Høgskulen på Vestlandet

BACHELOROPPGAVE

Dokumentasjon og presentasjon av kulturminner
Documentation and presentation of cultural heritage

Landmåling og eiendomsdesign
Institutt for byggfag

23.05.2018
Antall ord: 9682

Veileder: Stig F. Samnøy
Christoffer Heldal Helgesen, Gjermund Wold, Rune Kjærland, Trygve Waaktaar Åsheim

Vi bekrefter at arbeidet er selvstendig utarbeidet, og at referanser/kildehenvisninger til alle kilder som er brukt i arbeidet er oppgitt, jf. Forskrift om studium og eksamen ved Høgskulen på Vestlandet, § 1

Dokumentasjon og presentasjon av kulturminner

Landmåling og eiendomsdesign
Institutt for byggfag
Landmåling

2018

Veileder: Stig F.Samnøy

Christoffer Heldal Helgesen, Gjermund Wold, Rune Kjærland, Trygve Waaktaar Åsheim

Forord

Denne bacheloroppgaven er skrevet som avslutning på utdanningen i Landmåling og eiendomsdesign ved Høgskolen på Vestlandet. Oppgaven ble ferdigstilt våren 2018 på Campus Bergen.

Vi valgte oppgave med bakgrunn i personlig interesse, noe som var kilde til ekstra inspirasjon under arbeidet. Prosessen har vært svært omfattende og tidkrevende. Arbeidet har bydd på utfordringer som har gitt stort læringsutbytte og nye erfaringer.

Spesiell takk til veilederen vår, høgskolelektor Stig Frode Samnøy, for god veiledning og støtte under arbeidet. Vi vil også takke kontaktpersonen vår hos Riksantikvaren, Anders Olsson, for nyttig informasjon. Videre vil vi takke eier av Kaupanger stavkirke, Fortidsminneforeningen, og kirkeverge Jorunn Merete Haukås-Eide for tilgang til stavkirken. Til slutt vil vi takke kirketjener Eirik Asbjørn Thue for hans tilgjengelighet under det praktiske arbeidet.

Mai, 2018

Christoffer H.Helgesen

Gjermund Wold

Rune Kjærland

Trygve Waaktaar Åsheim

Sammendrag

Kulturminner er sårbare for ytre påvirkninger. Ved dokumentasjon kan kulturminnene bevares for ettertiden. Tradisjonelt har ikke hovedformålet med å samle inn og lagre disse dataene vært formidling. Formidlingen av eksisterende 3D-data om kulturminner er helt i oppstartsfasen. Eksisterende 3D-data blir ikke formidlet i stor skala gjennom nåværende plattformer på internett. Teknologi som WebGL gjør det mulig å fremvise 3D-modeller i alle moderne nettlesere. På datainnsamlingssiden muliggjør SFM-teknologien å lage 3D-modeller fra bilder tatt med digitalkamera. Det er derfor interessant å utvikle en metode for å fremskaffe 3D-data av kulturminner som kan presenteres i en GIS-basert webløsning.

Som et case-studie ble laserskanning og fotogrammetri brukt til å samle inn data og lage 3D-modeller av Kaupanger stavkirke. Stavkirken ble laserskannet innvendig og utvendig. Bilder ble tatt av kirken utvendig med digitalkamera. Gjennom datautveksling fikk vi tilgang til en punktsky generert med bilder tatt med drone. Kvaliteten på punktskyene fra fotogrammetrien ble målt opp mot punktskyen fra laserskanneren.

Resultatet fra oppgaven viser at det kan utvikles en webapplikasjon for fremvisning kulturminnedata i form av 3D-modeller, bildet og tekst. For å oppnå effektiv innlastingstid måtte den teksturerte polygonmodellen reduseres vesentlig. Laserskanningen produserte den mest detaljerte punktskyen med minst støy. Bildene fra skanneren gav dårligere grunnlag for teksturering enn bildene tatt med Nikon D600 digitalkamera. Resultatet fra dronen gav vesentlig dårligere data for å modellere stavkirken, sammenlignet med laserskanneren og digitalkameraet.

Abstract

Cultural heritage sites and objects are vulnerable to external influences. By documentation, cultural heritage may be preserverd for posterity. Traditionally, the main purpose of collecting and storing these data have not been dissemination. The dissemination of excisting 3D-data is in its early phases. Existing 3D-data is not widely disseminated through current platforms on the internet. Technology like WebGL makes it possible to display 3D-models in all modern browsers. SFM technology enables the creation of 3D-models from images taken with a digital camera. It is therefore interesting to develop a method of obtaining 3D-data of cultural heritage that can be presented in a GIS-based webapplication.

As a case study, laser scanning and photogrammetry were used to collect data and create 3D - models of Kaupanger stave church. Both the interior and exterior of the church was laserscanned. Images were taken of the church`s exterior with a digital camera. Through data exchange we gained access to a pointcloud generated from images taken by a drone. The quality of the pointclouds from the photogrammetry were measured against the pointcloud from the laserscanner.

The result of this bachelor's thesis shows that a webapplication can be developed for the purpose of displaying cultural heritage data in the form of 3D-models, images and text. In order to achieve effective loading time, the textured polygon model had to be significantly reduced. The laserscanner produced the most detailed pointcloud with the least amount of noise. The images from the scanner gave a poorer basis for texturing than the pictures taken with the Nikon D600 digital camera. The result from the drone survey gave significantly poorer data to model the stave church, compared to the laser scanner and digital camera.

Begrepsforklaring

GIS (Geografisk infomasjonssystem) er et system for lagring, håndtering og presentasjon av geografiske data (Ørstavik, 2015). I en bred tolkning av begrepet inngår følgende komponenter: nettverk, maskinvare, programvare, romlige data, prosedyrer og mennesker (Carver mfl., 2011, s. 19).

GIS-basert webapplikasjon defineres i denne oppgaven som en webapplikasjon der hovedformålet er å presentere geografiske data. Konkret for denne oppgaven vises det til kartet som en sentral del av applikasjonen.

Kulturarv er en samlebetegnelse på det historiske immaterielle og materielle fundamentet samfunnet er bygget på. Kulturarven er viktig da den legger grunnlaget for identitet og skaper forståelse for betydningen av etablerte tradisjoner (ABM-utvikling m.fl., u.å).

Kulturminner er en viktig del av kulturarven og defineres i Kulturminneloven (1978, § 2 første ledd) som "..alle spor etter menneskelig virksomhet i vårt fysiske miljø, herunder lokaliteter det knytter seg historiske hendelser, tro eller tradisjon til." Med dette kan kulturminner sies å utgjøre den materielle delen av kulturarven.

Webapplikasjon: En webapplikasjon er et klient-server program der kommunikasjonen mellom klient og server foregår over internett. Klientsiden av programmet blir kjørt i en nettleser og sender forespørsler til serveren som svarer med data (Rosen og Shklar, 2003, s. 5).

Innholdsfortegnelse

Tabelliste

Figurliste

1 Introduksjon

Ideen til denne oppgaven hadde sitt opphav på Hack4no, et arrangement i regi av Statens Kartverk og Høgskolen i Sørøst-Norge (HSN). Blant annet holdt Riksantikvaren, representert ved Anders Olsson, et foredrag om åpne kulturminnedata. Riksantikvaren satt nemlig på store mengder data om kulturminner som de ønsket å formidle på en engasjerende måte.

I samråd med veileder, Stig F. Samnøy, kom vi frem til at en løsning som kunne presentere kulturminnedata på web, der tradisjonelle geodata kombineres med fremvisning av 3D-modell kunne være en interessant problemstilling.

1.1 Bakgrunn

Kulturminner er en del av den norske kulturarven. Kulturminner vitner om historisk aktivitet og kan gi grunnlag for en identitetsforståelse. Historien kulturminnene dokumenterer kan også ha verdi for vitenskapelige disipliner som blant annet arkeologi og antropologi. Kulturminner er sårbare for naturkatastrofer, krig, utbygging og økonomiske prioriteringer. For å bevare kulturminnene for dagens og fremtidige generasjoner må de dokumenteres.

Lovverket er et virkemiddel for å gjøre dette. Verneverdien til kulturminner utrykkes i kulturminnelovens formålsparagraf. I paragrafens andre ledd kan vi lese at "Det er et nasjonalt ansvar å ivareta disse ressurser som vitenskapelig kildemateriale og som varig grunnlag for nålevende og fremtidige generasjoners opplevelse, selvforståelse, trivsel og virksomhet." (Kulturminneloven 1978, § 1 andre ledd).

Riksantikvaren er et statlig organ med ansvar for forvaltning av kulturminner. En av organisasjonens oppgaver er å gjøre kulturminnedata tilgjengelig for bruk i samfunnet. «Kulturminnesok.no» er en plattform til formidling av kulturminnedata. Tjenesten henter data fra Riksantikvarens kulturminnedatabase, Askeladden (Riksantikvaren, u.å.a). Brukere kan også bidra med informasjon til «Kulturminnesok.no».

"K-lab" var et samarbeid om digital formidling mellom Arkivverket, Kartverket og Kulturrådet fra 2012 til 2015. Samarbeidet har hatt fokus på hvordan åpne data kan gi muligheter for viderebruk. I Riksantikvarens digitaliseringsstrategi for 2018 – 2021 er ett av målene å videreutvikle dette samarbeidet og jobbe for å gjøre alle kulturminnedata tilgjengelig (Riksantikvaren, 2018, s.17). Handlingsplanen utrykker samtidig en del utfordringer i forvaltningen av store datasett og 3D-data. En annen utfordring er avklaring av eier- og ansvarsforhold. Målet innen 2020 er å klargjøre hvordan dataene skal tilgjengeliggjøres (Riksantikvaren, 2018, s.17).

Tilgjengeliggjøring av kulturminnedata i 3D er helt i oppstartsfasen (jf. Riksantikvarens digitaliseringsstrategi for 2018 – 2021). Formålet med innsamlingen av disse dataene har hovedsakelig vært dokumentasjon. Formatet og størrelsen på dataene gjør hensiktsmessig deling utfordrende. Eksisterende 3D-data blir ikke formidlet i stor skala gjennom eksisterende plattformer som «Kulturminnesok.no».

Nettleserene har i de senere årene fått mulighet for 3D-fremvisning, deriblant igjennom WebGL som støttes i alle moderne nettlesere (Angel og Shreiner, 2015, s. 109). SFM-teknologi gjør det raskt å innhente 3D-data sammenlignet med bare noen få år tilbake. Et konkret eksempel på dette er Telemark Fylkeskommune som har brukt plattformen

Sketchfab.com til å formidle en 3D-modell av Heddal stavkirke (Sketchfab, u.å.). Modellen ble laget på grunnlag av bilder tatt med digitalt kamera. Dette illustrerer at kompetent personale innen forvaltningen av kulturminner selv kan dokumentere et kulturminne, lage en 3D-modell og formidle denne til et større publikum. Prosessen er kostnadseffektiv og tidsbesparende.

Vi ønsket med dette å se på muligheten for å kombinere tradisjonell presentasjon av kulturminner på nett, slik som Kulturminnesøk.no, med visning av 3D-modeller, slik som Sketchfab.com. For å teste innsamling av data valgte vi også å gjennomføre et case-studie med innsamling av 3D-data til et kulturminne til bruk i resten av oppgaven.

1.2 Studieobjekt

Etter samtale med Anders Olsson, seniorrådgiver hos Riksantikvaren, valgte vi Kaupanger stavkirke som mulig studieobjekt. Fra før er kirken bygningsmessig oppmålt og dokumentert på den tradisjonelle måten (Bjerknes, Liden og Tschudi-Madsen, 1975). For Riksantikvaren var det dermed interessant å få dokumentert stavkirken med moderne metoder.

Kaupanger stavkirke ligger omlag en mil sørøst fra Sogndal (Storsletten, 1993, s. 178). Selve kirken ligger innerst i Almabukten, ikke så langt unna Kaupanger hovedgård. Terrenget rundt kirken er stort sett kupert, med et skogkledd område på vest- og sørsiden. Kirken er en av de 28 stavkirkene i landet som stammer fra middelalderen (Storsletten, 1993, s. 17). Kirken er automatisk fredet (Riksantikvaren, u.å.b), men blir fortsatt brukt som sognekirke (The Stave Church Portal, u.å).

1.3 Problemstilling

«Kulturminnesok.no» formidler kulturminner i bilder og tekst med kartvisning. Å kombinere disse funksjonene med visning av 3D-modeller skaper en mer helhetlig formidling av alle tilgjengelige kulturminnedata. I denne oppgaven vil vi utarbeide en metode for å kombinere disse funksjonene i en GIS-basert webapplikasjon.

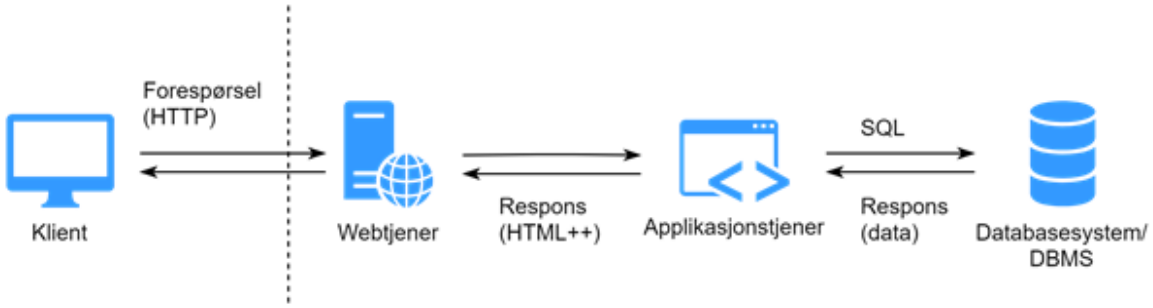
Vi fremsetter en hypotese om at høyoppløselige 3D-data fra kulturminner kan fremskaffes både ved bruk av bilder og laserskanner, og presenteres i en webløsning.

Problemstillingen for denne oppgaven er todelt. Første del omhandler utarbeidelse av en GIS-basert webapplikasjon for fremvisning av 3D-modeller, bilder og tekst. Andre del omhandler innsamling av 3D-data om kulturminner til produksjon av en 3D modell, som kan vises på webapplikasjonen.

2 Teori

2.1 Webapplikasjon

En webapplikasjon muliggjør fremvisning av kulturminnedata på nett. Selve webapplikasjonen er et klient-server program som kjører i nettleseren. Nettleseren sender forespørsler til serveren, som igjen genererer innhold som blir sendt tilbake til nettleseren. Begrepet er svært sammenfallende med «nettside» og de kan ofte brukes om hverandre. Det er likevel gjort forsøk på å skille begrepene fra hverandre. I boken "Web Application Architecture", blir de adskilt ved at en nettside typisk vil dele statisk innhold, slik som rene HTML-sider. En webapplikasjon vil ofte bli sett på som mer dynamisk, der brukeren kan endre og hente opp nytt innhold fra serveren etter at siden er lastet (Rosen og Shklar, 2003, s. 5).



Figur 1: Enkel webapplikasjon.

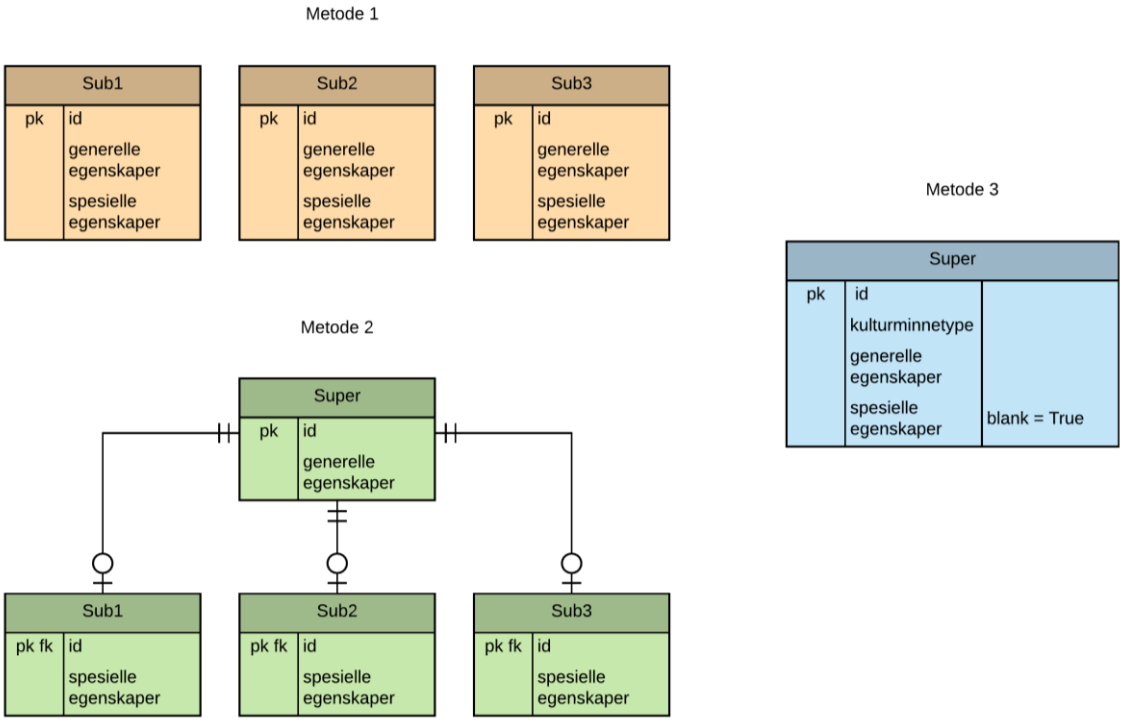
Figur 1 viser et enkelt oppsett på en webapplikasjon. Nettleseren sender forespørsler (http requests) til en webtjener (webserver) som er koblet til nettet via et domene. Forespørslene blir videresendt til applikasjonstjeneren der de blir tolket. Ved behov gjør applikasjonstjeneren spørringer mot databasesystemet, som oftest i form av SQL. SQL-spørringene kan endre innholdet i databasen, eller be om å få data tilsendt i retur. Applikasjonstjeneren genererer deretter en respons til klienten bestående av HTML-, CSS- og JavaScript-koder, i tillegg til dataobjekter i format som XML eller JSON. Webtjeneren, applikasjonstjeneren og databasesystemet ligger på en eller flere fysiske servere som også trenger et operativsystem i bunn. Til sammen utgjør disse komponentene en «utviklingsstakk» (Kristoffersen, 2016, s.318-321).

En vesentlig del av den GIS-baserte webapplikasjonen omhandler hvordan dataene om kulturminnene skal lagres. I dette kapitlet blir det gjennomgått noen sentrale konsept for opprettelsen av databasestrukturen.

Kulturminnedata kan blant annet lagres som punkt, linje og polygon i en objektrelasjonell database. En objektrelasjonell database er en utvidelse av en relasjonsdatabase (Kristoffersen, 2016, s. 416-417). I en tradisjonell relasjonsdatabase lagres alle data i tabeller. Hver kolonne i en tabell representerer en egenskap og hver rad en samling data knyttet til en entydig identifiserbar enhet (Bratbergsengen, 2017). En relasjonsdatabase begrenser hver rad til kun å lagre en verdi i hver celle. Dette er et problem i GIS-sammenheng, der det blant er ønskelig å

lagre et ubestemt antall punkt som representasjon av en polygon. Løsningen var å utvikle relasjonsdatabasen til å kunne lagre objekttyper. Et objekt kan inneholde nøstede rader, som tillater lagring av flere verdier i samme kolonne, slik som koordinatverdiene til et polygon (Kristoffersen, 2016, s. 416-418).

Kulturminner som gruppe inneholder flere ulike underkategorier. For eksempel er både en stavkirke og en gravhaug et kulturminne. Forholdet mellom hovedgruppen og undergruppen kan omtales som arv. Tabellstrukturen i en relasjonsdatabase er i utgangspunktet ikke egnet for håndtering av arv. Dette kan likevel løses på tre ulike måter som har hver sine fordeler og ulemper (Kristoffersen, 2016, s. 204-206):



Figur 2: Metoder for håndtering av arv i en relasjonsdatabase.

