Onderzoeksrapport

Crashdetectie voor connected e-bikes

Onderzoek naar het verbeteren van de ongevaldetectie voor elektrische fietsen

Michael Hoogendoorn
HZ University of Applied Sciences
11-6-2021

Crashdetectie voor connected e-bikes

Afstudeerscriptie: Onderzoek naar het verbeteren van de ongevaldetectie voor elektrische fietsen

Auteur: Michael Hoogendoorn

Studentnummer: 077020

Emailadres: Hoog0095@hz.nl

Opleiding: HBO-ICT

Onderwijsinstelling: HZ University of Applied Sciences

Eerste examinator: Dhr. A. Nieuwenhuize

Tweede examinator: Mw. E. Schippers-Vastrick

Stagebedrijf: Conneqtech B.V.

Stagebegeleider: Dhr. L. Bij de Vaate

Voorwoord

Crashdetectie is een woord wat ik niet snel meer zal vergeten. Aan het begin van de studie had ik nooit kunnen bedenken dat ik het laatste jaar zou afsluiten met een project waarbij ik met fietsen mocht gooien in naam van de wetenschap. Achteraf had ik mij geen leukere afstudeerstage kunnen wensen.

Afgelopen half jaar heb ik de ruimte gehad om mijn interesse in data science en machine learning te combineren met softwareontwikkeling, waarbij ik heb kunnen leren over hoe iets relatief eenvoudigs als een fiets opeens een onschatbare bron van data kan worden. Deze ruimte om kennis te kunnen maken met een hoop verschillende aspecten van ICT heeft ervoor gezorgd dat ik elke dag weer met plezier aan de slag kon gaan en een scriptie op kan leveren waar ik van mezelf trots op mag zijn.

Natuurlijk heb ik dit alles niet alleen hoeven doen. Ten eerste wil ik mijn begeleider, Leon Bij de Vaate, bedanken voor de ondersteuning en sturing tijdens het onderzoek. Dankzij onze wekelijkse updates heb ik altijd het gevoel kunnen houden op de goede weg te zitten en aan iets te werken wat voor Conneqtech echt meerwaarde heeft.

Ook de andere collega's binnen Conneqtech wil ik bedanken voor de prettige samenwerking en de zichtbare interesse als ik weer eens iets presenteerde over de crashdetectie. In het bijzonder wil ik Jens nog bedanken, die altijd naar Slack leek te staren wachtend op mijn volgende vraag en gelijk dingen ging regelen zodat ik tests kon uitvoeren en alle informatie tot mijn beschikking had.

Tot slot wil ik mede-afstudeerder Jano bedanken. Omdat we in hetzelfde schuitje zaten konden we altijd praten over het onderzoek en de eisen vanuit de opleiding, mede daardoor heb ik deze scriptie met vertrouwen durven inleveren.

Veel leesplezier!

Michael Hoogendoorn

4 juni 2021

Samenvatting

Een ongeluk zit in een klein hoekje, zo ook bij elektrische fietsen waarmee eenvoudig een snelheid van 25km/u kan worden bereikt. Deze zogenaamde e-bikes nemen snel toe in populariteit en daarmee neemt ook het aantal ongelukken toe.

Bij Conneqtech wordt gewerkt aan Internet of Things (IoT) oplossingen voor onder andere e-bikes, waarbij deze worden verbonden met het internet. Deze verbinding maakt het mogelijk om data te verzamelen zoals bijvoorbeeld locatie en batterijstatus. Deze gegevens kunnen door zowel consumenten als fabrikanten gebruikt worden om inzicht te krijgen in de status van de e-bikes.

De IoT hardware maakt het ook mogelijk om beweging en crashes te herkennen. Om de mogelijkheden van crashdetectie toe te passen had Conneqtech al software ontwikkeld. dit systeem maakte echter te veel fouten in de detectie, waardoor er vanuit Conneqtech onvoldoende vertrouwen was om dit op grote schaal uit te rollen en te presenteren aan klanten.

