv forståelse av oppførselen til nettverket, inkluderes en figur som viser de tre viktigste posisjoner i denne oppgaven; posisjonsestimatene fra QILS(xy_quuppa), prediksjonene fra LSTM nettverket(xy_pred) og sanne verdier (xy_label).



Figur 4.3: X- og y-verdier for posisjoner fra Quuppa, sanne og prediksjoner fra nettverket.

I denne figuren er de to rette strekene x- (grønn) og y- (rød) verdiene til de sanne posisjoner. Disse er regnet utifra laseren og vi ser at posisjonen endrer seg gjevnt med ingen oscillasjoner. Derimot er posisjonsestimatene fra QILS og LSTM nettverket noe oscillerende

rundt de sanne posisjonene. Det er vanskelig se noe betydelig forskjell på QILS og LSTM nettverket.

Nøyaktighet til QILS og ulike konfigurasjoner av LSTM nettverket har blitt beregnet på ulike datasett. Antall skjulte noder (SN) og vinduslengde (VL) i LSTM nettverket er de to mest sentrale hyperparametre, og har dermed blitt inkludert i tabellen nedenfor. Tabellen oppsumerer de viktgiste resultat.

Estimatmetode	Datasett	Skjulte noder	Vinduslengde	#Samples	Kumulativ kvadratisk feil	Kvadratisk gjennomsnitt
Quuppa	Treningssett	n/a	n/a	3684	481	0.005959
LSTM	Treningssett	512	7	3676	467	0.00588
Quuppa	Testsett	n/a	n/a	1579	210	0.00917
LSTM	Testsett	512	7	1575	215	0.00932
Quuppa	Hele datasett	n/a	n/a	526166	69561	0.000501
LSTM	Hele datasett	512	7	526155	71469	0.000508
LSTM	Treningssett	512	20	3570	808	0.00792
LSTM	Treningssett	1024	7	3675	640	0.00689
LSTM	Testsett	1024	7	1575	287	0.01074

Tabell 4.1

Kvadratisk gjennomsnitt for posisjonsestimatene fra QILS og LSTM nettverket er i alle tilfeller svært like. De varierer noe utifra hvilket datasett som undersøkes, noe som også er forventet da f.eks. data fra områder med store forstyrrelser gir større prediksjonsfeil. Det mest sentrale funnet er at kvadratisk gjennomsnitt for hele datasettet er nærmest indentisk mellom QILS og LSTM nettverket. Dette antyder at ytelsesforskjellen mellom de to metodene er svært liten.

Med de forutliggende resultater kan vi ta slutningen at LSTM nettverker har klart å lære å predikere posisjoner, men dog ikke oppnådd et bedre posisjonsestimat enn fra QILS. En grundigere diskusjon om grunnen til dette vil følge i neste kapittel. Konfigurasjonen av det best-ytende nettverket er nettverket med 512 skjulte noder og vinduslengde på 7.

Kapittel 5

Diskusjon

En viktig hyperparametre er vinduslengden, altså hvor mange tidligere datavektorer skal nettverket bruke på å predikere en posisjon. Antagelsen er at posisjoner fra QILS vil oscillere rundt den sanne posisjonen, og ved å tilføre temporal data vil nettverket ha mulighet til å lære seg langsiktige avhengighetsforhold. De endelige resultatene tilsier dog at nettverket ikke gjorde dette. Nettverket ble trent både med vinduslengde på 7 og 20, men nærmest ingen forskjell i ytelse. Derimot økte tiden det tok å trene nettverket betraktelig, og det vil også påføre tidsforsinkelser ved bruk av nettverket, da flere addisjoner og multiplikasjoner må finne sted før et estimat kan gjøres. Dette vil være negativt for et system som skal brukes for biler, da det er ønskelig med liten tidsforsinkelse.

Det mest overraskende er nettverkets behov for mindre treningsdata enn først antatt. I nettverket med best ytelse er kun 1% av hele datasettet brukt for å trene opp nettverket. Denne impliserer at det er mulig å utvikle et mer komplisert nettverk, og fortsatt ha nok treningsdata. En interessant undersøkelse er implementasjonen av ytterligere skjulte lag i de fire nevrale nettverkene. I [13] bruker de et Deep LSTM nettverk og viser til gode resultat. Generelt har fagfeltet innenfor Deep Learning, hvor de bruker mange skjulte lag, eksplodert de siste årene mye på grunn av disse nettverkenes evne til å produsere enestende resultat. Deep Learning krever vanligvis store datasett, men det eksisterer i dette tilfellet.