Den første fremgangsmåten (se Metode 1, figur 2) er å opprette egne tabeller for hver subtype der hver tabell inneholder egenskapene som hele datasettet har til felles, i tillegg til egenskaper som kun gjelder denne subtypen. Supertypen blir i dette tilfelle implisitt, og en ulempe med denne tilnærmingen er at man ikke kan gjøre spørringer mot det samlede datasettet (supertypen).

Den andre fremgangsmåten (se Metode 2, figur 2) er å gi alle de generelle egenskapene i en felles tabell. Det blir deretter opprettet en egen tabell for hver enkelt subtype der de spesielle egenskapene kan legges til. Denne metoden medfører en oppstykking av data for et objekt inn i flere tabeller, men beholder muligheten til å gjøre koblinger mot det samlede datasettet.

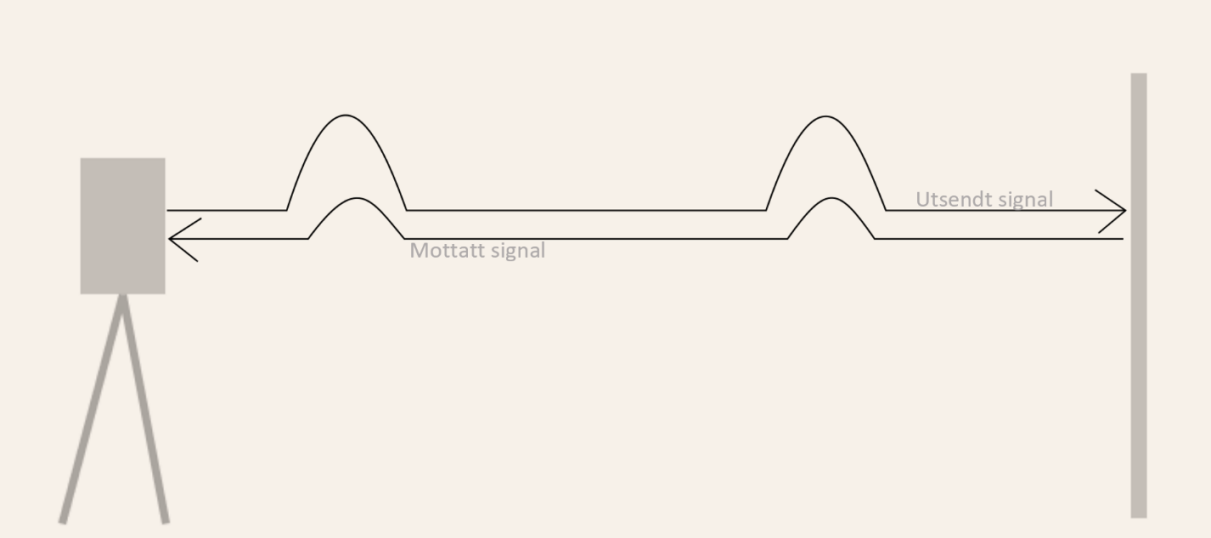
Den tredje fremgangsmåten (se Metode 3, figur 2) er å legge alle objektene i en felles tabell som har et felt som sorterer på subtype. De egenskapene som ikke deles av alle subtyper må ha muligheten til å stå tomme. En slik fremgangsmåte kan være fornuftig dersom det er få egenskaper som kun gjelder noen av subtypene. Hvis det er mange egenskaper som er spesielle for de ulike subtypene, kan dette resultere i en uoversiktlig databasestruktur.

2.2 Laserskanning

Laserskanning, også kjent som LiDAR (Light Detection and Ranging), er mye brukt til blant annet dokumentasjon av kulturminner (Vosselman og Maas, 2010, s. 271). Dette på grunn av metodens evne til å samle inn 3D-koordinater med høy nøyaktighet. Den tredimensjonale punktskyen kan brukes til overvåking og restaurering av kulturminner som en stavkirke (Gustavsen, 2009, s.15). En laserskanner kan måle flere tusen punkt per sekund og samler dermed inn mer data på kortere tid enn ved bruk av konvensjonelt utstyr som totalstasjon. Punktskyen som samles inn kan brukes til blant annet å lage en digital 3D- modell, men også til visualisering, overvåking og restaurering.

For å regne ut et punkts x-, y- og z-verdi benyttes både avstanden til objektet, horisontalvinkelen og vertikalvinkelen til objektet. Avstanden finnes ved å måle tidsforskjellen mellom utsendt og mottatt signal eller endringer i bølgefasen. I tillegg må utgangsvinkelen være kjent.

Laserskannere kan være fasebaserte eller pulsbaserte. Faseskannere benytter forskjellen i bølgefasene for å beregne avstanden til objektet. En faseskanner har høy datarate, men nøyaktigheten reduseres når avstanden øker (Vosselman og Maas, 2010, s. 8). Pulsbaserte skannere finner avstanden til et objekt ved hjelp av lyshastigheten. Denne metoden kalles «Time of flight» og går ut på å måle tiden mellom utsendt og mottatt puls (Vosselman og Maas, 2010, s. 3).



Figur 3: Pulsbasert laserskanning.

Pulsskannere kan ha nøyaktighet på rundt en halv centimeter (Vosselman og Maas, 2010, s. 37). Leica P20 er en pulsskanner som har oppgitt nøyaktighet på 3 mm ved 50 m og 6 mm ved 100 m (Leica ScanStation P20: User Manual, 2014, s. 130).

Avstanden (ρ) beregnes ved:

$$\rho = \frac{c}{n} \cdot \frac{\tau}{2}$$

Der c er lyshastigheten, n er en refraksjonsindeks og τ er gangtida (Vosselman og Maas, 2010, s. 3).

Nøyaktigheten til målinger gjort med laserskanner påvirkes blant annet av intensiteten til retursignalet som igjen påvirkes av innfallsvinkel, avstand, atmosfæriske forhold og reflektivitet (Boehler og Marbs, 2003, s. 6).

Når avstanden øker vil også fotavtrykket til signalet bli større. Laserstrålens diameter er nemlig ikke konstant men øker med avstanden, kalt stråledivergens. En perfekt kollimert stråle ville ikke divergert med økt avstand men på grunn av diffraksjon skjer ikke dette (Vosselman og Maas, 2010, s. 12).

Kulturminner kan være objekter med irregulære overflater. Ved laserskanning av slike objekter kan det oppstå kanteffekter. Når strålen treffer et objekt med en kant, vil deler av strålen kunne reflekteres av bakenforliggende objekt (Boehler og Marbs, 2003, s. 3). Avstanden som beregnes blir da feil siden retursignalet blir reflektert fra et objekt i bakkant.

Albedo er en betegnelse på flaters evne til å reflektere lys (Skaar, 2018). Svarte overflater har svak refleksjonsevne, mens hvite overflater har sterk refleksjonsevne (Boehler og Marbs, 2003, s. 6). Overflater med dårlig refleksjonsevne vil gi et svakt retursignal og motsatt for overflater med sterk refleksjonsevne. Noe som igjen påvirker hvor mange signaler som når mottakeren.

Atmosfæriske forhold vil også påvirke styrken på retursignalet og igjen nøyaktigheten. Avhengig av merke og modell vil laserskanneren ha en spesifisert ytre temperaturområde den jobber best under. Trykk og temperatur vil kunne påvirke forplantingen til lysstrålen (Vosselman og Maas, 2010, s. 12).

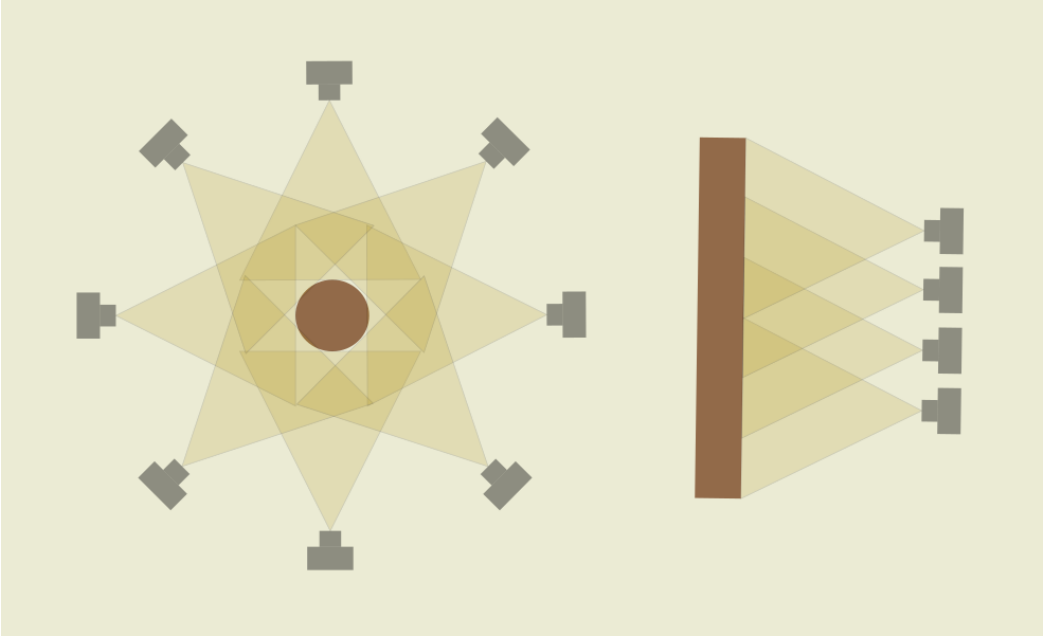
2.3 Fotogrammetri

En stor del av den nyeste utviklingen av fotogrammetrisk programvare bruker Structure-From-Motion (SFM) tilnærmingen. Sentrale komponenter i slik programvare bygger blant annet på professor David Lowe's SIFT-algoritme (Scale Invariant Feature Transform) publisert i 1999 (UBC, u.å.). Algoritmen er bygget for å kunne gjenkjenne objekter og detaljer i bilder, og ved dette finne fellespunkter i et bildesett. Her kan tilfeldige feil i form av ukorrekt "match" av bilder oppstå i løp av prosessen, og det kan derfor være fordelaktig å sortere ut uklare bilder i forkant (Bedford, 2017, s. 13).

Med SIFT som utgangspunkt blir det gjennom en iterativ prosess mulig å gjøre en samtidig beregning av kameraets kalibreringsparametere (indre orientering). Dette i tillegg til orientering og retning i rommet (ytre orientering), som er relativt til objektet som avbildes. Denne prosessen refereres til som "Bundle adjustment" eller på norsk; "strålebuntutgjevning" (Dick, 2018). Systematiske feilkilder knyttet til indre orientering kan ved dette kompenseres for. Dette åpner ikke bare for bruk av ordinært kamerautstyr, men da prosessen kjøres for hvert bilde blir det også mulig å variere bruk av utstyr og zoom i samme bildesett (Bedford, 2017, s. 8). Feilkalkulerte kalibreringsparametere kan imidlertid oppstå og for å minimere sannsynligheten for dette kan kalibreringsparameterne holdes konsistente, for eksempel ved bruk av et objektiv med fast brennvidde (Bedford, 2017, s. 20).

Når SFM-delen av prosessen er unnagjort og kameraets optiske karakteristikk og relativ posisjon av bildene er beregnet, er det klart for rekonstruksjon av en tettere punktsky ved hjelp av en tilnærming kalt multi-view stereo (MVS). Det er flere algoritmer tilgjengelig for utførelse av dette, og løsningen vil derfor variere fra program til program. (Bedford, 2017, s. 9) Resultatet er en punktsky i en vilkårlig skala. Det vil ofte være ønskelig å oppnå 1:1 skalering da dette gir grunnlag for videre målsetting og analyse. Dette kan oppnås ved å legge inn lengdemål mellom tydelige avgrensninger på objektet eller på/mellom noe i nærheten med kjent avstand. Skal modellen sammenstilles med andre data må det også sørges for at den kan plasseres inn i den aktuelle referanserammen. Det er vanlig å løse dette ved å fordele kontrollpunkt jevnt ut over det aktuelle området. Disse kan så måles inn og tildeles koordinater i den aktuelle referanserammen (Bedford, 2017, s. 8 og 43). Feil knyttet til innmålingen vil påvirke punktskyens nøyaktighet (Bedford, 2017, s. 13)

Generell fotograferingsstrategi for terrestrisk fotogrammetri er å fotografere med en overlapp på rundt 60% (Matthews, 2008, s. 29). Skissering av metoden slik det fremgår av Bedford sine anbefalinger er gjengitt i figur 4. Det er i tillegg anbefalt å fotografere slik figuren viser langs to eller flere høydeplan. (Bedford, 2017, s. 13 og 38).



Figur 4 Generell fotograferingsstrategi for objekt (venstre) og fasade (høyre)

Selv om SFM-tilnærmingen ikke stiller spesielle krav til utstyr vil kvaliteten på resultatet likevel svare til kvaliteten på datafangsten. Godt utstyr og grunnleggende kunnskap om bruk vil derfor være fordelaktig. Generelt vil et DSLR (Digital Single-Lens Reflex) -kamera gi bedre resultater enn et kompaktkamera, og et kompaktkamera bedre resultater enn et mobilkamera. Oppløsningen spiller en vesentlig rolle, men vel så viktig er størrelsen på sensoren som skal fange opp lyset. Når denne sensoren er på størrelse med den typiske filmen brukt i analoge kamera (35mm), kalles dette "fullformat". Disse sensorene vil ha bedre evne til å fange lys, og vil produsere bilder med mindre støy (Bedford, 2017, s. 22).

Som ved vanlig fotografering vil en optimal eksponering for fotogrammetri være et balansert samspill mellom blenderåpning (f-verdi), lukkerhastighet, og bildesensorens lysfølsomhet (ISO). Generelt vil en lav ISO gi mindre støy, høy lukkerhastighet bedre skarphet, og lav

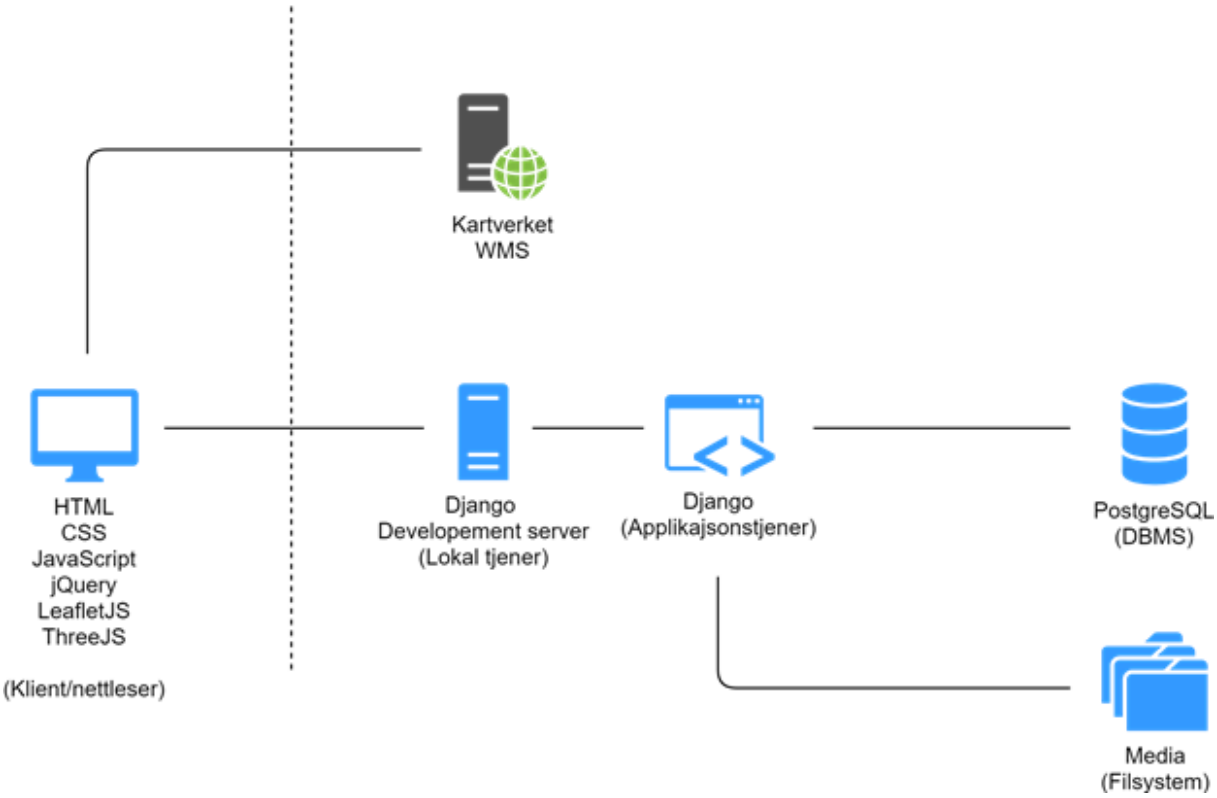
blenderåpning bedre dybdeskarphet. Innstillingene må vurderes i hvert enkelt tilfelle etter forholdene på stedet (Bedford, 2017, s. 19).

Ved utendørs fotografering vil lysforholdene være skiftende. Klarvær gir et stort spenn mellom solsiden og skyggesiden av et objekt. Dette er utfordrende da undereksponerte skyggepartier og overeksponerte høylys vil danne et dårlig grunnlag for rekonstruksjonen. Stadig skiftende lysforhold vil også kunne gi problemer da metoden baserer seg på å finne likhetstrekk i bildene. Et bedre utgangspunkt er å fotografere i overskyet vær, da et tett skydekke vil gi et såkalt flatt lys, hvor lysspredningen i større grad hvisker ut skillet mellom sol- og skyggesider (Bedford, 2017, s. 27).

Den lagrede informasjonen i de ulike bildeformatene spiller også en stor rolle. Det såkalte RAW-formatet er den digitale utgaven av negativene fra analoge kamera, og er det formatet som gir mest fleksibilitet med tanke på postprosessering. Selv om alle kamera gjør opptak i RAW, vil et billig kamera umiddelbart konvertere til et JPEG-format hvor informasjonen i vesentlig grad komprimeres. Bruk av RAW-formatet i fotogrammetrisammenheng vil ha klare fordeler med hensyn på å hente frem informasjon i bildene. Deretter må bildene konverteres til et format som SFM-programvaren aksepterer som input. Ofte vil JPEG her være fordelaktig med tanke på effektiviteten av den videre prosesseringen. Mindre komprimerte rasterformater er også en mulighet som varierer med det aktuelle programmet. (Bedford, 2017, s. 31)

3 Utarbeidelse av webapplikasjon

En oversikt over det endelige oppsettet til webapllikasjonen vises i figur 5. På den fysiske serveren kjører applikasjonstjeneren Django. Lagring av data skjer i en PostgreSQL-database og i tillegg et filsystem hvor større mediafiler blir lagret, slik som 3D-modeller og bilder. Klienten får sendt HTML-, CSS- og JavaScript-kode i tillegg til de eksterne JavaScript-bibliotekene jQuery, Leaflet og ThreeJS. Leaflet gir støtte for visning av et WMS-kart, som blir hentet fra Kartverket sine servere. Grunnet den begrensede tiden, har vi kun valgt å kjøre en lokal tjener med Django sin integrerte «Development server». Dette er et verktøy for å teste applikasjonen under utvikling, før man laster den opp på en server koblet til nettet (Django, 2018, s. 15).



Figur 5: Overordnet oppsett av webapplikasjon.