Conneqtech was benieuwd naar de mogelijkheden om de crashdetectie te verbeteren. Uit deze vraag is dit onderzoek voortgekomen, met als doel om een verbetering van de crashdetectie te realiseren waarbij Conneqtech voldoende vertrouwen zou hebben om deze functionaliteit te gebruiken en presenteren aan klanten.

Om de vraag te beantwoorden is onderzocht hoe de beschikbare data gebruikt kon worden om crashes op een correcte manier te kunnen herkennen. Hierbij is onderzoek gedaan naar de mogelijkheden om acceleratiedata van een paar seconden vóór en na de vermeende crash te gebruiken. Om dit te onderzoeken is eerst data gegenereerd door verschillende crash scenario's te simuleren.

Vervolgens zijn er twee prototypes gemaakt: een Machine Learning model en een Go applicatie. Met behulp van de gegenereerde crash data zijn hierbij de prestaties van de prototypes geëvalueerd, waarna de keuze is gemaakt om de Go applicatie verder te ontwikkelen en implementeren. Voor het realiseren van deze applicatie is verder onderzoek gedaan naar de mogelijkheid om een valhoek te berekenen, waarbij uiteindelijk een algoritme is geïmplementeerd welke in staat is op basis van de acceleratiedata te bepalen of een fiets op zijn kant ligt na een vermeende crash.

Na implementatie is de applicatie getest binnen een testomgeving waarbij de werking is geëvalueerd. Na een correctie van de crash gevoeligheid is geconcludeerd dat de applicatie naar behoren werkt en beter presteert dan de oude crashdetectie. Op basis van de testdata is gebleken dat de nieuwe crashdetectie zeker 95% accuraat is in het correct classificeren van vermeende crashes.

Advies voor vervolgstappen zijn om de prestaties van de nieuwe crashdetectie over een langere termijn te evalueren en hierbij een systeem op de zetten waarbij gebruikers feedback kunnen geven over wat er is gebeurd na een vermeende crash. Met behulp van deze feedback kan de accuraatheid van de detectie gecontroleerd worden. Ook geeft de feedback inzicht in wanneer de applicatie fouten maakt, zodat er gericht aanpassingen gemaakt kunnen worden voor optimale prestatie.

Inhoud

Hoofdstuk 1: Inleiding

1.1 De organisatie

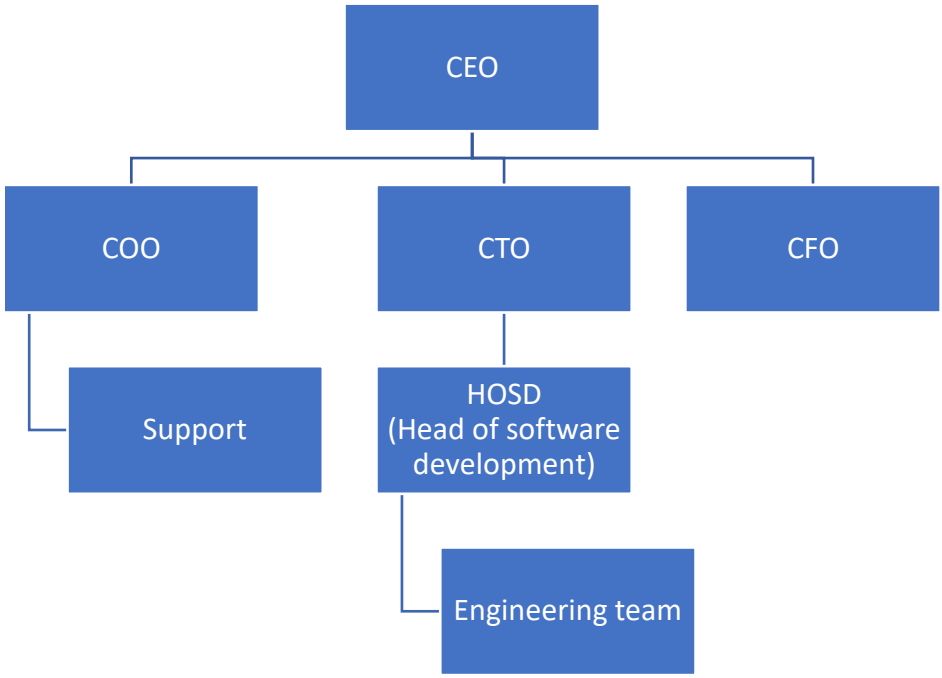
Conneqtech is een ICT-bedrijf welke oplossingen levert aan organisaties om producten en/of diensten te verbinden met het internet. Deze zogenaamde Internet of Things (kortweg IoT) oplossingen geven de klanten mogelijkheden om bijvoorbeeld de locatie van producten te volgen of de batterijstatus van een elektrische fiets in te zien met een app op een smartphone.