Metodikken har også noen svakheter. Start- og sluttposisjonene er målt med en laser, og innfører feilkilder som ikke er kvantifisert og heller ikke tatt hensyn til. Om f.eks. et startpunkt har blitt målt til en posisjon 1m ifra sann startpunkt, så vil treningsdataen bli korrupt og nettverket vil prøve å korrigere for dette. Under trekning av papp-plate vil også enkelte feil innføres da trekningen ikke er fullstendig lang en linje.

Den største svakheten til systemet er at det er basert på kun den data som er tilgjengeliggjort av QILS sitt API. Dette begrenser seg til posisjonsestimat og usikkerheten deres, og det vil derfor være fundamentalt utfordrende å produsere et betydelig bedre posisjonses-

timat. Det mest optimale er å bruke rådata fra QILS sine locators. Om dette hadde vært mulig ville samme metodikk som har blitt brukt i denne oppgaven kunne bli anvendt, og på grunn av LSTM-nettverkets egenskap til å huske langsiktige avhengigheter vil den muligens ha kunne gitt bedre posisjonsestimater.

Kapittel 6

Konklusjon

Målet med denne oppgaven var å undersøke om et nevralt nettverk med minne, som baserer seg på data fra QILS sitt API, kan lage et posisjonsestimat som overstiger nøyaktigheten til det originale posisjonsestimatet fra QILS. I tillegg til det nevrale nettverket ble en metodikk for datainnsamling og databehandling utviklet. Dette inkluderer oppsett av utstyr fra Quuppa og realisering av en mobil plattform for datainnsamling. Et nødvendig steg i databehandlingen var å lage et felles koordinatsystem som både posisjonesestimatene fra QILS og sanne posisjoner, målt med laser på mobil plattform, kunne representeres i.

Kvadratisk gjennomsnitt viser at posisjonsestimatfeilen til QILS over det fullstendige datasettet, er 0.000501, mens feilen for nettverket, over samme datasett, er på 0.000508. Disse resultatene impliserer at posisjonsestimatene for QILS og nettverket er nærmest identiske, og nettverket oppnår derfor ikke økt nøyaktighet.

Det ble observert at små endringer i nettverkets parametre medførte store endringer i ytelsen. Videre arbeid er derfor nødvendig for å utforske ytterligere hvilke parametre som kan forbedre ytelsen, samt se på muligheten for at andre nettverksarkitekturer kan gi bedre resultat. En av de største svakhetene med systemet benyttet i denne oppgaven er at det baserer seg på data tilgjengeliggjort fra QILS sitt API. Disse dataene, begrenser seg til posisjonsestimat og usikkerheten av disse. Selv om nettverket ikke oppnår en forbedring av posisjonsestimatet, vil metodikken for datainnsamling og databehandling alikevel kunne anvendes ved nye forsøk med rådata fra locators.

Bibliografi

[1] Altini, M., Brunelli, D., Farella, E., and Benini, L. (2010). Bluetooth indoor localization with multiple neural networks. In *Proceedings of the 5th IEEE International Conference on Wireless Pervasive Computing*, ISWPC'10, pages 295–300, Piscataway, NJ, USA. IEEE Press.

[2] Babu, P. (2016). 10 airports using beacons to take passenger experience to the next level.

[3] Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166.

[4] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4):303–314.

[5] Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251 – 257.

[6] https://quuppa.com/ (2019).

[7] https://www.regjeringen.no/no/dokumenter/meld-st 26-20122013/id722102/ (2019).

[8] Hussain, G., Jabbar, M. S., Cho, J.-D., and Bae, S. (2019). Indoor positioning system: A new approach based on lstm and two stage activity classification. *Electronics*, 8(4).

[9] Iqbal, Z., Luo, D., Henry, P., Kazemifar, S., Rozario, T., Yan, Y., Westover, K., Lu, W., Nguyen, D., Long, T., Wang, J., Choy, H., and Jiang, S. (2018). Accurate real time localization tracking in a clinical environment using bluetooth low energy and deep learning. *PLOS ONE*, 13(10):1–13.

[10] Marek, J. and Štěpánek, L. (2010). Accuracy and availability of the satellite navigation system gps. In *15th Conference on Microwave Techniques COMITE 2010*, pages 121–124.

[11] Sadowski, S. and Spachos, P. (2018). Rssi-based indoor localization with the internet of things. *IEEE Access*, 6:30149–30161.