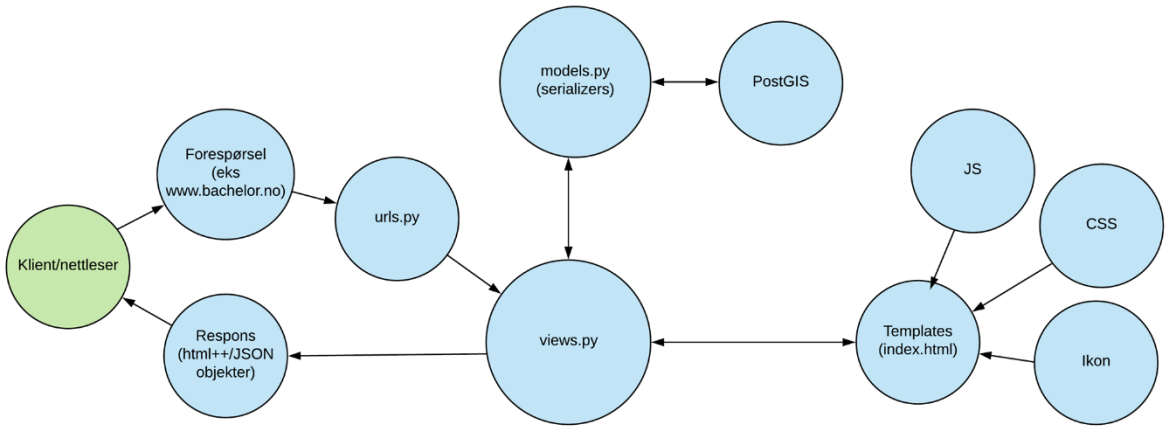
3.1 Valg av programmeringsspråk og webrammeverk

Gruppen hadde i utgangspunktet lite programmeringserfaring, men ønsket å lære dette. En av gruppens medlemmer hadde noe erfaring med programmeringsspråket Python, og en annen hadde laget en nettside med HTML og CSS. Da vi bestemte oss for å lage en applikasjon, var det derfor naturlig å benytte noe av de teknologiene vi hadde noe kjennskap til. HTML, CSS og JavaScript er grunnlaget for nesten all programmering i nettleseren, så det ble et naturlig valg. På serversiden finnes det flere alternativer deriblant PHP, Python, Java og C# for å nevne noen. Python er også et praktisk språk å kjenne til i GIS-sammenheng da det er støttet som scriptingspråk blant annet i QGIS (QGIS, 2018, s. 9) og ArcGIS (ESRI, 2017).

Av tilgjengelige webrammeverk som baserer seg på Python, sto valget mellom Flask og Django. Begge disse er populære, og har derfor god dokumentasjon og mange eksempler på nettet. Flask er et lettvekts rammeverk som er raskt å få en applikasjon på nettet, men med lite funksjonalitet «ut av boksen» og må derfor kombineres med andre moduler/bibliotek for å få dekkende funksjonalitet (Flask, 2018 s. 3). Django på sin side er et mye større og heldekkende rammeverk, blant annet med sin integrerte ORM for å kommunisere med databaser (Django, 2018, s. 7). Vi kom til at Flask sin modularitet nok kan være å foretrekke for programmerere som vil velge sine egne verktøy. For oss som hadde begrenset erfaring med programmering, var det godt med et rammeverk som dekket de fleste behovene. Et annet stort pluss for Django var for oss grenen GeoDjango som inneholder mye GIS funksjonalitet (Django, 2018, s. 770-771).

Etter at valget ble tatt om å bruke GeoDjango, ble så spørsmålet hvilken database som skulle brukes. GeoDjango har integrert støtte for databasene PostGIS, Oracle, MySQL og SpatiaLite (Django, 2018, s. 783). Vi valgte å bruke PostgGIS som vår geodatabase da vi hadde kjennskap til denne fra før igjennom studiets tidligere GIS-faglige emner. PostGIS (PostgreSQL) har åpen kildekode som vi også anså som positivt (PostgreSQL, 2018, s. lxx).

3.2 Applikasjonstjener og funksjonalitet i klient



Figur 6: Oppsett applikasjonstjener.

Figur 6 over viser dataflyten fra klienten sender en forespørsel, til den mottar en respons. Nettleseren sender en forespørsel til webtjeneren. Denne blir tolket i urls.py (vedlegg 10.1.8) filen, som henviser til hvilken visning som skal sendes tilbake i views.py (vedlegg 10.1.9). Views.py henter HTML-templaten som skal vises og/eller samler sammen informasjon fra databasen via «serializers» som oversetter innholdet fra databasen til JSON-objekter. I HTML-dokumentet kan det skrives JavaScript og CSS direkte, men har i vårt tilfelle blitt sortert til egne filer. JavaScript (JS) og CSS refererer til eksterne biblioteker slik som Leaflet og jQuery, men også egendefinerte filer slik som load_map.js (vedlegg 10.1.3) og main.css (vedlegg 10.1.5).

I valget av visningsverktøy til fremstilling av WMS-kart i nettsiden, ble JavaScriptbiblioteket Leaflet valgt. Dette biblioteket er godt integrert i Django sitt økosystem gjennom

programvareutvidelsen «Django Leaflet» (Makina Corpus, 2017). Koblingen mot kartverkets WMS-tjeneste blir gjort i settings.py-filen slik det er vist i figur 6. Vi ønsket i utgangspunktet å benytte et WMS-kart fra Statens Kartverk, fortrinnsvis i Euref89 UTM 32 eller 33, som er blant de offisielle kartprojeksjonene i Norge (Statens Kartverk, 2017). LeafletJS støtter i utgangpunktet ikke disse kartprojeksjonene, men de kan tilpasses ved tilleggsbiblioteket Proj4Leaflet (Agafonkin og Leaflet, 2017). Kartverket deler WMS-kart over fire ulike protokoller (Statens Kartverk, 2018). Av disse testet vi WMTS, og Google Maps API. I WMTS var det mulighet for å velge Euref89-projeksjonene, men disse hadde betraktelig dårligere opplastingsytelse. Vi valgte med dette den minst kompliserte løsningen, som var å bruke Google Maps API, med koordinater i EPSG: 4326, WGS 84.

Panoreringsmuligheten ble begrenset til Norge. Denne begrensningen ble valgt et godt stykke utenfor landegrensene. Dette var for at den ikke skulle føles restriktiv, men samtidig opprettholde sin funksjon. Begrensningen følger koordinatene vist i figur 7 på linje 155. Koordinatene er oppgitt på formen maksverdi: (vest, sør, øst, nord).

```
143
144  # LEAFLET
145
146  LEAFLET_CONFIG = {
147      'TILES': [('kartverket',
148                 'http://opencache.statkart.no/gatekeeper/gk/gk.open_gmaps?layers=norges_grunnkart_graatone&zoom={z}&x={x}&y={y}',
149                 {'attribution':'<a href="http://www.kartverket.no/">Kartverket</a>'})
150                 ],
151      'RESET_VIEW': False,
152      'DEFAULT_CENTER': (60.5, 8.5),
153      'DEFAULT_ZOOM': 7,
154      'MIN_ZOOM': 5,
155      'SPATIAL_EXTENT': (-36, 40, 71 , 77),
156      'SCALE': False,
157      }
158
```

Figur 7: Konfigurasjon av funksjon som henter WMS-kart.

Når man besøker webapplikasjonen blir index.html-filen lastet opp til nettleseren (vedlegg 10.1.2). I denne er det en Django-Leaflet tag (linje 29) som henter WMS-kartet fra Kartverkets sider. Når kartet er lastet blir skriptet load_map.js kjørt (vedlegg 10.1.3). Dette skriptet henter alle kulturminneobjektene fra PostGIS-databasen i JSON-format og plotter de i kartet med tilhørende ikon. Load_map.js skriptet sørger for å sette opp AJAX-funksjonalitet med et utvidelsesbibliotek av LeafletJS. Dette gjør at webapplikasjonen kan laste innhold fra serveren etter at applikasjonen er lastet inn i nettleseren (Kristoffersen, 2016, s. 338-339). I applikasjonen vår er dette data knyttet til spesifikke kulturminner som ikke trengs å lastes inn før aktuelle kulturminne er klikket på. Disse dataene blir delt på en URL på formen «domenenavn/data/media/km_id», der "km_id" er identifikatoren til det aktuelle kulturminnet. Dataene blir delt på JSON-format, slik figur 8 viser. Ved klikk på et kulturminne, vil bildegalleri, og en tekstboks med informasjon om kulturminne bli lastet inn. Deretter lastes bildefremviseren inn i bakgrunnen. Denne kommer først til syne når det klikkes i bildegalleriet. I bildegalleriet ligger representasjoner av bilder og 3D-modeller som får tildelt forskjellige klikkfunksjoner.

JavaScript knyttet til 3D-fremviseren er separert til en egen JavaScriptfil kalt load_model.js (vedlegg 10.1.4). Modellene som lastes inn er begrenset til filformatet .OBJ, og kun en medfølgende teksturfil.

```
[
    {
        "kulturminne": 154,
        "bilde": "http://127.0.0.1:8000/media/Modell/Kaupanger%20stavkirke/Screenshot_50.png",
        "modell_obj": "http://127.0.0.1:8000/media/Modell/Kaupanger%20stavkirke/Kaup_07.obj",
        "modell_mtl": "http://127.0.0.1:8000/media/Modell/Kaupanger%20stavkirke/Kaup_07.mtl",
        "modell_tekstur": "http://127.0.0.1:8000/media/Modell/Kaupanger%20stavkirke/Kaup7_color.png",
        "type": "modeller"
    },
    {
        "kulturminne": 154,
        "bilde": "http://127.0.0.1:8000/media/Bilde/Stavkirke/Kaupanger%20stavkirke/Kaupanger_stavkirke_11.jpg",
        "info": "Dette er et felt for å fylle inn informasjon om aktuelle bilde",
        "type": "bilder"
    },
```

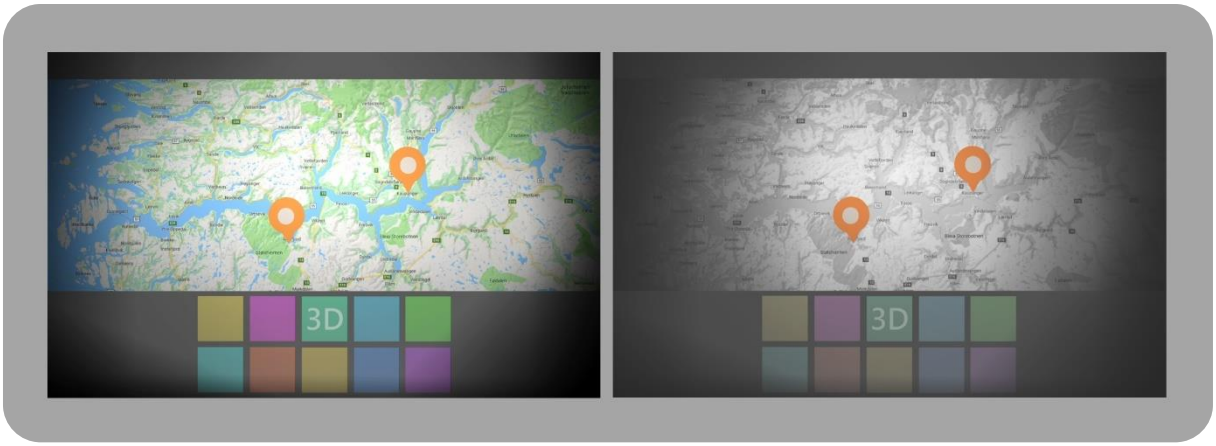

Figur 8: Serialisert data på JSON-format.

3.3 Design av brukergrensesnitt

For å oppnå mest mulig kontroll på det visuelle, valgte vi å unngå bruk av ferdige "templates". Dette ble gjort slik at utformingen kunne defineres fra grunnen ved hjelp av HTML og CSS. Skissering av ulike løsninger for utforming og layout ved å skrive HTML og CSS kan imidlertid være svært tidkrevende. Skisseutkast ble derfor utarbeidet i GIMP 2.8.14, et bilderedigeringsprogram med åpen kildekode som dekker det aller meste av funksjonaliteten som er å finne i tilsvarende programmer fra proprietære aktører som Adobe Photoshop. Rasterelementer ble ved hjelp av dette organisert i ulike lag som kunne flyttes og justeres hver for seg. Kartlaget besto av et skjermutklipp av Sognefjorden (Google maps, u.å.). På denne måten ble det relativt raskt å prøve ut forskjellige visuelle grep.

Som ramme for designet falt valget på et "single page layout" konsept, hvor elementer som skal vises og skjules legges delvis over det som allerede er der, slik at man befinner seg på samme sted selv om man navigerer innenfor denne rammen. En annen ramme som ble satt var å ha kartet i fokus. Ut over dette ble det bestemt å ta utgangspunkt i et mørkt brukergrensesnitt hvor det hele rammes inn av en oval skyggegradering ut mot kantene for å lukke fokuset inn mot sentrum og gi et inntrykk av belysning.

Det ble prioritert å fremheve kartmarkørene ved å anvende et bakgrunnskart i gråtoner, samt å tone ned fargemetningen i bildegalleriet knyttet til kartelementene, som vist i figur 9. Dette gir også et mer ryddig inntrykk.



Figur 9: Utprøving av fargemetning på kart og tilknyttede bildeelementer.

For å tilføre designet elementer assosiert med noe norsk og særegent, ble det prøvd ut hvordan karakteristikk forbundet med stavkirker kunne brukes. Mange av disse er utsmykket med intrikate nettverk av svært egenartet treskurd. Det ble derfor besluttet å la elementer fra dette prege designet. Med utgangspunkt i en illustrasjon av en portal fra Lomen stavkirke (Øverland, 1886) ble det ved hjelp av GIMP hentet ut deler fra denne som ble eksportert til et gjennomsiktig png-format som vist i figur 10.

Figur 10: Arbeidsflyt i GIMP.

I det endelige utkastet vist i figur 11, ble fokuset på kartet styrket ved å fjerne det heldekkende mørke feltet som dannet skillet mellom kartet og bildegalleriet. Også en økning av gjennomsiktigheten på tittelfeltet i toppen fremhevet kartet mer. Disse grepene styrker også helhetsinntrykket da layouten blir mindre oppdelt og mer sømløs og flytende.

Figur 11: Endelig skisseutkast av brukergrensesnittet

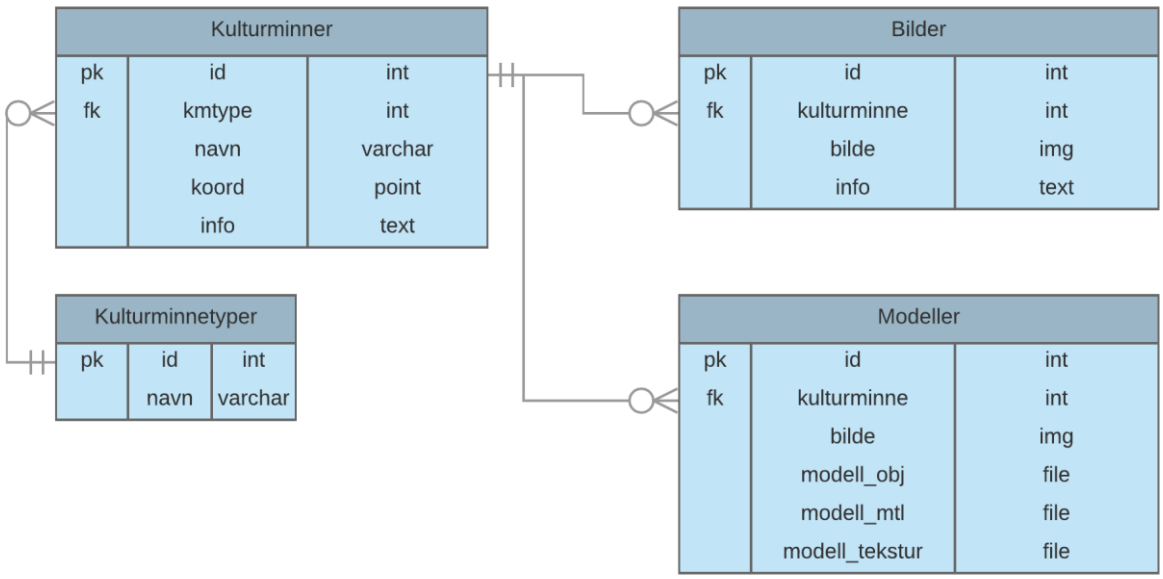
Et sentralt element i kartet er kartikonene som angir kulturminnetypen. Det ble vektlagt å opprettholde et ryddig kart ved å la kartikonene ha samme farge og utforming. Skillet mellom de ble ved dette nødt til å bero seg på utformingen av de ulike motivene satt til å representere den aktuelle kulturminnetypen som vist i figur 12.

Ikonene er utformet ved hjelp av InkScape 0.92, et vektorbasert tegneprogram med åpen kildekode, som dekker det aller meste av funksjonaliteten som er å finne i tilsvarende programmer fra proprietære aktører som Adobe Illustrator. Ikonene ble satt sammen lagvis som vist i figur 12.

Figur 12: Konstruksjon av kartikoner i InkScape.

3.4 Valg av databasestruktur

ER-diagrammet under i figur 13 viser databasestrukturen i webapplikasjonen. Kardinalitet-notasjonen viser eksempelvis at hvert kulturminne kan ha 0-eller mange modeller. Mens hver modell må ha ett og bare ett kulturminne. Relasjonen "Bilder" knyttes til kulturminner på samme måte som relasjonen "Modeller". På samme måte har hvert kulturminne en, og bare en, kulturminnetype. Kolonnene til venstre i hver relasjon viser primærnøkkelfelt (pk) og fremmednøkkelfelt (fk). Kolonnene i midten viser benevnelsen på hver egenskap i relasjonen, og kolonnen til høyre viser tilhørende datatype.



Figur 13: ER-diagram databasestruktur.

Kulturminneobjekt kan håndteres på flere måter i databasen. I teoridelen ble det redegjort for 3 ulike metoder (se figur 2). Disse løsningene ble vurdert.

Med fremgangsmåten «kun subtype» (ref. Metode 1, figur 2) finnes det ingen felles koblingspunkt for alle kulturminner. Et av målene med webapplikasjonen var å ha muligheten til å lagre flere bilder og 3D-modeller til hvert kulturminne. Det var derfor behov for et felles koblingspunkt og denne metoden ble derfor ikke aktuell.

Django har støtte for metoden «subtype og supertype» (ref. Metode 2, figur 2) igjennom sin ORM kalt Multi-Table Inheritance (Django, 2018, s. 98-100). Vi valgte å kun lagre egenskaper som er felles for alle «Kulturminnetypene» i databasen. Webapplikasjonens funksjoner gjorde det overflødig å lagre flere egenskaper. Det ble derfor ikke behov for å opprette «subtyper» for hvert enkelt «Kulturminnetype».

Valget falt derfor på Metode 3 (figur 2) hvor alle kulturminnene blir representert i en tabell. «Kulturminner» og «Kulturminnetyper» knyttes sammen gjennom koblingsnøkkelen til feltet «id» i «kmtype», slik tabell 35 under viser.

ID (INTEGER)	NAVN (VARCHAR)
1	'Stavkirke'
2	'Gravhaug'
3	'Rune'

Tabell 1: Innlastet data "kulturminnetype".

3.5 Koblingen mellom Django og PostgreSQL

Django utnytter et integrert ORM (Object Relational Mapper) som er laget for å håndtere all interaksjon med databasen. Dette inkluderer spørringer, opprettelse og endringer av tabeller. Noe som gir god fleksibilitet i valg av databasesystem, men medfører også visse begrensninger ettersom spesielle egenskaper i de ulike databasene i mindre grad er støttet. Databasetilkoblingen blir gjort i prosjektet sin settings.py-fil som vist i figur 14.

```
DATABASES = {
    'default': {
        'ENGINE': 'django.contrib.gis.db.backends.postgis',
        'NAME': 'KulturminneDB',
        'USER': 'postgres',
        'PASSWORD': 'postgres',
        'HOST': 'localhost',
        'PORT': '5432',


    }
}
```

Figur 14: Utsnitt av settings.py. Viser databasetilkoblingen i Django.