Conneqtech bestaat op dit moment uit 16 medewerkers en één stagiair (L. Bij de Vaate, persoonlijke communicatie, 3 december 2020) De medewerkers zijn verdeeld over twee locaties:

- Het hoofdkantoor in Bunschoten (Utrecht) van waaruit het bestuur en het support-team werkt.
- Een kantoor in Vlissingen waar een team van 9 man (het engineering team) werkt aan de ontwikkeling van software en apparatuur.

Organigram

Onderstaande Figuur 1 (L. Bij de Vaate, persoonlijke communicatie, 3 december 2020) toont de organisatiestructuur van Conneqtech door middel van een Organigram.



Figuur 1: Organigram Conneqtech 2020 [Illustratie]
(L. Bij de Vaate, persoonlijke communicatie, 3 december 2020)

De afstudeerstage en communicatie zal primair plaatsvinden via het kantoor in Vlissingen. Waarbij afhankelijk van de situatie thuis of op locatie gewerkt kan worden.

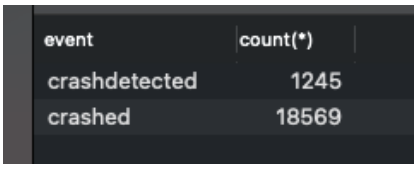
1.2 Het afstudeeronderwerp

Het onderwerp voor de afstudeerstage is: **Het verbeteren van de crashdetectie voor de elektrische fietsen.**

In samenwerking met fietsfabrikanten heeft Conneqtech instrumenten ingebouwd in elektrische fietsen waaronder g-kracht sensoren welke beweging en acceleratie kunnen meten. Als de g-krachten boven een bepaalde drempelwaarde komen wordt dit gedetecteerd als een 'crash' waarbij de bestuurder van de fiets een ongeluk zou hebben gehad. Bij een vernomen crash worden er gegevens verstuurd via het internet met data van vóór en na het crashmoment.

Een tweede softwaresysteem ontvangt en verwerkt de data van de vermoedelijke crash en bepaald of het gaat om een echte crash of dat bijvoorbeeld de fiets is omgevallen na het afstappen. Hierbij wordt gekeken of de fiets in beweging was op het moment van de crashmelding. Als de software heeft bepaald dat er een valide crash heeft plaatsgevonden wordt er een melding verstuurd naar familieleden of andere contactpersonen die aangeeft dat er een ongeluk is gebeurd.