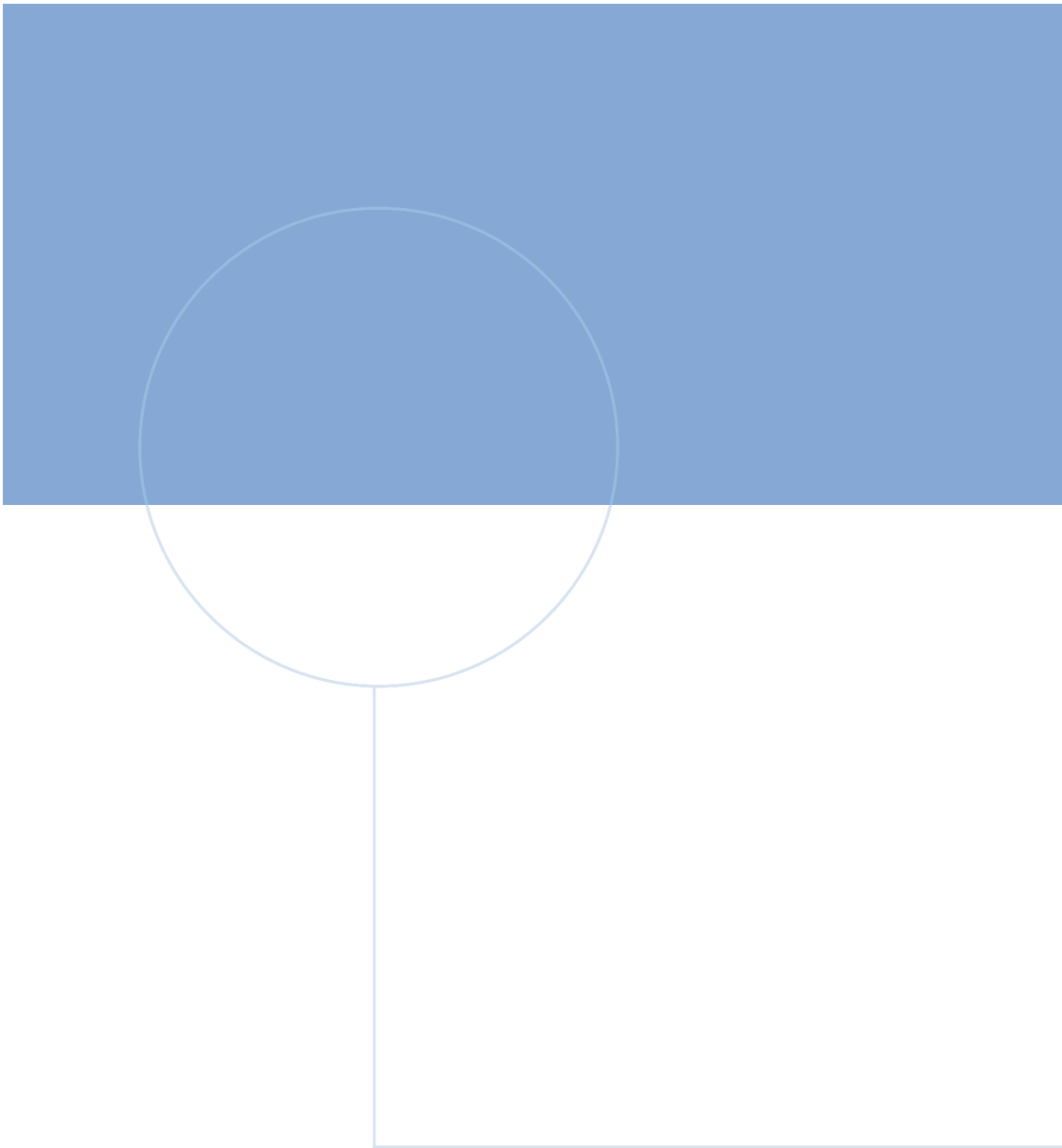
[12] Shaheen, S. and Finson, R. (2013). *Intelligent Transportation Systems*.

[13] Wang, X., Yu, Z., and Mao, S. (2018). Deepml: Deep lstm for indoor localization with smartphone magnetic and light sensors. pages 1–6.

[14] Wikipedia (2019). Received signal strength indication — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Received%20signal%20strength%20indication&oldid=897319913`. [Online; accessed 19-July-2019].

[15] Yu, L., Wang, S., and Lai, K. K. (2006). An integrated data preparation scheme for neural network data analysis. *Knowledge and Data Engineering, IEEE Transactions on*, 18:217– 230.

[16] Zhang, De-Yi, Wang, Wen-Yue, and Lv, Lian-Rong (2017). Research on algorithm of indoor positioning system based on low energy bluetooth 4.0. *ITM Web Conf.*, 11:03007.



NTNU

Kunnskap for en bedre verden