Databasestrukturen har blitt kodet i models.py (vedlegg 10.1.6). Hver modell har blitt kodet som en subklasse i Python. Denne arver egenskaper fra superklassen django.db.models.Model (Django, 2018 s. 84). Dette er vist i figur 15.

```
class Kulturminner(models.Model):
    navn = models.CharField(max_length=50, unique=True)
    koord = geo_models.PointField(blank=False)
    info = models.TextField(blank=True)
    kmtype = models.ForeignKey(KulturminneTyper, db_column='kmtype', on_delete=models.PROTECT)

    def __str__(self):
        return self.navn

    class Meta():
        verbose_name = 'Kulturminne'
        verbose_name_plural = 'Kulturminner'
        ordering = ['navn']
```

Figur 15: Utsnitt av models.py. Viser koding av databasestrukturen.

Oversettelsen mellom Django og PostgreSQL gjøres med ORM som vist i figur 16. Dersom det ikke er spesifisert en primærnøkkel i modellen, vil Django automatisk opprette et integer felt «id» og bruke dette som primærnøkkel. Tabellene får følgende navngivning: applikasjon_klassenavn. Dersom annet ikke er spesifisert vil også en fremmednøkkelfelt bli navngitt: egenskapsnavn_id.

ER-skjemaet i figur 13 viser koblinger og datatyper slik de fremstår i GeoDjango. I Django kan det modelleres med datatyper som integer, varchar, text, point, img og file. De tre sistnevnte er ikke direkte oversettbare til en PostgreSQL-database (PostgreSQL, 2018, s.117-118).

```
--
CREATE TABLE "km_app_kulturminnetyper" ("id" serial NOT NULL PRIMARY KEY, "navn" varchar(50) NOT NULL UNIQUE);
--
--
-- Add field kmtype to kulturminner
--
ALTER TABLE "km_app_kulturminner" ADD COLUMN "kmtype" integer NOT NULL;
--
ALTER TABLE "km_app_kulturminner" ADD CONSTRAINT "km_app_kulturminner_kmtype_88b88ec7_fk_km_app_ku" FOREIGN KEY ("kmtype") REFERENCES "
km_app_kulturminnetyper" ("id") DEFERRABLE INITIALLY DEFERRED;
```

Figur 16: Oversettelse fra Django til SQL v.h.a ORM.

«Point» oversettes til en «geometry» datatype som er et spesialfelt for lagring av geografisk informasjon i PostGIS (PostGIS, 2018, s. 33). «Img»- og «file»-feltene blir opprettet som varchar(100) i databasen. Dette feltet inneholder referanse til filen, slik den blir lagret i filsystemet. Denne tilnærmingen er foretrukket i Django (Django, 2018, s. 1103). En annen tilnærming er å laste filen opp som en BLOB (Binary Large Object) (Django, 2018, s. 1103).

Figur 17 viser oppsettet for kulturminnetabellen i PGAdmin. I koordinatfeltet feltet blir dataene lagret som Well Known Binary (PostGIS, 2018, s. 36). Innholdet i koordinatfeltet kan oversettes til Well Known Text. Resultatet blir 'POINT (7.23346750950286 61.1842046547582)'. Koordinatene er lagret i referansesystemet EPSG: 4326, WGS84.

	id integer	navn character varying (50)	koord geometry	info text	kmtype integer
1	154	Kaupanger stavkirke	0101000020E610000032F9571B12EF1C408DFAA30494974E40	Lore...	1

Figur 17: Koordinatfeltet i PGAdmin.

Bilder og andre store filer lagres som forklart i applikasjonens filsystem i en mappe kalt «media». Dersom ikke annet er spesifisert, vil Django laste opp navnet på bilder og filer slik de er og legge de direkte i rotmappen. Vi valgte å lage en funksjon som navngir bildene og sorterer de i mapper basert på hvilket kulturminne de hører til (figur 18). Dette medfører at mediamappen blir mer logisk strukturert.

```
# FILE DIRECTORY

def image_directory_path(instance, filename):
    extension = filename.split('.')[-1]
    fk_classname = str(instance.kulturminne.kmtype.navn)
    fk_objname = str(instance.kulturminne)
    mediatype = 'Bilde'
    directory = os.path.join(settings.MEDIA_ROOT, mediatype, fk_classname, fk_objname)

    if os.path.exists(directory):
        antall_bilder = len(os.listdir(directory))
    else:
        antall_bilder = 0

    return '{0}/{1}/{2}/{2}_{3}.{4}'.format( mediatype, fk_classname,
                                            fk_objname,
                                            antall_bilder,
                                            extension)
```

Figur 18 Funksjon som navngir bilder og sorterer de basert på tilhørende kulturminne.

Bildene, som blir lagret for Kaupanger stavkirke, blir sortert i mapper med strukturen «mediatype/kulturminnetype/kulturminnenavn/kulturminnenavn_index.filtype» som vist i figur 19 under.

	id integer	bilde character varying (100)	info text	kulturminne_id integer
1	1	Bilde/Stavkirke/Kaupanger stavkirke/Kaupanger_stavkirke_0.jpg	Bilde...	154
2	2	Bilde/Stavkirke/Kaupanger stavkirke/Kaupanger_stavkirke_1.jpg	Bilde...	154
3	3	Bilde/Stavkirke/Kaupanger stavkirke/Kaupanger_stavkirke_2.jpg	Innv...	154

Figur 19: Oppsett i bildetabellen.

3.6 Innsetting av data i databasen

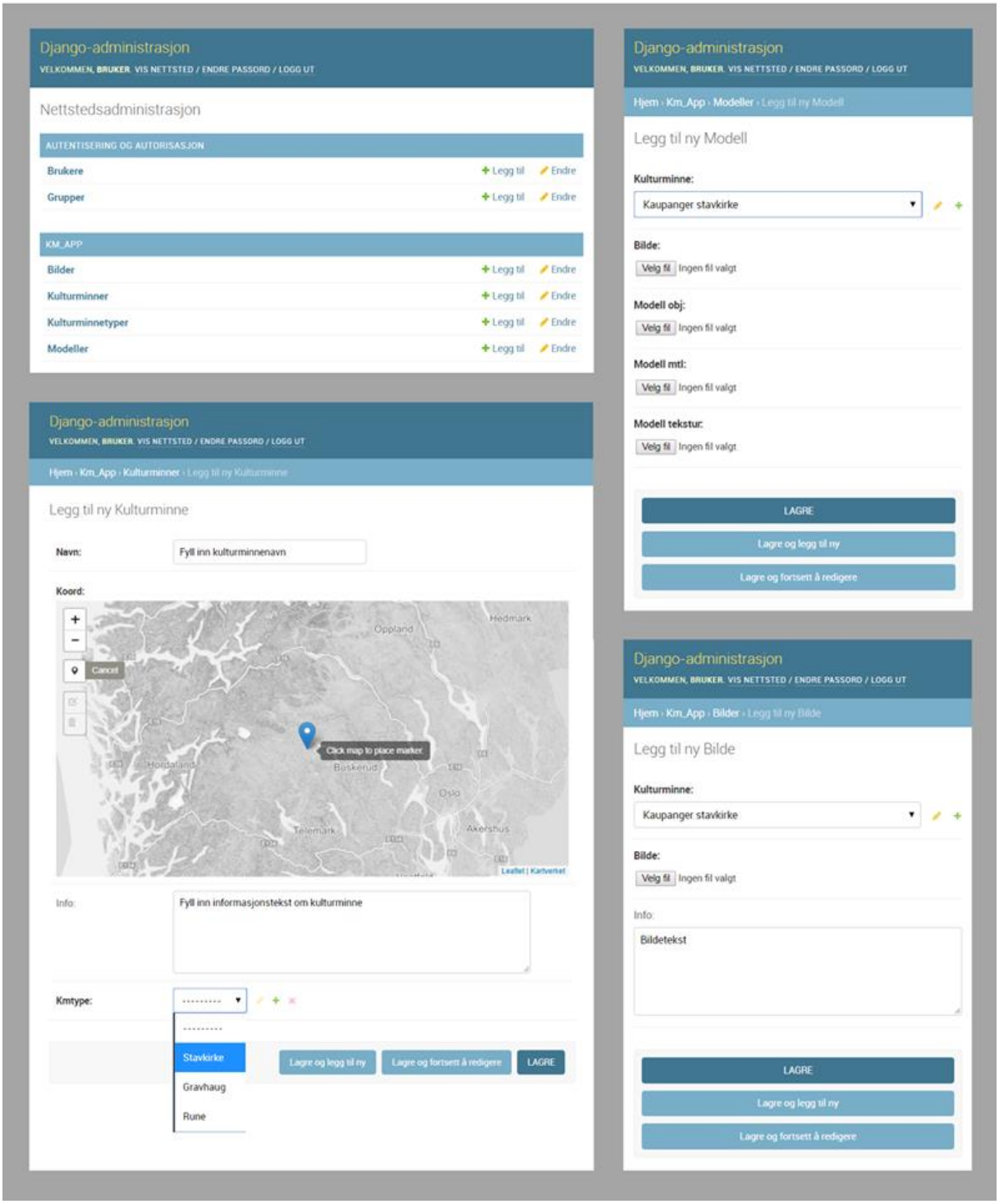
Databasestrukturen (se ER-skjema figur 13) krever at «Kulturminnetypene» legges inn før data kan legges inn i andre tabeller. Radene i tabellen «Kulturminnetype» er forhåndslagret i filen «kmtyper.json».

For å fylle databasen lastet vi ned shapefiler med punktdata fra registrerte kulturminner i Norge. Disse ble hentet fra Askeladden, som er Riksantikvarens database over kulturminner (Riksantikvaren, u.å.a). Shapefilene inneholdt informasjon som databasen vår ikke har blitt laget for å kunne ta imot. Det var uoverensstemmelse mellom koordinatsystem i nedlastede filer fra Askeladden og WMS-kartet til webapplikasjonen. Shapefilene ble derfor lagt inn i QGIS. Der transformerte vi koordinatene og fjernet unødvendig data. Tekstfeltene ble fylt med stedfortredertekst av typen Lorem ipsum. Kun for Kaupanger stavkirke har det blitt produsert informativ tekst. Shapefilene ble så lastet inn i PostgreSQL-databasen ved hjelp av et loadingscript vist i figur 20.

load_shapefiles.py

```python
from os import path
from django.contrib.gis.utils import LayerMapping
from km_app.models import Kulturminner

shape2db = {
    'navn': 'NAVN',
    'info': 'INFO',
    'koord': 'POINT',
    'kmtype': {'id': 'TYPE'},
    }

datafolder = path.abspath(path.join(path.dirname(__file__),'data/'))
shapefiles = ['Stavkirker/Stavkirker.shp', 'Gravhauger/Gravhauger.shp', 'Runer/Runer.shp']

def run(verbose=True):

    for i in range(len(shapefiles)):
        km_path = path.join(datafolder, shapefiles[i])
        stav2db = LayerMapping(Kulturminner, km_path, shape2db, encoding='utf-8', transform=False,)
        stav2db.save(strict=True,verbose=verbose)
```

Figur 20: Load_shapefiles.py.



Figur 21: Administrasjonsside Django.

Django har en integrert administrasjonsside som enkelt kan settes til opp å behandle data fra webapplikasjonen (Django, 2018, s. 698). Denne ble i satt opp slik at man får tilgang til å legge til, endre og slette kulturminner, bilder og 3D-modeller. For oppsett vises det til admin.py (vedlegg 10.1.1). Figur 21 over viser et utsnitt fra administrasjonssiden, slik den ble brukt til å legge til 3D-modeller og bilder.

4 Datainnsamling

Hensikten med dokumentasjonen av Kaupanger stavkirke var blant annet å innsamle data til produksjon og fremvisning av en 3D-modell på webapplikasjonen. I tillegg var det ønskelig å sammenligne nøyaktigheten på punktskyen fra laserskanneren mot punktskyene fra bildene tatt med digitalkamera og drone.

4.1 Etablering av grunnlagsnett og innmåling av blinker

For å kunne måle inn blinker ble det først opprettet et grunnlagsnett med fastmerker. Grunnlagsnettet ble opprettet i NTM sone 7. Av utstyr ble det brukt Leica TS12 totalstasjon og Leica rundprismer. Leica Viva GS14 GNSS ble også benyttet.

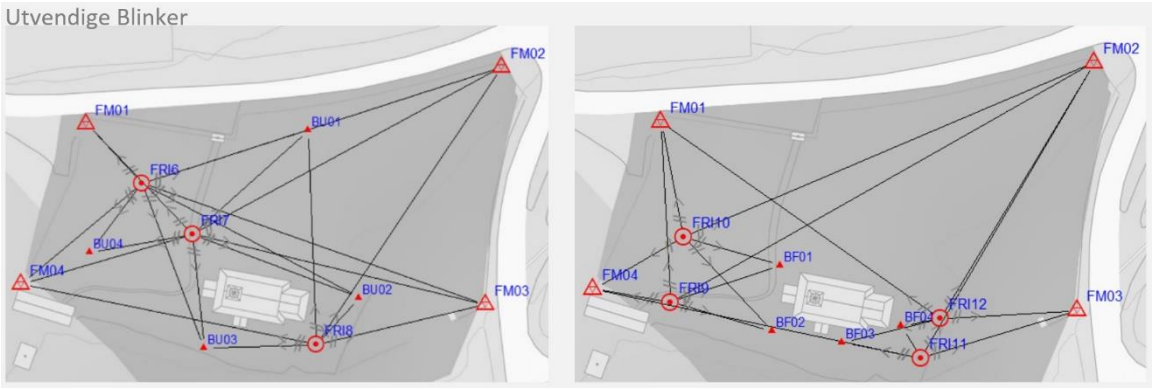
Figur 22: prisme over fastmerke

Et grunnlagsnett bestående av fire fastmerkepunkt(FM01-04) ble etablert i utkanten av måleområdet. Fastmerkene ble etablert på toppen av steinmuren rundt kirkegården. En liten fordypning ble frest ut slik at spissen til prismet sto støtt, uten at fastmerket ble synlig for ettertiden. Ulike plasseringer for fastmerkene ble vurdert. Blant annet på bakken inne i kirkegården og i trestubber etter hogst på andre siden av muren. Plassering på muren var gunstig med tanke på siktlinjer og stabilitet.

For at fastmerkene skulle få en absolutt plassering ble de først målt inn med sanntids-GNSS (G01-G04). Koordinatene fra GNSS-målingene ble videre brukt til å etablere en frioppstilling med totalstasjonen. Fra frioppstillingen målte vi inn blinkene på nytt. Koordinatene på fastmerkeblinkene fikk da en bedre innbyrdes nøyaktighet.

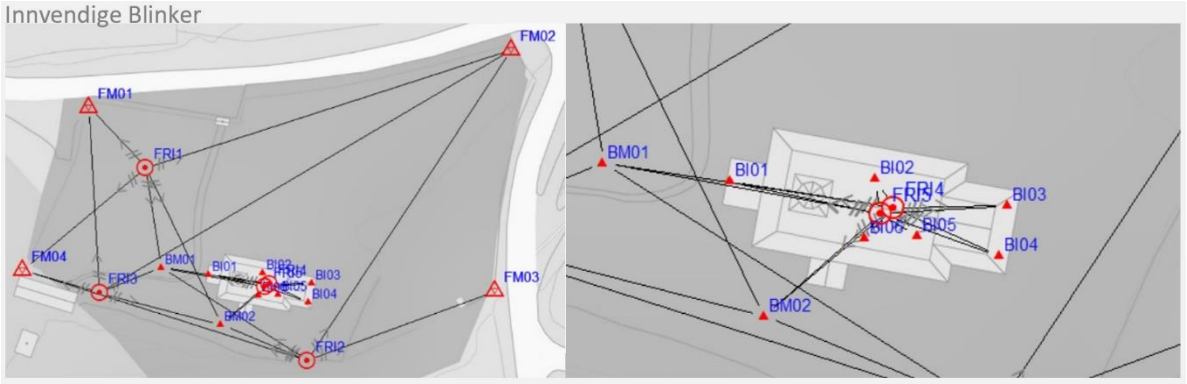
I datafangsten ble det benyttet tre sett blinker. Ett for utvendig laserskanning, ett for innvendig laserskanning og ett for utvendig fotogrammetri. Blinkene ble målt inn med totalstasjonen fra minst to frioppstillinger med best mulig spredning.

De utvendige blinkene (BU01-04) til laserskanningen ble målt inn slik at de holdt seg innenfor skannerens virkeområde på 120 meter i aktuelle oppstillingspunkt (leica_manual, s. 131). Blinkene ble montert på oppstilte stativ. Skanneren kunne således plasseres på innsiden av blinkene slik at den fikk fri sikt til kirkens fasade. Dette bedret geometrien fra skannerens oppstillinger til blinkene. Blinkene til fotogrammetrien (BF01-04) ble plassert nærmere fasaden på kirken for at blinkene skulle bli synlig på flest mulig bilder.



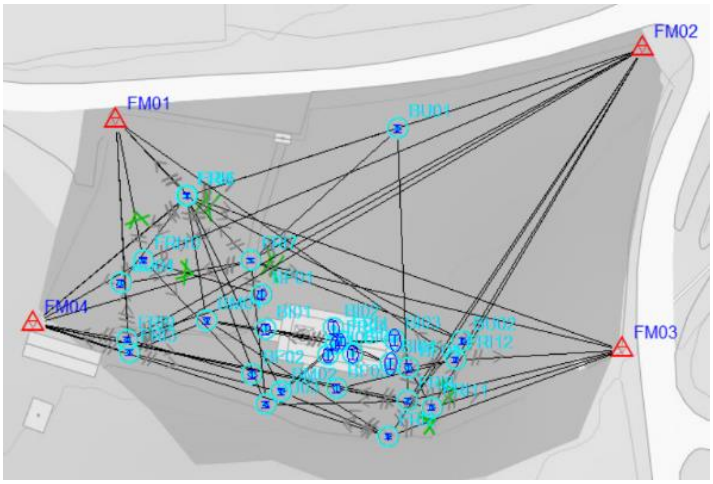
Figur 23: Utvendige blinker.

Ved innmålingen av de innvendige blinkene ble det målt inn to midlertidige blinker på utsiden (BM01-02). Sistnevnte blinker var synlige fra innsiden av kirken. Disse ble brukt til å etablere to frioppstillinger på innsiden av kirken. Deretter ble disse brukt til å måle inn de innvendige blinkene (BI01-06).



Figur 24: Innvendige blinker.

Beregning av koordinatene til blinkene for hele nettet ble gjort i en samlet 3D-utjevning. Alle blinkene var målt inn fra minst to frioppstillinger. På bakgrunn av grovfeilsøkene valgte vi å utelukke 6 målinger. Den globale redundansen for nettet ble deretter beregnet til 0.55 (vedlegg 10.2.1).



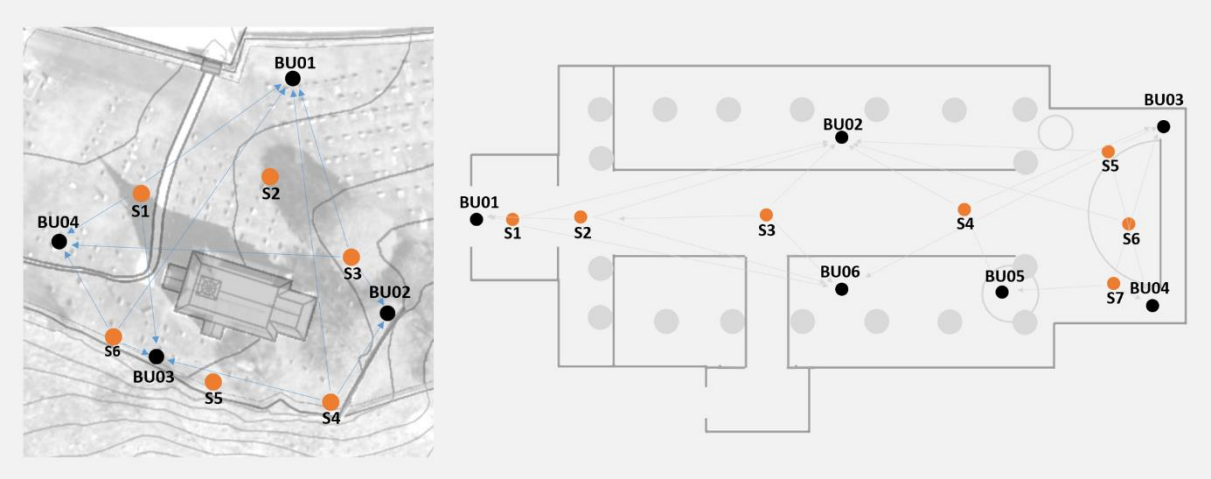
Figur 25: Utjevning i Gemini Oppmåling.