Uit de praktijk is gebleken dat de vermeende crashes lang niet altijd valide zijn, zo wordt de drempelwaarde ook overschreden bij het op of afrijden van een stoeprand. Dit zorgt voor veel 'false positives' en hiermee veel verstuurde data die verwerkt moet worden. Onderstaande Figuur 2 (L. Bij de Vaate, persoonlijke communicatie, 3 december 2020) laat de cijfers uit de huidige situatie zien. Hierbij zijn "crashes" het aantal keer dat de hardware in de fiets een crash detecteert en de '"crashdetected" het aantal keer dat de verstuurde crashdata na verwerking ook als valide wordt aangemerkt. In de huidige situatie betekent dit dat meer dan 93% van de gedetecteerde crashes na validatie geen crash blijken te zijn.



event	count(*)
crashdetected	1245
crashed	18569

Figuur 2: Vermeende en daadwerkelijke crashes [Illustratie]
(L. Bij de Vaate, persoonlijke communicatie, 3 december 2020)

Conneqtech wil de crashdetectie verbeteren omdat deze huidige opzet onvoldoende in staat is om crashed events correct te valideren. Klanten van Conneqtech gebruiken functionaliteiten zoals de crashdetectie in de marketing bij het verkopen van de fietsen. Als de crashdetectie wordt verbeterd door deze betrouwbaarder te maken kan Conneqtech dit aanprijzen als een belangrijke functionaliteit van het product.

Software

De software van Conneqtech welke wordt gebruikt voor het verwerken van de data uit de IoT apparaten (zoals de sensoren in de fietsen) bestaat uit verschillende microservices waarvan het grootste deel is geschreven in programmeertaal Go.

Voor het opslaan van data, waaronder crashgegevens, wordt gebruik gemaakt van een MongoDB NoSQL database.

De crashgegevens die de apparatuur van de fiets verstuurt bij het constateren van een crash bestaat uit een uniek IMEI-nummer, waarmee de fiets geïdentificeerd kan worden, en crash data. In deze data staan X, Y en Z waarden zoals gemeten door de g-kracht sensoren op een bepaald moment, deze data bevat meerdere metingen van deze waarden in een periode van enkele seconden vóór en na de gedetecteerde crash.

De aanleiding en het probleem

De crashdetectie is een veelgevraagde functionaliteit vanuit de klanten van Conneqtech. Deze klanten willen de veiligheid van de fietsen gebruiken als marketing om de verkoop te stimuleren. Op dit moment is de crashdetectie functionaliteit die Conneqtech aanbiedt nog heel basaal. De huidige opzet zorgt voor veel fouten in de detectie waardoor Conneqtech onvoldoende vertrouwen heeft in deze functionaliteit en daarom niet durft aan te prijzen aan klanten.

Vanuit Conneqtech is al bekend dat niet alle beschikbare informatie van een crash wordt gebruikt. Zo wordt momenteel de frame data nog niet ingezet om een mogelijke crash te valideren. In deze frame data zit informatie over de gemeten g-krachten gemeten in een periode voor, tijdens en na de crash.

Conneqtech wil door middel van deze afstudeerstage onderzoek laten doen naar de mogelijkheden om de crashdetectie te verbeteren en om een passende oplossing te realiseren.