4.2 Terrestrisk laserskanning

Til laserskanningen ble det brukt Leica P20 instrument med tilhørende blinker. Etter at de utvendige blinkene var målt inn ble kirken skannet utvendig. I hver av de 6 skannerposisjonene ble følgende prosedyre gjennomført:

- Grovskann i hele skannerens arbeidsfelt (360˚x 270˚).
- Blinker ble målt inn.
- Finskann med skannerens arbeidsområde begrenset til kirken.
- Kontroll av blinker.

Innvendig ble det gjort 7 oppstillinger. Blinkene ble plassert slik at de omsluttet skannerposisjonenene for å oppnå god geometri i horisontalplanet. For å oppnå vertikal spredning ble blink BI05 ble plassert på toppen av prekestolen, mens BI03 og 04 ble plassert på bakkeplan. Samme prosedyre som over ble fulgt med unntak av at finskanning ble gjort med hele skannerens arbeidsfelt. Måleopplegget er illustrert i figur 26. Svart punkt er blink. Oransje punkt er skannerposisjon.



Figur 26: Utvendig og innvendig måleopplegg.

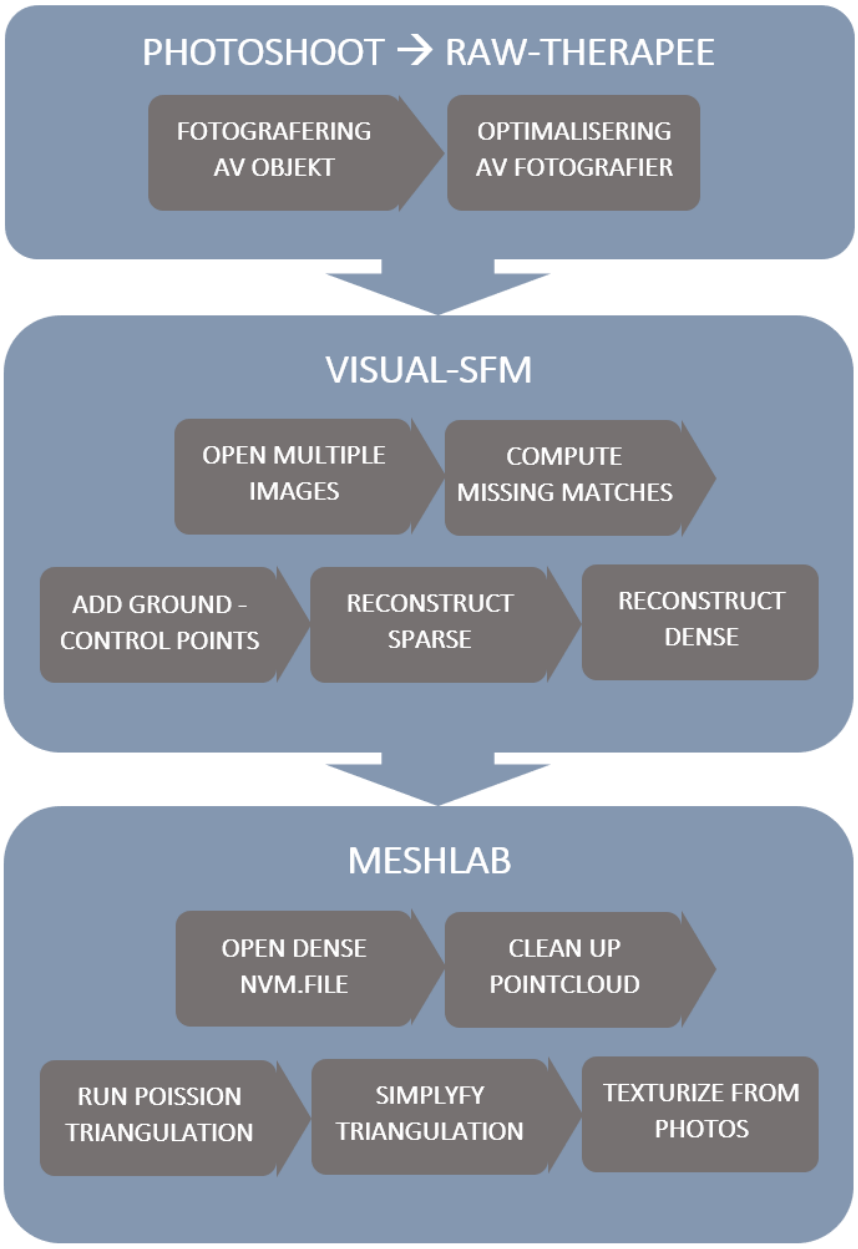
Laserskanndataene ble bearbeidet i Leica Cyclone 9.1. Blinkregistreringer ble gjort med og uten georeferering. Utvendig og innvendig punktsky med blinkregistreringer ble eksportert.

For å hente frem detaljer i skannerens undereksponerte bilder av kirkens innside ble det utført et histogramstrekk ved hjelp av RawTherapee 5.3. Dette er et program med åpen kildekode som dekker det aller meste av funksjonaliteten som er å finne i tilsvarende programmer fra proprietære aktører som Adobe Lightroom. Utsnitt av operasjonens resultat er vist i figur 27.

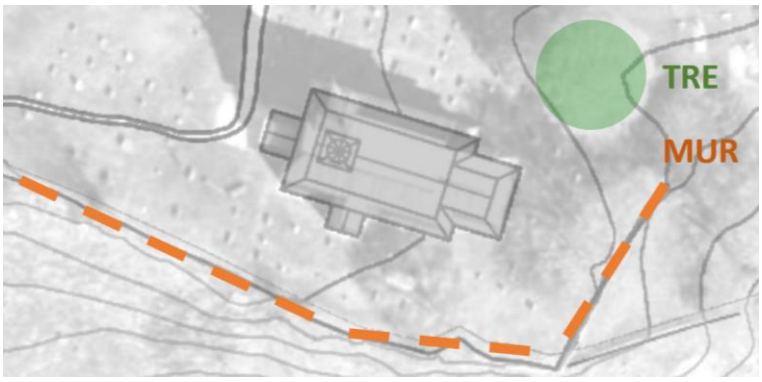
Figur 27: Resultat av histogramstrekk.

4.3 Terrestrisk fotogrammetri

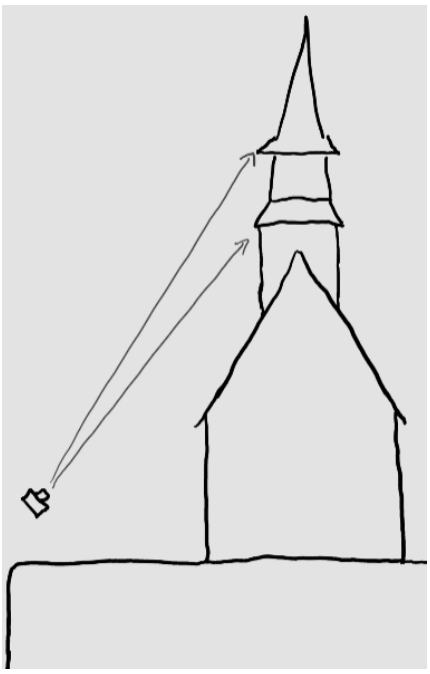
SFM-programvare har gjort det mulig å redusere utstyrsbehovet til et ordinært kamera. Vi har prøvd ut og vurdert resultater fra en arbeidsflyt basert på et speilrefleks digitalkamera og gratis programvare som illustrert i figur 28. Datainnsamlingen ble utført ved hjelp av kamerahuset Nikon D600, sammen med objektivet AF-S NIKKOR 14-24mm f/2.8G ED. Dette kamerahuset inneholder en fullformat bildesensor på 24 x 35,9mm. Oppløsningen er på 24,3 millioner effektive piksler, med et bildefelt på 4016 x 6016px.



Figur 28: Arbeidsflyt for utarbeidelse av 3D-modell vha. kamera og gratis programvare



Figur 29: De fysiske forholdene som formet rammen for datainnsamlingen



Figur 30: Illustrasjon av vinkelforhold.

Fotograferingen ble utført i overskyet vær med god spredning av diffust dagslys. 14-24 millimeteren ble gitt en fast brennvidde på 20mm for å minimere sannsynligheten for feilkalkulering av kalibreringsparametere under strålebuntutgjevningen. Blenderåpning ble satt til f/10 for å sikre tilstrekkelig dybdeskarphet, og ISO (International Organization for Standardization) til 100 for å minimere støy i bildene. Med en lukkertid på 1/125 sek. ble det unødvendig med stativ. Som sikkerhet mot tap av data i de lyseste delene av bildene, ble det på solsiden av kirken gjort en svak undereksponering på -0,7 EV (Exposure Value). Da disse innstilingene ble holdt konstant, ble undereksponeringen ganske stor på kirkens skyggeside. Bildene ble imidlertid tatt i RAW-format for å ta vare på mest mulig av bildenes potensiale i postprosessering, slik at detaljer i skyggesiden kunne hentes frem i etterkant.

Datainnsamlingen ble gjort innenfor rammen av de fysiske forholdene på stedet (figur 29). Platået som kirken hvilte på ble holdt på plass og avgrenset av en mur, langs hele kirkens sør- og østside. Dette, kombinert med et tre på kirkens østside, resulterte i at avstanden som kunne holdes til kirken under fotograferingen, her begrenset seg til 8-10m. Den relativt korte avstanden fikk konsekvenser for vinkelen mellom de øvre delene av kirken og kamera. Dette ga et spesielt dårlig utgangspunkt for datainnsamling av takpartienes skråplan (figur 30). Kirken ble innenfor dette fotografert ved forflytning rundt hele bygningen med håndholdt kamera, og med overlapp av bilder, godt innenfor 60%. Det ble i tillegg til dette tatt nærbilder av blinkene for å sikre godt utgangspunkt for georefereringen ved å styrke oppløsningen i disse områdene.

SIFT-algoritmen som skal finne og gjenkjenne strukturer i bildene vil ha et dårligere utgangspunkt dersom bildene preges av ensartede fargeflater (Wu, u.å.). I vårt tilfelle ville derfor undereksponerte skyggeflater utgjort en svakhet. Postprosessering ble derfor utført i RawTherapee 5.3. Ved hjelp av "batch processing", ble samme endring påført alle RAW-filene, som deretter ble eksportert til JPEG. Med verktøyene under eksponeringsmenyen ble

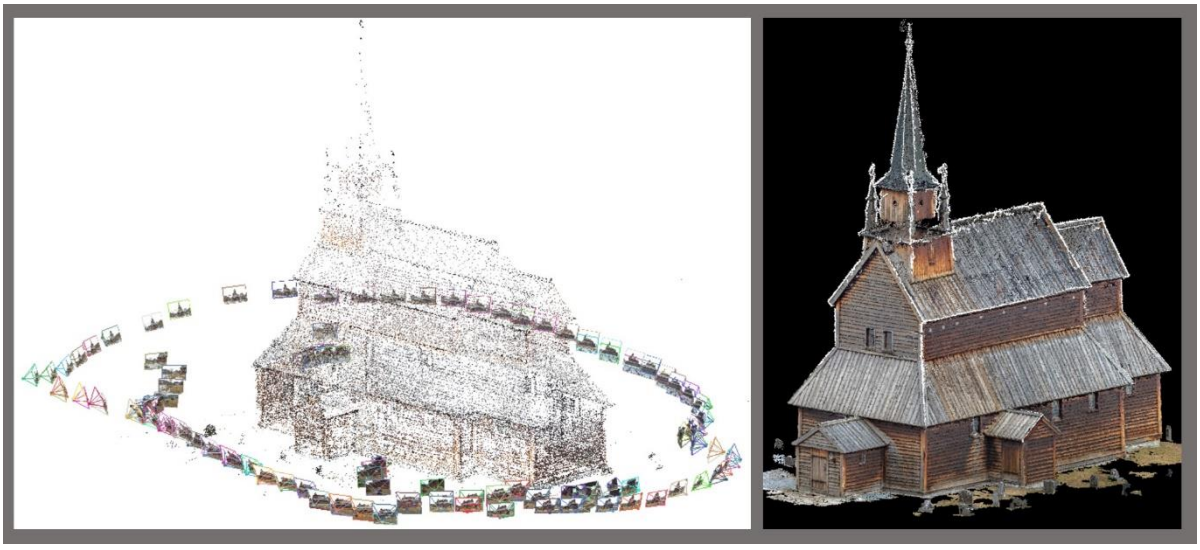
det konstruert et histogramstrekk som hentet frem undereksponerte skyggepartier. Endringen er illustrert i figur 31.

Figur 31: Viser før og etter histogramstrekk i RawTherapee.

Punktsky ble generert ved hjelp av VisualSFM 0.5.26, utviklet av Changchang Wu. Wu har ved dette stått for utviklingen av ett av svært få programvarealternativer med åpen kildekode som muliggjør rekonstruksjon i 3D basert på bilder (Wu, u.å.).

Fremgangsmåten som ble fulgt er utarbeidet av Jacob A. Morgan og Daniel J. Brogan ved "Department of Civil and Environmental Engineering" på "Colorado State University" (Morgan og Brogan, 2016).

Med utgangspunkt i professor David Lowe's SIFT-algoritme (UBC, u.å.), har Wu utviklet SiftGPU, hvor grafikkortet anvendes for å effektivt finne felles særegenheter i bildene som lastes opp. På bakgrunn av dette beregner Wu's "Multicore Bundle Adjustment", gjennom en iterativ prosess, kameraets relative posisjon og rotasjon i forhold til objektet for hver kameraoppstilling. I tillegg til dette korrigeres det for optisk forvrengning påført av kameraoptikken. Denne informasjonen brukes til å konstruere en "sparse cloud" som vist til venstre i figur 32.



Figur 32: Rekonstruksjon av "sparse cloud" (venstre) og "dense cloud"(høyre).

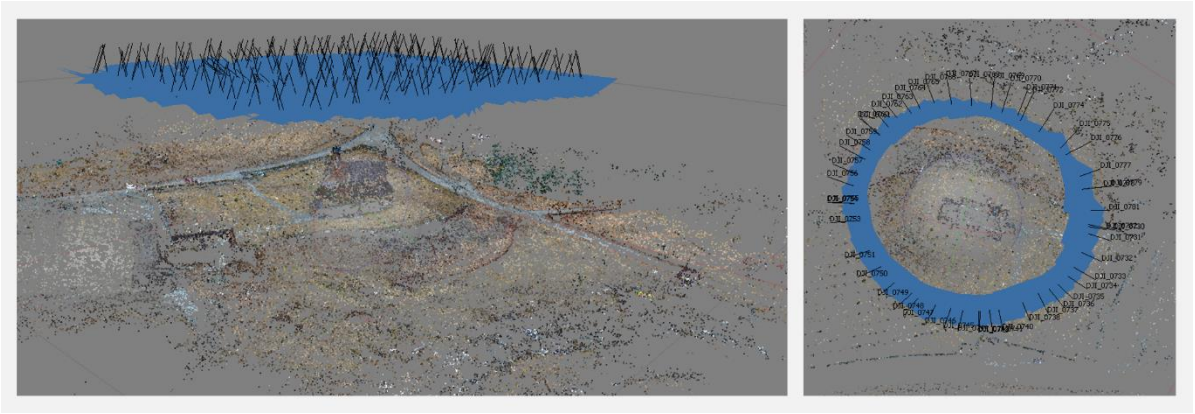
Georeferering ble utført i VisualSFM da programmet også har funksjonalitet for dette. Alle bildene ble gjennomgått og koordinatene på de innmålte blinkene ble lagt inn på bilder hvor blinkene var godt synlig og med god oppløsning.

For å kunne generere en tett punktsky har Wu integrert Patch-based Multiview Stero (PMVS) utviklet av Yasutaka Furuwaka and Jean Ponce, i tillegg til Clustering Views for Multi-view Stereo (CMVS) utviklet av Yasutaka Furuwaka. CMVS tar resultatet fra SFM som input og dekomponerer bildesettene inn i mindre og mer håndterbare sett med bilder som kan bli prosessert individuelt eller parallelt. PMVS bruker så bildesettene til å generere et tette punktsett (høgre i figur 32) hvor både x,y,z og normal blir beregnet for hvert punkt.

4.4 Dronebåren fotografering

Kaupanger Stavkirke inngikk også som studieobjekt i et annet bachelorprosjekt ved HVL våren 2018, og siden det var gjensidig interesse for å sammenligne resultater fra flere innsamlingsmetoder ble det besluttet å utveksle data. Bygg ingeniørstudentene Trond-Jøran Stokke og Ask Kvernberg Saastad undersøkte i sitt prosjekt datakvaliteten som kan oppnås ved bruk av "DJI Mavic Pro", en rimelig drone tilgjengelig fra Elkjøp. Denne har en bildesensor på 1/2.3" (6.17 · 4.55mm) med 12mp oppløsning.

Datainnsamlingen ble utført ved å fly over området i et rutenett 40m over bakken. Dette ble supplert med bilder tatt ved sirkulering rundt kirken også 40m over bakken. Georeferering ble gjort på bakgrunn av 6 kontrollpunkt som ble målt inn ved hjelp av GNSS. Punktsky ble generert i Agisoft Photoscan Professional v1.4.1.5925, med 75% reduksjon av bildestørrelse.



Figur 33: Generering av punktsky i Agisoft Photoscan.

5 Analyse og modellering

5.1 Sammenligning av punktskyer

Laserskyen ble brukt som grunnlag for sammenligningen. Ved å anvende denne som "fasit" ble det ved sammenligning estimert nøyaktighet på resultatet fra den terrestriske fotogrammetrien og dronen.

Sammenligningen av punktskyene ble utført i CloudCompare 2.9.1 med funksjonen Cloud/cloud distance (Cloudcompare.org, 2016). Avstand mellom 2 punktskyer beregnes for hvert punkt ved hjelp av nærmeste naboavstander (Cloudcompare.org, 2016). Laserskannskyen ble brukt som «Referance cloud» som avstandene regnes relativt til. Sammenligningsskyene var fotogrammetriskyen og droneskyen. For hvert punkt i sammenligningsskyen beregnes det en avstand relativt til referanseskyen (Cloudcompare.org, 2016).

Tabell 2 viser antall punkt til de ulike punktskyene i sammenligningene.

Punktsky	Antall punkt
Fotogrammetri	3 744 209
Drone_UTM	453 576
Laserskann_NTM	20 253 012
Laserskann_UTM	20 790 330

Tabell 2: Antall punkt brukt i sammenligningene.

5.2 Utarbeidelse av 3D-modell

Visning av 3D modell på web krever filstørrelse som nettleseren kan håndtere. Med tanke på innlastingstid i nettleseren er det en fordel at modellen har en relativt liten filstørrelse. En forholdsvis rask metode for å redusere filstørrelsen er å bruke punktskyen til å generere et nettverk av polygoner eller en såkalt "mesh". Denne må igjen forenkles kraftig for å oppnå lav filstørrelse. Selv om dette vil innebære forenkling og deformering av objektets opprinnelige form, vil en tekstur fra de tilknyttede bildene kunne projiseres over overflaten og gi en naturtro gjengivelse av objektet. Vi produserte derfor modeller i ulike størrelser og format som vi kunne teste.

For å kunne gi en vurdering av hvilken grad av kvalitetsmessige kompromiss denne prosessen medfører har vi foretatt en visuell sammenligning av en forenklet modell bygd på bakgrunn av dataene fra Nikon D600, mot en ukomprimert triangulering av dataene fra Leica P120, teksturert med bildene fra Nikon D600.

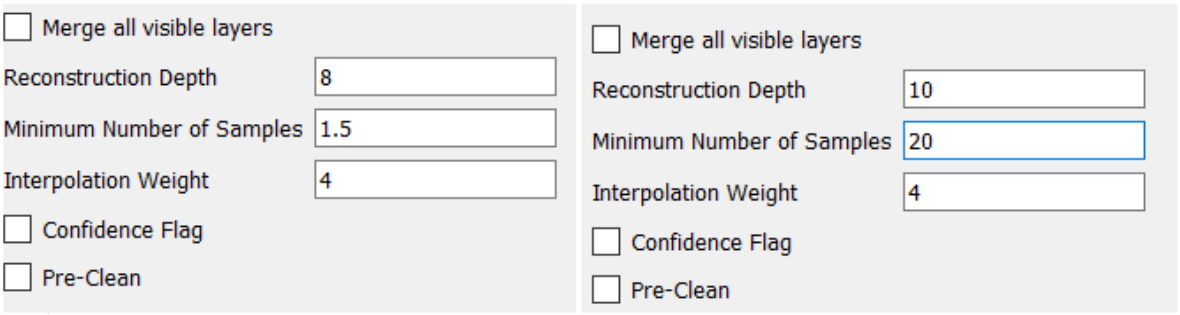
Modellene som ble bygget fra fotogrammetriskyen ble lagd i MeshLab versjon 2016.12, et åpent kildekodeprogram for prosessering og redigering av polygonnett. RealityCapture 1.0.3.4119 Demo RC ble brukt for å kombinere laserskannsky og bilder tatt med digitalkamera.

Genereringen av 3D-modellen tok utgangspunkt i den tette punktskyen fra VisualSFM på 3 744 209 punkt. Resterende støy ble fjernet manuelt ved hjelp av klippeverktøy i MeshLab. Resultatet av denne operasjonen er vist i figur 34.

Figur 34 tett punktsky importert i MeshLab

Funksjonen «Screened Poisson Surface Reconstruction» ble brukt til å triangulere punktskyen. Dette er en interpoleringsmetode som tar utgangspunkt i alle orienterte punkt og triangulerer en overflate. Hovedsakelig er det 2 parametere som gir stor innvirkning på overflaten. «Reconstruction depth» og «Minimum number of samples».

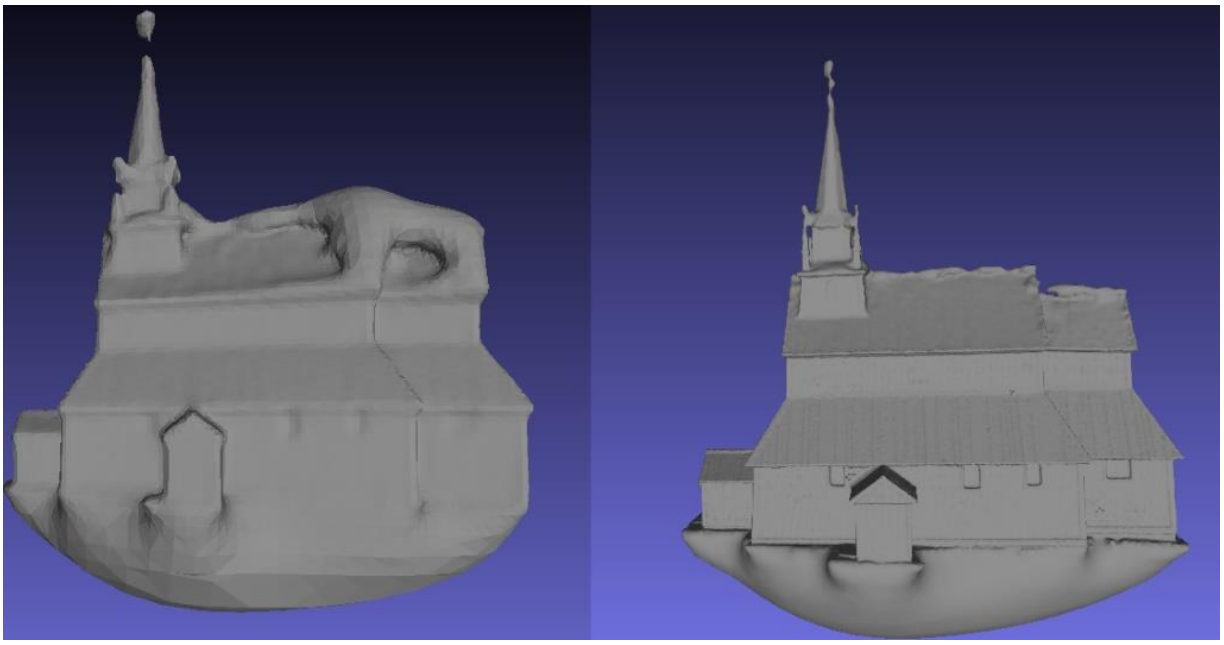
«Reconstruction depth» angir dybden på okt-treet. Denne parameteren påvirket detaljene på overflaten. Ved å øke verdien på denne parameteren ble overflaten glattere.



Figur 35: Parameter verdier brukt i Screened Poisson recontruction.

«Reconstruction Depth» på 10 og «Minimum Number of Samples» på 20 gav det beste resultatet.

Figur 36 illustrerer forskjellen i overflaten mellom bruk av standard parameterverdier og verdiene vi brukte i den endelige modellen. Disse verdiene reduserte konsekvensen av gjenværende støy i punktskyen og gav best detaljnivå.



Figur 36: Overflate generert med ulike parametere.

Modellen ble teksturert med funksjonen «Parameterization + texturing from registered rasters». Siste steg i prosessen var å redusere modellen til en håndterbar størrelse. Med tanke på fremvisning på en webapplikasjon var det ønskelig med en liten modell for å redusere innlastingstiden. Modellen hadde opprinnelig 13 556 826 «faces» og 6 779 847 «vertices».

Til å forenkle modellen brukte vi funksjonen «Simplification: Quadric Edge Collapse Decimation (with texture)». Deretter var det en iterativ prosess for å komme frem til en størrelse som gjorde modellen håndterlig.

Ved hjelp av RealityCapture ble det mulig for oss å kombinere punktskyen fra Leica P20 med bilder tatt med Nikon D600 for å lage et teksturert nett av treangler. Vi lastet inn en laserskannsky med 199 654 396 punkt og 390 bilder tatt med Nikon D600 i programmet. Bildene ble brukt til å lage en midlertidig punktsky slik at kameraposisjonene ble plassert riktig i forhold til laserpunktskyen. Deretter ble det lagd et polygonnett av kun laserpunktskyen uten noen form for reduksjon (desimering), eller rensing. Polygonnettet ble teksturert med bildene fra kameraet. Resultatet vises i figur 43 under kapittel 7.3.

6 Resultat

6.1 Webapplikasjon

Figur 37: Kartvindu.

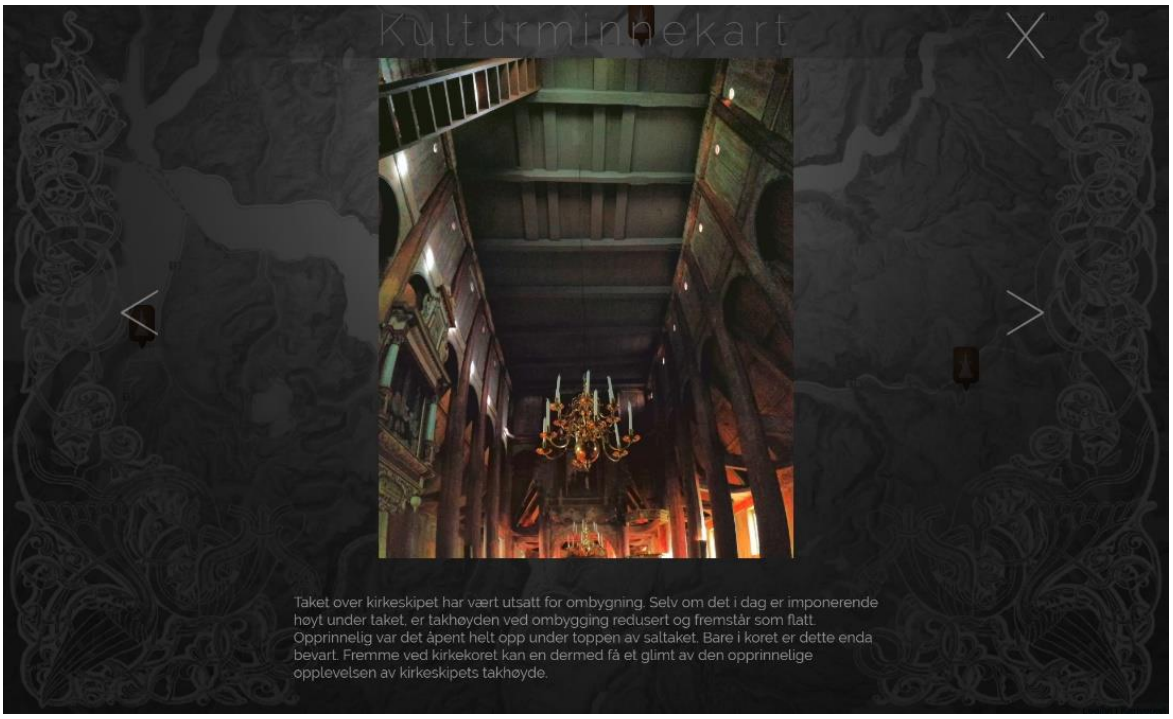
Webapplikasjonens åpningsside vises i figur 37. Kulturminnene blir plottet med ulike ikoner basert på den tilhørende kulturminnetypene. Dersom man holder musepekeren over et ikon, vil navnet på det aktuelle kulturminnet bli fremvist. Ikonene endrer størrelse basert på zoomnivå, slik at de minskes når man zoomer ut fra kartet.

Figur 38: Viser infoboks og bildegalleri som vises ved klikk på kartikon

Dersom man klikker på et ikon vil informasjonen om det aktuelle kulturminnet vises i en boks midt på skjermen, som vist i figur 38. Under boksen blir det dannet et bildegalleri der 3D-modellene får et overlag som gjør at man kan skille 3D-modellene fra bildene. Bildegalleriet har rullefelt horisontalt og tekstfeltet i informasjonsboksen har rullefelt vertikalt. Klikkbare elementer, som bilder i bildegalleriet og krysset i infoboksen, vil gløde dersom musepekeren holdes over. Klikkes det utenfor kartet, eller på krysset, vil informasjonsboksen og bildegalleriet forsvinne.

Figur 39: 3D-fremviser.

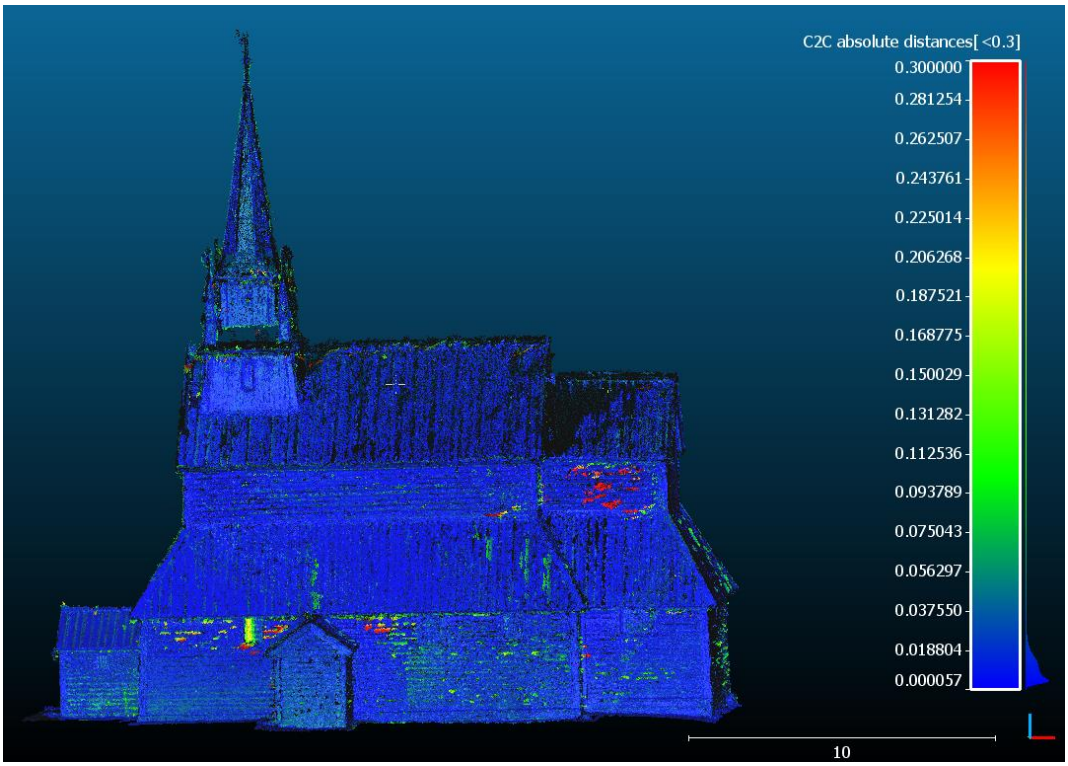
Figur 39 viser hvordan 3D-modellen fremvises i webapplikasjonen. Når man trykker på en 3D-modell i galleriet (figur 38) vil informasjonsboksen og galleriet bli skjult, og 3D-modellen lastet inn. Hele applikasjonsvinduet vil da fungere som en 3D-fremviser, der man kan rotere og zoome inn og ut fra modellen. Ved klikk på krysset vil 3D-viseren stoppe. Deretter blir informasjonsboksen og bildegalleriet fremvist.



Figur 40: Bildefremviseren.

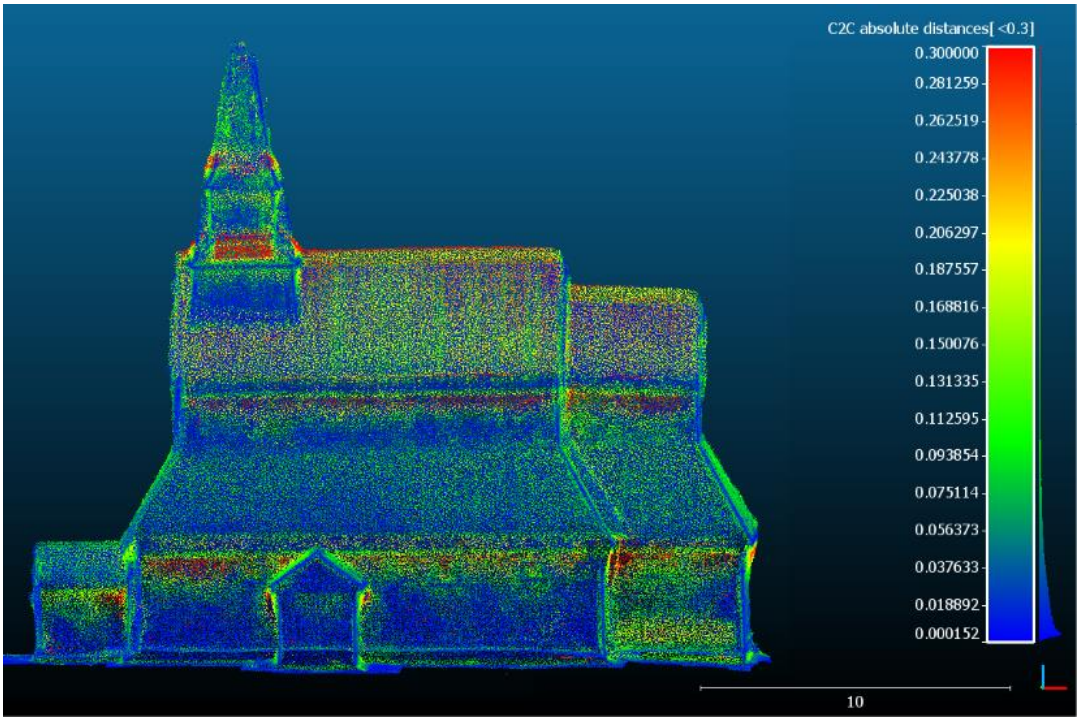
Bildefremviseren vises i figur 40. Denne blir bygget i bakgrunnen etter at bildegalleriet og tekstboksen (figur 38) er konstruert. Ved klikk på et bilde i bildegalleriet dukker fremviseren opp og viser bildet i sin helhet. Rullefeltet med informasjonsteksten under bildet viser lagret innhold til hvert enkelt bilde (se figur 13 ER-diagram databasestruktur). Klikk på pilene veksler mellom de ulike bildevinduene. Klikk på krysset lukker bildefremviseren og viser bildegalleri og informasjonsboks.

6.2 Sammenligning av punktskyer



Figur 41: Cloud/cloud dist. lasersky og fotogrammetrisky. Skala avgrenset til 30 cm.

Figur 41 viser fotogrammetriskyen med et skalarfelt som representerer avstanden i meter, mellom laserskyen og fotogrammetrisky. Resultatet fra sammenligningen av laserskannskyen og fotogrammetriskyen viser en gjennomsnittlig avstand på 1.1 cm med et standardavvik på 1.6 cm. Bilde 43 viser differansen mellom punktskyene.



Figur 42:Cloud/cloud dist. lasersky og dronesky. Skala avgrenset til 30 cm.

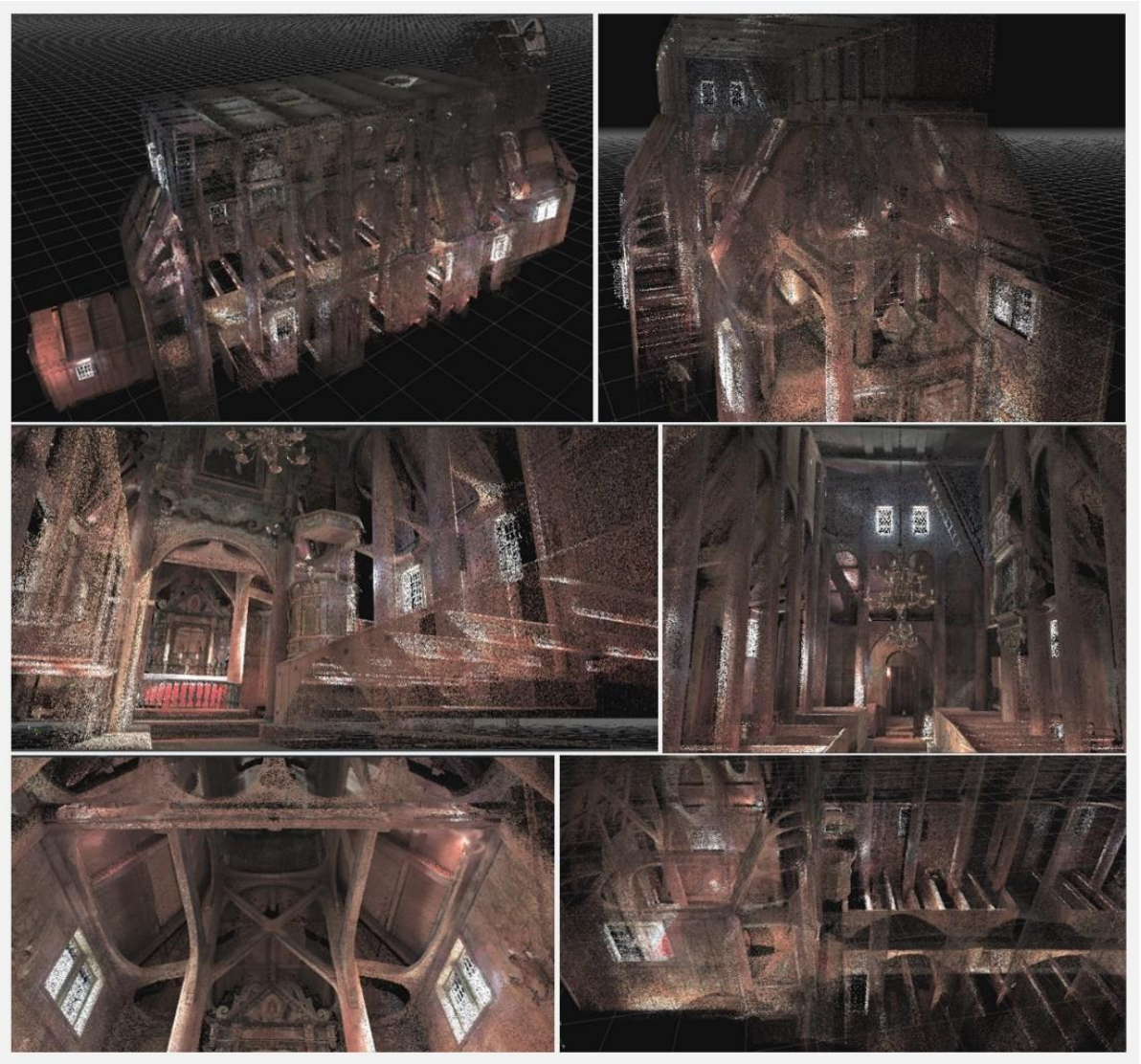
I figur 42 vises droneskyen med et skalarfelt som representerer de beregnede avstandene mellom laserskyen og droneskyen i meter. Resultatet fra sammenligningen mellom laserskannsky og droneskyen viser en gjennomsnittlig avstand på 6.3 cm med et standardavvik på 6.9 cm.