De doelstelling

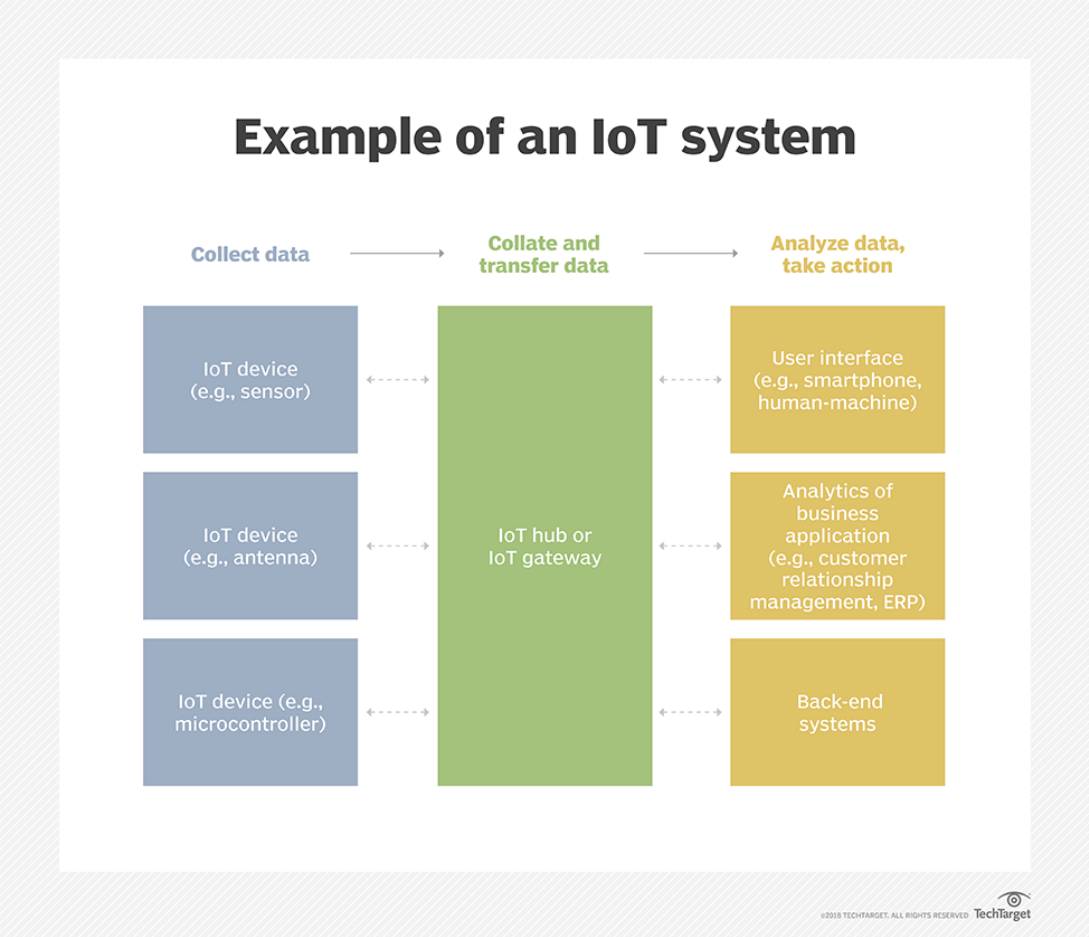
Het beoogde resultaat van deze afstudeeropdracht is een gerealiseerde verbetering van de crashdetectie waarbij Conneqtech genoeg vertrouwen heeft om deze functionaliteit aan te prijzen en hiermee een kwalitatief beter en betrouwbaarder product kan aanbieden aan de klanten.

Hoofdstuk 2: Context

IoT

Allerlei apparaten zoals koelkasten en auto's kunnen tegenwoordig verbonden worden met het internet en 'slim' gemaakt worden. Dit systeem wordt ook wel 'The internet of Things' (kortweg IoT) genoemd. Hierbij worden apparaten voorzien van computers, sensoren en een connectie met het internet zodat deze autonoom data kunnen verwerken en verzenden (Khan & Yuce, 2019, p.1).

Onderstaande Figuur 3 (TechTarget, z.d.) laat een voorbeeld zien over hoe een Internet of Things systeem kan werken.



Figuur 3: TechTarget. (z.d.). Example of an IoT system [Illustratie]. Geraadpleegd van https://cdn.ttgtmedia.com/rms/onlineimages/iota-iot_system.png

E-bike

Conneqtech werkt samen met AXA Bike Security om elektrische fietsen te voorzien van IoT apparaten en deze te verbinden met het internet. Het gebruikte IoT apparaat wordt door AXA de 'AXA IN module' genoemd. Deze module werkt samen met een smartphone app om een fiets 'slim' te maken en inzicht te krijgen in informatie zoals aantal gereden kilometers, locatie en gereden routes (AXA Bike Security, z.d.).