6.3 Utarbeidelse av polygonmodell

Figur 43: Side om side visning av ukomprimert og kraftig forenklet polygonmodell

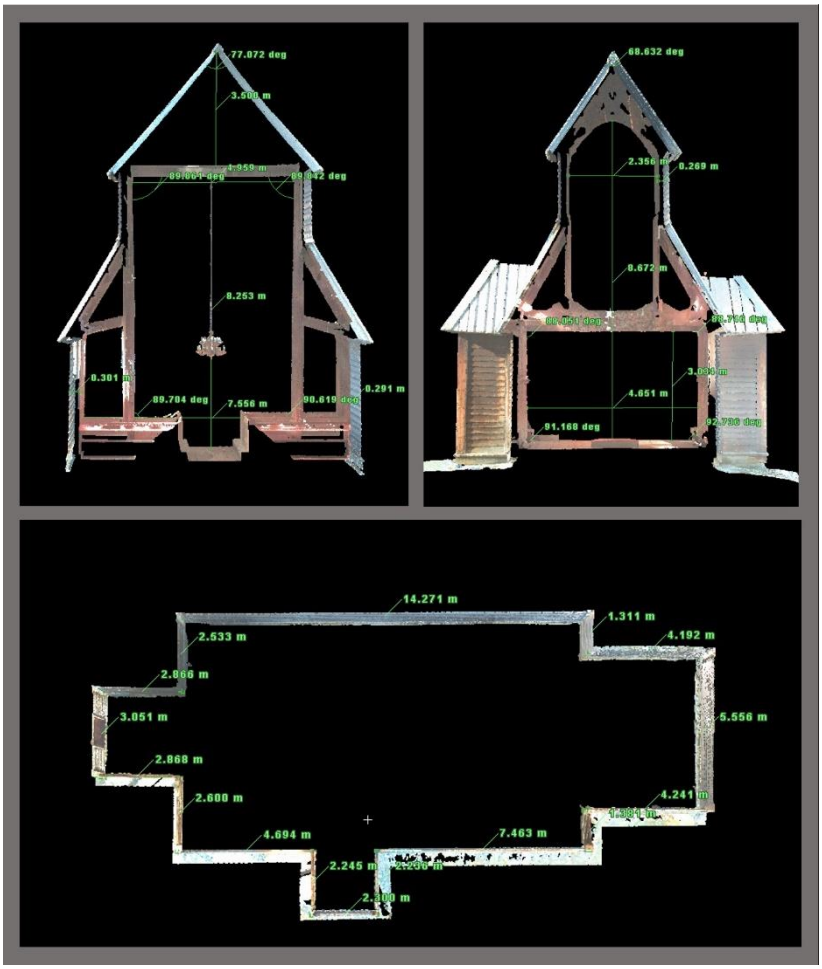
De to modellene vises side om side i figur 43. Polygonmodellen fra RealityCapture er basert på punktskyen fra Leica P20 skanneren, teksturert med bilder fra Nikon D600 kameraet. De grønne områdene på bakken i RealityCapture-modellen er innmålte punkt fra laserskyen, som ikke er dekket i bildesettet fra Nikon D600. Polygonmodellen fra MeshLab er basert på en punktsky generert av bilder fra Nikon D600, og teksturert med de samme bildene. Modellen fra RealityCapture har om lag 1000 ganger så mange "vertices" og "faces" sammenlignet med modellen fra MeshLab (se figur 43).

6.4 Innvendig skann



Figur 44: Innvendig punktsky, vist i 3DF Zephyr

Figur 44 viser resultatet fra den innvendige skanningen gjort med Leica P20 etter utført histogramstrekk av bildene fra skanneren. Punktskyen er importert i 3D-moddeleringsprogrammet 3DF Zephyr 3.702. I dette programmet har det blitt laget en gjennomflygningsvideo, vedlegg 10.4.6.

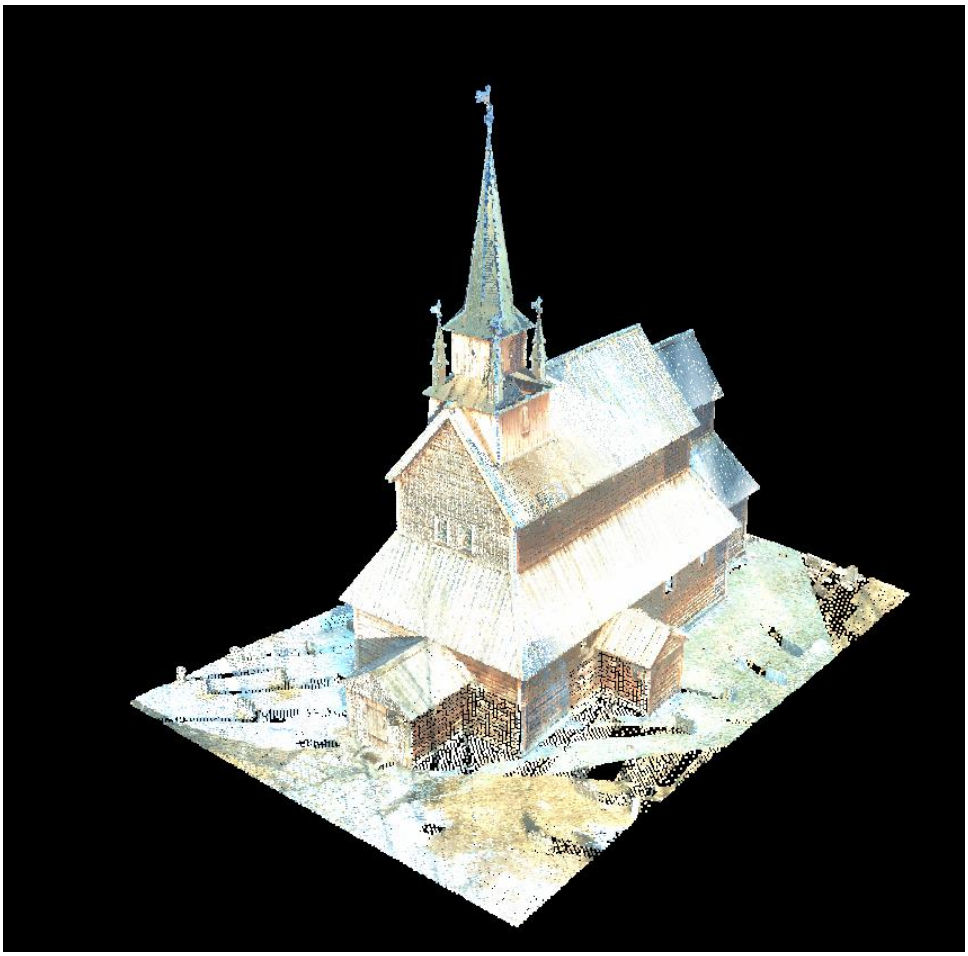


Figur 45: Måling i Leica Cyclone.

Ved presentasjon av stavkirken, kan det være interessant å gi et inntrykk av bygningens dimensjoner. Ved hjelp av Leica Cyclone ble det derfor utført målinger i tverrsnitt av kirkeskipet og kirkens kor, samt et utklipp av kirkens omriss nært grunnmuren.

6.5 Innmåling av blinker og registrering

Utjevningen viste et standardavvik på de innmålte blinkene på 1-3mm. For koordinatverdier, henvises det til vedlegg 10.2.1. Registrering av de utvendige punktskyene i Cyclone gav en gjennomsnittlig samlet feil på 4mm (vedlegg 10.2.3). Registrering av innvendige punktskyer resuleterte i en samlet feil på 2mm (vedlegg 10.2.2). Figur 46 under, viser den utvendige punktskyen fra laserskanneren, påført RGB verdier basert på bilder fra skannerens innebygde kamera.



Figur 46: Viser registrerte laserdata med RGB verdier fra skannerens autoinstilte kamera.

7 Drøfting av resultater

7.1 Webapplikasjon

En utfordring i arbeidet med utvikling av webapplikasjonen var at vi hadde lite programmeringserfaring. En viktig del av denne oppgaven var derfor å gjøre grundig research på forhånd, slik at vi fant programvare som var godt egnet til vårt formål. Under arbeidet ble det klart at vi hadde truffet godt, i og med at alle delene av webapplikasjonen har god integrering med hverandre. Programvarene er også er såpass populære at vi fant nødvendig dokumentasjon på nettet til å løse alle utfordringene som kommet opp.

Ettersom de ulike typene kulturminner er noe begrenset i vår webapplikasjon, var det hensiktsmessig å gi kulturminnetypene primærnøklene 1,2 og 3 (se tabell 1). Ved utvidelse av webapplikasjonen til å inneholde flere typer kulturminner kunne det vært aktuelt å gi primærnøkkelverdiene en overført betydning. For eksempel ved å la de første sifrene være en grovere inndeling etter kulturminnets tidsalder, eller om kulturminnet er en bygning eller ikke. Dersom det videre skal tilføres egenskaper som er spesifikke for ulike kulturminnetyper, må det gjøres en ny avveining om metode 2 eller 3 er mest egnet til håndtering av arv (se figur 2). Dersom det er få egenskaper som skiller seg ut, og de er felles for mange ulike kulturminnetyper, vil metode 3 fortsatt være godt egnet. Om det tilføres egenskaper som er svært variert, vil metode 2 være en et godt alternativ.

Vi valgte omskalering av kartikoner ved ulike zoomnivåer som løsning på utfordringen med overlappende kartikoner. Dette fungerte etter vår vurdering godt med den relativt lave tettheten av kulturminner i vårt datasett. Ved en vesentlig høyere tetthet av kulturminner i kartet, kunne samleikoner på høyere zoomnivåer muligens vært en supplerende løsning. Videre ville en økning i omfang av kulturminnetyper krevd mulighet til å sortere, vise og skjule de ulike kulturminnetypene i ulike kartlag.

Det var viktig for oss at brukerne av webapplikasjonen beholdt følelsen av å befinne seg på samme sted. Vi valgte derfor å gjøre 3D-viseren gjennomsiktig slik at kartet skinte gjennom, uten å være responsivt mens modellen vises. Når 3D-viseren er fremme bidrar det mørke overlaget til at 3D-modellen blir fremhevet. Vi valgte også å skjule informasjonsboksen og bildegalleriet, da dette gav unødvendig støy i bildet.

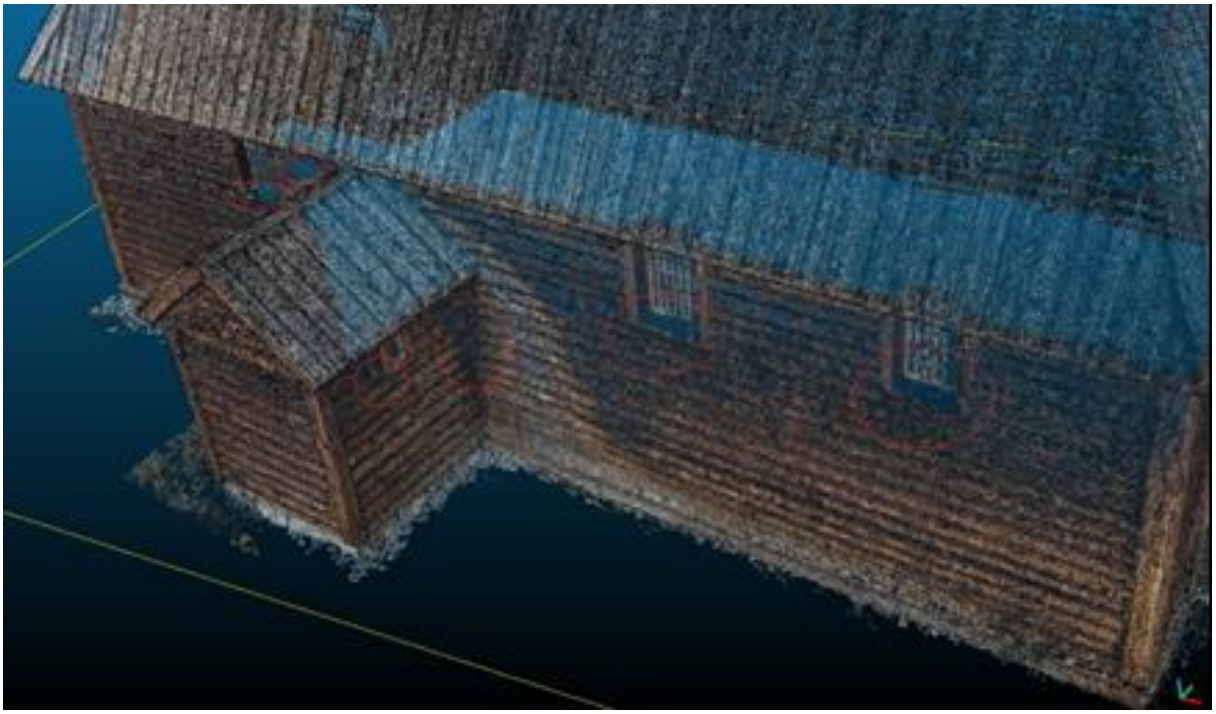
At klikkbare elementer lyser opp når musen holdes over er strengt tatt ikke nødvendig. Det gir imidlertid en god visuell indikasjon på at det skjer noe når man klikker på bildene, og gir med dette et mer responsivt inntrykk.

Lukking av tekstvinduet og bildegalleriet kunne løses på to måter. Enten ved klikk i kartet eller ved å klikke på et kryss i hjørnet av boksen. Ved å utelate lukking av boksene ved klikking i kartet ville siden oppleves mindre responsiv. På den annen side vil det være ønskelig med et visuelt virkemiddel som viser brukeren hvordan boksen kan lukkes. Særlig for en ny bruker som besøker siden for første gang. Vi valgte derfor å inkludere begge da det gjør siden både responsiv og intuitiv.

7.2 Sammenligning av punktskyer

Laserskannskyen og fotogrammetriskyen ble innmålt i samme grunnlagsnett i NTM-koordinater. Noe som gav et godt grunnlag for sammenligningen. Droneskyen er ikke innmålt i det samme grunnlagsnettet. Andre kontrollpunkter er brukt med UTM-koordinater. For å kunne gjøre en sammenligning er derfor laserskyen blitt registrert med UTM-koordinater til den aktuelle sammenligningen.

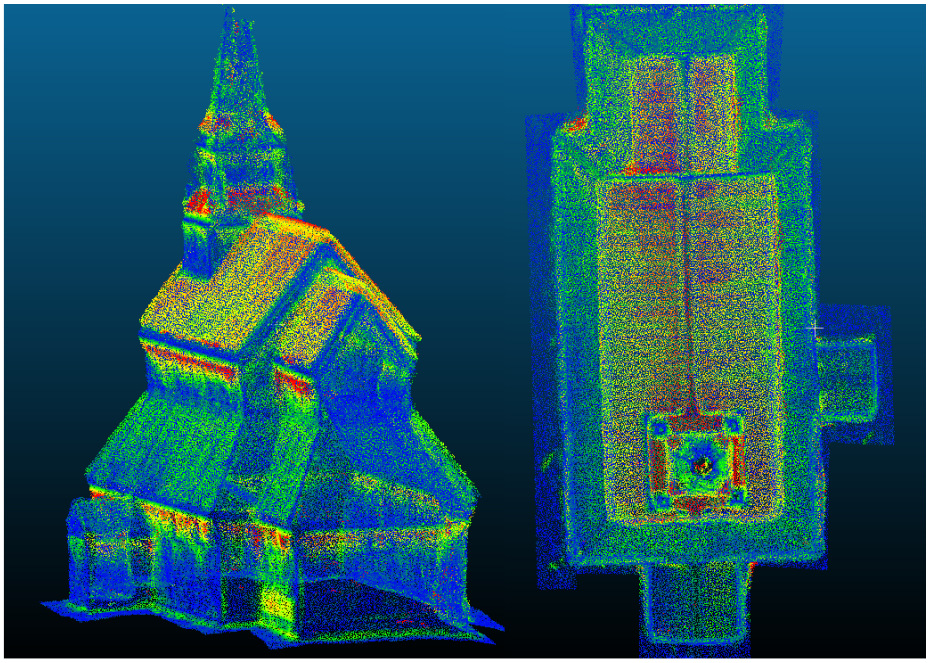
Når det kommer til fotogrammetriskyen hadde denne en gjennomsnittlig avstand fra laserskannskyen på 1.1cm. Noen områder i punktskyen skilte seg ut. Det kan synes som om flere større avstander foreligger på kirkens langside med et utbygg (røde punkt se figur 41). Når vi zoomet inn på punktskyen, så vi en mulig årsak til de større avstandene på denne siden. Rundt vinduene og innerst i hjørnene på utbygget var det få punkt. Avstandene beregnet ved hjelp av en nærmeste nabofunksjon, kan ha blitt målt til omkringliggende punkt. Hullene har ikke vært store nok til at algoritmen har latt være å beregne en avstand.



Figur 47: Punktsky av kirkens langside med tilbygg.

Samlet sett kan dette tyde på at rekonstruksjonen fra bildene som ble tatt på denne langsiden har skapt noe støy i enkelte områder. Tross dette var gjennomsnittlig avvik på 1,1cm fra laserskyen, etter vår vurdering, et tilstrekkelig modelleringsgrunnlag for et nettbasert formidlingsformål.