Deze elektrische fietsen, ook wel E-bike genoemd, zijn gebaseerd op traditionele fietsen waar een elektrische motor aan toegevoegd is om te helpen bij de voortstuwing. De motor krijgt zijn energie van een oplaadbare batterij die op de fiets is bevestigd. Een gemiddelde elektrische fiets heeft een bereik van 35 tot 50 kilometer met een snelheid van 20 kilometer per uur (Salmeron-Manzano & Manzano-Agugliaro, 2018, pp. 1-2).

Volgens de Nederlandse wetgeving hebben normale elektrische fietsen trapondersteuning tot 25 kilometer per uur. Daarnaast zijn er aparte regels voor zogenaamde speed-pedelecs, deze mogen trapondersteuning bieden tot 45 kilometer per uur (Ministerie van Algemene Zaken, 2019). Met deze snelheden vormen de gebruikers van elektrische fietsen een risicogroep waarbij moet worden nagedacht over de veiligheid. Om de veiligheid te vergroten kan gebruik gemaakt worden van sensoren die ongelukken kunnen herkennen.

Crashdetectie

Het kunnen herkennen van een val of crash kan de veiligheid voor personen verhogen door gebruik te maken van automatische systemen die communiceren met hulpdiensten of familieleden in geval van een ongeluk. Dit zorgt ervoor dat zelfs als de persoon in nood niet in staat is zelf om hulp te roepen het geautomatiseerde systeem deze hulp kan inschakelen. Om val/crashdetectie in te bouwen in een elektrische fiets kan gebruik gemaakt worden van een IoT apparaat met een versnellingsmeter.

Versnellingsmeter

Een versnellingsmeter, ook wel g-kracht meter of accelerometer, is een apparaat die lineaire en hoekversnellingen kan meten. Met een versnellingsmeter kunnen versnellingen als gevolg van schokken, inslagen, vibraties en trillingen worden gemeten (TME, 2020).

De meest gebruikte versnellingsmeters zijn 3-assig en bestaan uit drie meters welke samen de X, Y en Z as versnelling meten. Als er geen versnelling plaatsvindt wordt alleen de zwaartekracht van de aarde gemeten en zal de Y waarde 1g meten, ervan uitgaande dat de Y-as naar boven/beneden is georiënteerd (TME, 2020).

TME (2020) beschrijft drie basistypen versnellingsmeters:

- **MEMS capacitieve versnellingsmeters**
  Dit zijn de goedkoopste en kleinste sensoren welke werken met veren en een condensatie-element. Door de afmetingen worden deze meters vaak gebruikt in draagbare elektronica.
- **Piëzo-elektrische versnellingsmeters**
  Deze meters worden gebruikt voor het meten van trillingsniveaus en werken door het meten van elektrische spanning welke wordt gegenereerd door vervorming van materiaal.
- **Piëzoresistieve versnellingsmeters**
  Deze meten de vervorming in een materiaal waarbij de weerstand veranderd. Voordeel van deze meters is dat ze in staat zijn om trillingen met hoge frequenties of langzaam veranderende signalen kunnen waarnemen, wat bruikbaar is bij botsproeven en navigatiesystemen.

Algoritmes voor crashdetectie

Om op basis van de metingen uit een versnellingsmeter te bepalen of een val of crash heeft plaatsgevonden wordt gebruik gemaakt van algoritmes. Williams (2018) heeft onderzoek gedaan naar het gebruik van een voice-assistant om fietscrashes beter te kunnen melden. Hoewel het doel van dit onderzoek afwijkt van het eigen onderzoeksdoel benoemt Williams wel verschillende algoritmen die gebruikt kunnen worden om op basis van sensordata een crash te kunnen herkennen. In het onderzoek zijn de volgende algoritmen besproken:

- **Basic Threshold Monitoring**
  Een op drempelwaarde gebaseerd algoritme welke doorgaans gebruik maakt van data uit versnellingsmeters met drie assen om een 'signal magnitude vector' of SMV te berekenen. Deze SMV is een getal die de 'grootte' van de meting beschrijft.

  De SMV kan berekend worden met de volgende formule:

  $$SMV_i = \sqrt{|A_{x_i}|^2 + |A_{y_i}|^2 + |A_{z_i}|^2}$$

  In bovenstaande formule zijn Ax, Ay en Az de waarden gemeten door die drie assen van de versnellingsmeter.
  i representeert hierbij de tijd van de meting.

  Een andere formule die gebruikt kan worden is de 'amplitude vector' in relatie tot de 'absolute vertical direction', welke berekend wordt met:

  $$A_v = |A_x \sin\theta_z + A_y \sin\theta_y - A_z \cos\theta_y \cos\theta_z|$$

  Met deze algoritmen kan een waarde berekend worden waarbij als een drempelwaarde wordt overschreden dit geclassificeerd kan worden als een crash.

- **Fall Index**
  De Fall Index zoals beschreven door Yoshida et al. (2005) is een extra waarde die berekend kan worden om accurater een val te kunnen herkennen. Hierbij wordt opgemerkt dat deze waarde in sommige situaties kan helpen, maar ook de accuraatheid kan verlagen als een val (of crash) plaatsvindt over een langere tijd.

  De formule voor de Fall Index is:

  $$FI_i = \sqrt{\sum_{k=x,y,z} \sum_{i-19}^{i} ((A_k)_i - (A_k)_{i-1})^2}$$

- **Two-Phase Detection (PerFallD)**
  PerFallD is een valdetectiesysteem welke bedoeld is voor smartphones. Het gebruikte algoritme is een uitbreiding op de basic threshold monitoring en combineert de twee formules om een val te detecteren.

- **iFall**
  iFall is een ander populair algoritme gebaseerd op drempelwaardes. Het algoritme maakt gebruik van een SMV-waarde maar onderscheid zich in dat het de gebruiker tijd geeft om terug te keren naar een eerdere positie voor de val. Dit verlaagt het aantal false positives omdat als de gebruiker in staat is om direct weer op te staan dit niet als een serieuze val geregistreerd zou moeten worden.

iFall reduceert het aantal false positives verder door de gebruiker tijd te geven om de valmelding te annuleren zodat er geen onnodige melding wordt verstuurd naar aangewezen contactpersonen voor noodgevallen.

Na het detecteren van een crash met behulp van de data uit een versnellingsmeter en algoritmes moet er iets met deze informatie gedaan worden. Om de veiligheid van auto's te verbeteren bestaat een initiatief genaamd eCall. Hierbij wordt gebruik gemaakt van automatische crashmeldingen naar de dichtstbijzijnde alarmcentrale in geval van een ongeluk. (Candefjord et al., 2014, p. 1).

Candefjord et al. (2014) hebben onderzoek gedaan naar het uitbreiden van de eCall functionaliteit zodat ook de veiligheid van fietsers kan worden verbeterd. De conclusie uit dit onderzoek is dat moderne smartphones met goede hardware sensoren gebruikt kunnen worden om een val of ongeluk te kunnen herkennen en alarm te slaan. In de conclusie van dit onderzoek wordt aangegeven dat het voorgestelde algoritme op grote schaal getest moet worden in realistische crash tests om de kwaliteit te valideren.

Het voorgestelde algoritme voor het herkennen van een crash is:

$$|Acc| = \sqrt{{Acc_x}^2 + {Acc_y}^2 + {Acc_z}^2},$$

Deze formule is vergelijkbaar met de SMV-berekening zoals gebruikt bij basic threshold monitoring.

Hoofdstuk 3: Aanpak

3.1 Analyse

De eerste stap tot het behalen van de doelstelling is het documenteren van de huidige situatie en het bepalen van het precieze probleem waar een oplossing voor gevonden moet worden.