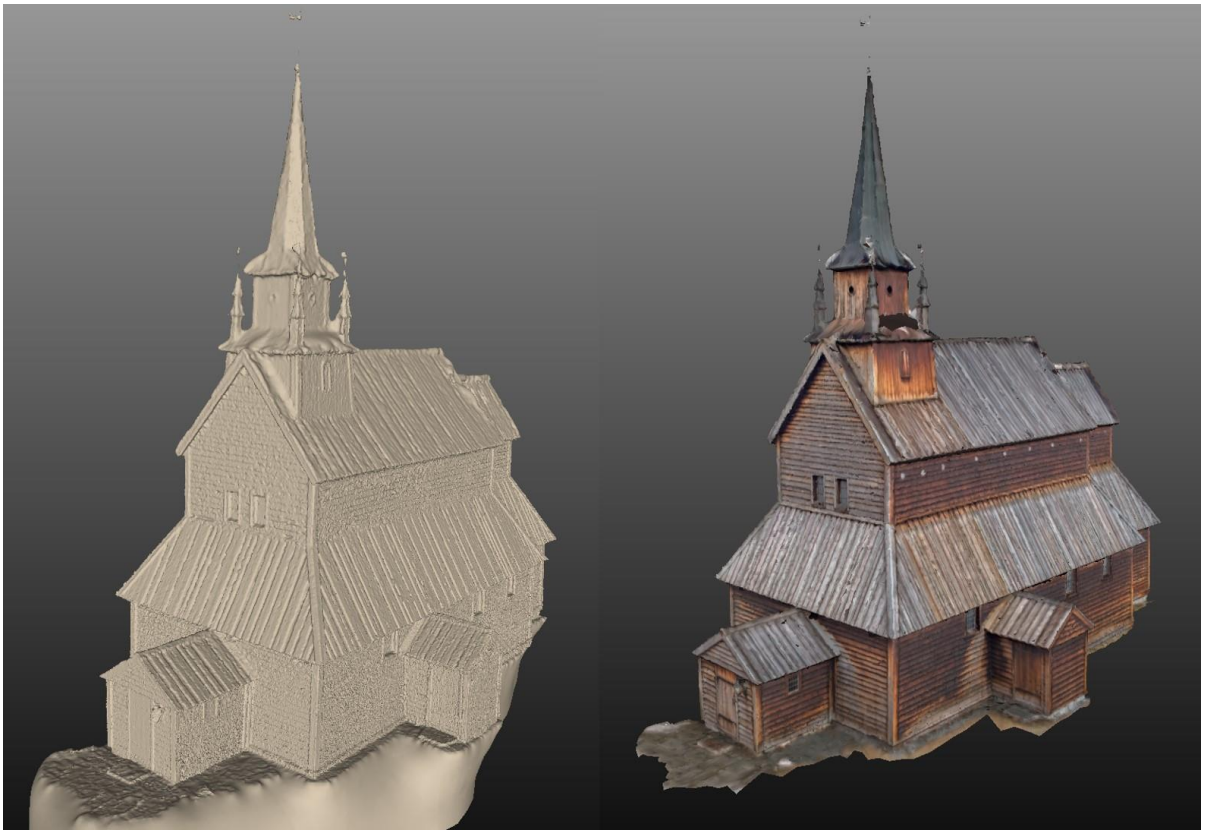
Droneskyen hadde en gjennomsnittlig avstand fra laserskyen på 6.3cm. På figur 48 under kan det synes som mange av de største avstandene (røde pkt.) ligger under gesimskassene. Punktene på taket over og under er bedre bestemt. Dronen har flydd i et grid frem og tilbake over kirken mens bilder har blitt tatt. Innfallsvinkelen kan ha gjort at punktene under gesimskassene har blitt dårligere bestemt. Igjen har derfor avstanden målt mellom punktskyene blitt større her.



Figur 48: Dronesky med skalarfelt basert på avstandsberegning.

Fra figur 48 ser vi en antydning til at taket har doble kanter og møner. Dette kan være fordi droneskyen er satt sammen av flere punktskyer som ikke er komplett overlappende. Avstandsberegningen er basert på en nærmeste nabofunksjon. Derfor vil det være viktig at modellene er plassert korrekt i forhold til hverandre. Overlappingen i droneskyen kan ha bidratt til noe større avstander sammenlignet med fotogrammetriskyen.

7.3 Utarbeidelse av 3D-modell



Figur 49: Viser den forenklede modellen (OBJ-fil) fra MeshLab, med og uten tekstur.

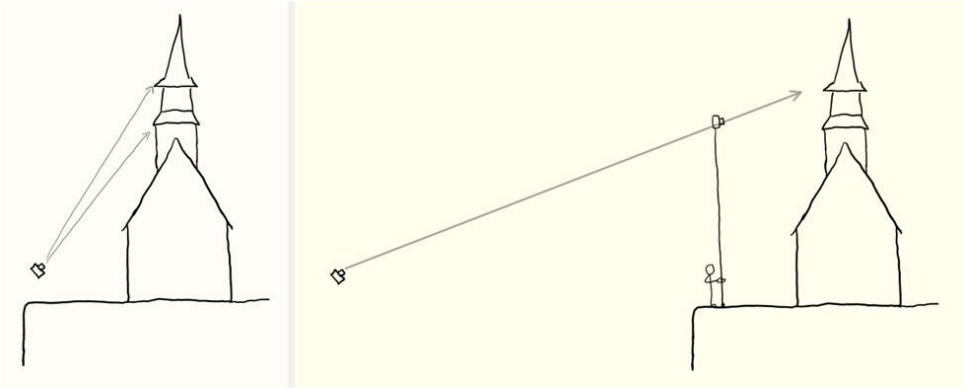
For å oppnå en tilstrekkelig komprimert OBJ-fil som kunne vises på nett ble det nødvendig å foreta en kraftig reduksjon av detaljrikdommen i overflatene og teksturen på modellen fra MeshLab. Etter en visuell vurdering av kvalitetstapet illustrert i figur 41 er resultatet fra Meshlab etter vår vurdering godt nok til et formidlingsformål.

Visuelt er det ikke tvil om at den ukomprimerte modellen fra RealityCapture har gitt det beste resultatet. Denne modellen er imidlertid så stor og uhåndterlig at den ikke kan vises i sanntid på ordinære datamaskiner. Ved visning av en interaktiv 3D-modell, bør datamaskinen klare å rendere et tilstrekkelig antall bilder i sekundet til at det gir en god flyt. Det kan tenkes at modeller i denne størrelsesorden er håndterbare i profesjonelle arbeidsstasjoner, men for alle praktiske formål knyttet til kommersiell visning på nett er dette ikke en realitet i dag. Likevel finnes det bruksområder for slike modeller, slik som nettopp rendering av stillbilder og video, der det ikke er et krav at renderingen må foregå i sanntid.

Færre punkt på visse steder i overflaten viste seg å være den største utfordringen ved bruk av Poisson trianguleringen. Dette er illustrert i figur 36 som viser deformasjonen dette medførte. Poisson-rekonstruksjonen søker å lage en "vanntett" overflate. Når rekonstruksjonen av overflaten tar utgangspunkt i et lite utvalg og det foreligger få punkt eller mye støy i et område oppstår boblene vi ser i figuren. Særlig på delene av taket lengst borte fra spiret.

På denne siden av kirken var det av terrengmessige forhold vanskelig å få ønsket avstand og vinkel til objektet. Vi ser at overflaten på andre siden av taket har blitt bedre. Bilder tatt fra en høyere posisjon i forhold til objektet ville vært optimalt. Allikevel fikk vi bøtet på dette problemet gjennom innstillingene på funksjonene vi brukte til å rekonstruere overflaten.

En justering av datafangstmetoden kunne også vært en løsning. Om forholdene tillot å øke avstanden til objektet, ville en oppnådd et mer gunstig vinkelforhold som kunne bøte på problemet (se figur 50). Dette kombinert med en telelinse ville heller ikke gått på bekostning av oppløsningen over objektet. Da forholdene ikke tillot dette kunne en annen løsning vært å montere kamera på en påle eller et teleskopstativ og brukt fjernkontroll for å aktivere utløser. Det ville ved en slik løsning også være praktisk med et oppsett hvor visning av bildeutsnittet gikk via mobiltelefon eller lignende.



Figur 50: Mulige løsninger på utfordrende vinkelforhold til deler av objektet.

8 Konklusjon

Oppgaven viser at det i dag eksisterer tilstrekkelig utviklede verktøy til å opprette en webapplikasjon egnet til kulturminneformidling i en kombinasjon av 3D, bilder, tekst og kart. Effektiv innlasting og visning av teksturert polygonmodell over nettet krevde en vesentlig reduksjon av detaljnivået den opprinnelige datafangsten var i stand til å frembringe. Den mest fremtredende deformasjonen ved Poisson triangulering i MeshLab forekom i områdene som manglet punktdekning. Laserskanneren P20 produserte i vårt tilfelle den mest detaljerte punktskyen, med minst støy. Det innebygde kameraet i skanneren ga imidlertid et dårligere grunnlag for teksturering, sammenlignet med bilder tatt fra Nikon D600. Den komprimerte datafangsten ved hjelp av DJI Mavic Pro dronen ga et vesentlig dårligere grunnlag for å modellere frem detaljene på stavkirken, sammenlignet med P20 skanneren og Nikon D600 kameraet.

9 Veien videre

Punktsettet fra VisualSFM som dannet utgangspunktet for modellen hadde flere områder med dårlig punktdekning. Det ville derfor være interessant å se om en fullstendig punktdekning av objektet ville forbedret resultatet av Poisson trianguleringen. Her er det flere metoder som kunne være spennende å prøve ut. Mulige metodeforbedringer med hensyn på datainnsamlingen kunne være å supplere innsamlingen med kameramontering på høyt teleskopstativ, bruk av telelinse i kombinasjon med lengre avstand til objektet, eller å kombinere terrestrisk og dronebåren fotografering. Fotografi fra dronen ville også kunne benyttes til å supplere bygningsmodellen med omkringliggende landskap. Det ble ikke lagd polygonmodell av den innvendige datafangsten. Å kombinere innvendig og utvendig modell er også en mulighet som kunne være verdt å prøve ut.

Webapplikasjonen slik den er laget fremstår som en prototyp for et konsept, og tar kun for seg et mindre antall kulturminnetyper. Ved en realisering av webapplikasjonen ville det være naturlig å utvide databasestrukturen til å kunne håndtere flere kulturminnetyper, samt egenskaper knyttet til disse.

10 Vedlegg

10.1 Utskrift applikasjonskode

11.1.1 admin.py

11.1.2 index.html

11.1.3 load_map.js

11.1.4 load_model.js

11.1.5 main.css

11.1.6 models.py

11.1.7 settings.py

11.1.8 urls.py

11.1.9 views.py

10.2 Rapporter

11.2.1 Gemini utgjevningsrapport blinker NTM

11.2.2 Cyclone registrering innvendig NTM

11.2.3 Cyclone registrering utvendig NTM

11.2.4 Cyclone registrering utvendig Euref89

10.3 Rådatafiler

11.3.1 Blinker fotogrammetri

11.3.2 Blinker innvendig

11.3.3 Blinker utvendig

11.3.4 Fastmerker frioppstilling

11.3.5 Fastmerker GNSS

10.4 Vedlegg ekstern harddisk

11.4.1 Bilder i RAW fra Nikon D600

11.4.2 Cyclone prosjektfiler

11.4.3 Modell til webapplikasjon

11.4.4 Punktsky fra VisualSFM

11.4.5 Punktsky fra Drone

11.4.6 Video gjennomflygning punktsky fra skanner

11.4.7 Video innsetting data i database

11.4.8 Video webapplikasjon

11.4.9 Webapplikasjon prosjektfiler

11 Referanser

ABM-utvikling, Norsk kulturråd og Riksantikvaren (u.å.) Hva er kulturarv? [Internett]. Tilgjengelig fra: <http://kaffkultur.com/hva-er-kulturarv/> [Lest 22. mai 2018].

Agafonkin, V. og Leaflet (2017) *WMS in Leaflet* [internett]. Tilgjengelig fra: <https://leafletjs.com/examples/wms/wms.html>[Lest 12.05.2018].

Angel, E. og Shreiner, D. (2015) Teaching an Introductory Computer Graphics Course with WebGL. I: Cozzi, P. red. *WebGL Insights.* Boca Raton: Taylor & Francis Group, s. 107-119.

Bjerknes, K., Liden, H. og Tschudi-Madsen, E. (1975) *The stave churches of Kaupanger*. Oslo: Fabritius Forlag.

Bedford, j. (2017) *Photogrammetric Applications for Cultural Heritage. Guidance for Good Practice*. [Internett]. 2017 Okt. Swindon: Historic England. Tilgjengelig fra: <https://historicengland.org.uk/advice/technical-advice/recording-heritage/> [Lest 17.04.2018].

Boehler, W. og Marbs, A. (2003) *Investigating Laser Scanner Accuracy* [Internett]. Mainz: i3mainz. Tilgjengelig fra: <https://hds.leica-geosystems.com/hds/en/Investigating_Acurracy_Mintz_White_Paper.pdf> [Lest 13.05.2018].

Bratbergsengen, K. (26.sep. 2017) *Relasjonsmodellen*, i Store norske leksikon [Internett]. Tilgjengelig fra: <https://snl.no/relasjonsdatabase> [Lest 01.05.2018].

Carver, S., Cornelius, S. og Heywood, I. (2011) *An Introduction to Geographical Information Systems*. 4. utg. Harlow: Pearson Education Limited.

CloudCompare.org (2015) Cloud-to-Cloud Distance [Internett]. Oppdatert 6. oktober 2015. Tilgjengelig fra: <http://www.cloudcompare.org/doc/wiki/index.php?title=Cloud-to-Cloud_Distance> [Lest 13.03.2018].

Dick, Ø.B. (2018/ 20. februar 2018) *Fotogrammetri*, i Store norske leksikon [Internett]. Tilgjengelig fra: <https://snl.no/fotogrammetri> [Lest 29.04.2018].

Django. Django Software Foundation. (2018) *Django Documentation: Release 2.0.6.dev20180503013239* [Internett]. Oppdatert 03.05.2018. Tilgjengelig fra: <https://media.readthedocs.org/pdf/django/2.0.x/django.pdf> [Lest 06.05.2018]

ESRI (2017) *What is Python?* [internett]. Tilgjengelig fra: <http://desktop.arcgis.com/en/arcmap/latest/analyze/python/what-is-python-.htm> [Lest 2018-05-07].

Flask (2018) *Flask Documentation Release 0.13.dev* [Internett]. Oppdatert 18.09.2017 Tilgjengelig fra: <https://media.readthedocs.org/pdf/flask/latest/flask.pdf> [Lest 08.05.2018].

Google maps (u.å.) [Internett]. Tilgjengelig fra:<https://www.google.no/maps/@61.2112282,6.3641085,9.58z> [Lest 20.02.2018].

Gustavsen, L. (2009) *Laserskanning av Urnes stavkirke, Luster kommune, Sogn og Fjordane* [Internett]. NIKU; 2009. Tilgjengelig fra:

<https://brage.bibsys.no/xmlui/bitstream/handle/11250/176427/Stavkirkeprogrammet_Urnes_NIKUOppdragsrapport_180_2009.pdf?sequence=1> [Lest 14.05.2018].

Kristoffersen B. (2016) *Databasesystemer*. 4. utg. Oslo: Universitetsforlaget.

Kulturminneloven. Lov 9. juni 1978 nr. 50 om kulturminner.

Leica Geosystems (2014) *Leica ScanStation P20: User Manual*. 3. utg. Heerbrugg: Leica Geosystems AG. Tilgjengelig fra: <https://kb.sccssurvey.co.uk/download/139/leica-p15-p20/2201/leica-scanstation-p20-user-manual.pdf> [22.05.2018].

Makina Corpus (2017) *Django Leaflet Documentation Release 0.20* [Internett]. Versjon 28.11.2017. Tilgjengelig fra: <https://media.readthedocs.org/pdf/django-leaflet/latest/django-leaflet.pdf> [Lest 08.05.2018].

Matthews, N. A. (2008) *Aerial and Close-Range Photogrammetric Technology: Providing Resource Documentation, Interpretation, and Preservation* [Internett]. Denver, Colorado: Bureau of Land Management. Tilgjengelig fra: <https://www.blm.gov/nstc/library/pdf/TN428.pdf> [Lest 03.03.2018].

Morgan, J. A. og Brogan, D. J. (2016) *How to VisualSFM* [Internett]. January 2016. Fort Collins, Colorado: Colorado States University. Tilgjengelig fra: <https://d32ogoqmya1dw8.cloudfront.net/files/getsi/teaching_materials/high-rez-topo/visual_sfm_tutorial.pdf> [Lest 27. mars 2018].

PostGIS (2018) *PostGIS 2.4.5dev Manual, rev. 16550* [Internett]. Versjon 21.04.2018. Tilgjengelig fra: <https://postgis.net/stuff/postgis-2.4.pdf> [Lest 28.04.2018].

PostgreSQL. The PostgreSQL Global Development Group (2018) *PostgreSQL 9.6.8 Documentation* [Internett]. Tilgjengelig fra: <https://www.postgresql.org/files/documentation/pdf/9.6/postgresql-9.6-A4.pdf> [Lest 23.04.2018].

QGIS. QGIS Project (2018) *QGIS User Guide Release 2.18.* [Internett]. Versjon 07.05.2018. Tilgjengelig fra: <https://docs.qgis.org/2.18/pdf/en/QGIS-2.18-UserGuide-en.pdf> [Lest 07.05.2018]

Riksantikvaren (2018) *Høyring Riksantikvarens digitaliseringsstrategi 2018 - 2021*. Forvaltningsarkivet REF. 18/00996-1. Oslo: Direktoratet for kulturminneforvaltning.

Riksantikvaren (u.å.a) *Askeladden* [Internett]. Oslo: Direktoratet for kulturminneforvaltning; u.å. Tilgjengelig fra: <https://www.riksantikvaren.no/Veiledning/Data-og-tjenester/Askeladden> [Lest 03.05.2018].

Riksantikvaren (u.å.b) *Kaupanger stavkyrkje/ Kirkested* [Internett]. Tilgjengelig fra:< https://kulturminnesok.no/minne/?queryString=https%3A%2F%2Fdata.kulturminne.no%2Faskeladden%2Flokalitet%2F84766 >[Lest 23.01.2018].

Rosen, R. og Shklar. L. (2003) *Web Application Architecture: Principles, Protocols and Practices.* 1. utg. West Sussex: John Wiley & Sons, Ltd.

Sketchfab (u.å.) *h* [internett]. Tilgjengelig fra: <https://sketchfab.com/models/70ca2fbacde140339ab18effe7010084> [Lest 29.01.2018].

Skaar, J. (2018/ 20. februar 2018) *albedo*, i Store norsk leksikon [Internett]. Tilgjengelig fra: <https://snl.no/albedo> [Lest 13.05.2018].

Statens Kartverk (2017) *Kartprojeksjonar* [internett]. Sist oppdatert 12.07.2017. Tilgjengelig fra: <https://www.kartverket.no/kunnskap/kart-og-kartlegging/Jordas-rutenett/Kartprojeksjoner/> [Lest 12.05.2018].

Statens Kartverk (2018) *Kartverkets API-er og tjenester* [internett]. Sist oppdatert 16.05.18. Tilgjengelig fra: <https://www.kartverket.no/data/api-og-wms/> [Lest 05.05.2018].

Storsletten, O. (1993) *En arv i tre: de norske stavkirkene*. Oslo: Aschehoug.

The Stave Church Portal (u.å) *Kaupanger stavkyrkje* [Internett]. Tilgjengelig fra: <http://www.stavkirke.no/index.php?option=com_djclassifieds&view=item&cid=1:stavkirke&id=14:kaupanger-stavkyrkje&Itemid=127&lang=no> [Lest 23.01.2018].

UBC. University of British Columbia (u.å.) *SIFT: scale invariant feature transform* [Internett]. Vancouver: University of British Columbia. Tilgjengelig fra: <https://uilo.ubc.ca/sift-scale-invariant-feature-transform> [Lest 29.04.2018].

Vosselman, G og Maas, H. (2010) *Airborne and Terrestrial Laser Scanning*. Dunbeath: Whittles Publishing.

Øverland, O.A. (1886) *Norges Historie 02 Fra Kristendommmens Indførelse til Magnus Erlingssøns Fald* [Internett]. Kristiania (Oslo): Folkebladet. Tilgjengelig fra: <https://commons.m.wikimedia.org/wiki/File:O.A._%C3%98verland_Illustreret_Norges_Historie_02_Fra_Kristendommmens_Indf%C3%B8relse_til_Magnus_Erlingss%C3%B8ns_Fald_Folkebladet_Kristiania_1886_Page_538_Portal_fra_Lomens_kirke_Lomen_stave_church.jpg> [Lest 20.01.2018].

Wu, C. (u.å.) VisualSFM: *A Visual Structure from Motion System* [Internett]. Tilgjengelig fra: <http://ccwu.me/vsfm/doc.html> [Lest 05.03.2018].

Ørstavik, E. (21. okt. 2015) *Geografisk Informasjonssystem*, i Store norske leksikon [Internett]. Tilgjengelig fra: <https://snl.no/geografisk_informasjonssystem> [Lest 09.05.2018].