Voor het in kaart brengen van de huidige situatie zal gebruik gemaakt worden van een interview met een ervaren ontwikkelaar bij Conneqtech die kennis heeft van het bestaande systeem en kan uitleggen hoe deze werkt. Voor dit interview is het doel om genoeg informatie te verzamelen zodat de huidige werking van de crashdetectie kan worden beschreven en gevisualiseerd met een data flow diagram. Om de huidige situatie beter te begrijpen zal tijdens het interview gevraagd worden of de werking van de crashdetectie gedemonstreerd kan worden.

Om te bepalen welke belanghebbenden er zijn zal een stakeholder analyse uitgevoerd worden. Om de belangrijke stakeholders te identificeren zal gesproken worden met de opdrachtgever, vervolgens zal middels e-mail contact gezocht worden met deze belanghebbenden waarbij zij op de hoogte worden gebracht van de afstudeeropdracht. Informatie over de stakeholders zal ook gedocumenteerd worden waarbij zal worden beschreven hoe deze belanghebbenden invloed hebben op het onderzoek. Door met de stakeholder analyse de belanghebbenden te identificeren kan tijdens het onderzoek rekening gehouden worden met verschillende belangen en kunnen requirements vanuit meerdere invalshoeken gevalideerd worden.

Verdere verdieping in de huidige situatie zal uitgevoerd worden door middel van en het analyseren van de bestaande documentatie, waarbij deze doorgelezen zal worden en belangrijke begrippen en beschrijvingen gebruikt zullen worden voor het documenteren van de huidige situatie. Met gebruik van de bestaande documentatie kunnen de technische aspecten en beperkingen bepaald en gedocumenteerd worden welke als input dienen om een haalbare oplossing te ontwerpen.

Nadat de huidige situatie is beschreven zullen functionele requirements, kwaliteitseisen en andere voorwaarden voor de te realiseren oplossing verzameld worden. Hiervoor zal gebruik gemaakt worden van een interview met de opdrachtgever, een interview met een senior ontwikkelaar binnen Conneqtech en interviews met andere belangrijke stakeholders. Bij deze interviews zal gesproken zal worden over zowel functionele, technische en kwaliteitseisen waar de oplossing aan zou moeten voldoen. De requirements die voortkomen uit de interviews zullen gedocumenteerd worden en gebruikt om gericht naar een geschikte oplossing te zoeken. Voor de documentatie van het onderzoek, waaronder de requirements, zal gebruik gemaakt worden van Confluence. Conneqtech maakt gebruik van Confluence voor het beheren van kennis en informatie op een manier waarbij deze eenvoudig gedeeld kunnen worden.

Om het project beheersbaar te houden zal gebruik gemaakt worden van requirement prioritization met het toepassen van de MoSCoW methode, waarbij de requirements verdeeld worden op basis van prioriteit. De prioriteit van de requirements wordt bepaald met de volgende vier groepen (*MoSCoW – projectmanagementsite*, z.d.):

- **Must have:**
  Dit zijn de harde eisen waar de oplossing aan moet voldoen om de doelstelling te kunnen bereiken.
- **Should have:**
  Deze requirements zijn eisen waarbij het gewenst is dat de oplossing eraan voldoet, maar er is ruimte om enigszins af te wijken mits dit goed beargumenteerd kan worden.
- **Could have:**
  Dit zijn wensen waarover nagedacht kan worden als aan alle eisen is voldaan en er nog tijd en middelen over zijn om deze te realiseren.
- **Won't have:**
  Dit zijn de requirements waarbij met de opdrachtgever wordt afgesproken dat dit niet binnen de scope van het afstudeerproject valt en dus niet gerealiseerd zal worden.

De requirements zullen gevalideerd worden op compleetheid en haalbaarheid door deze na te lopen met de belangrijkste stakeholders. Uiteindelijk zal er een evaluatiemoment plaatsvinden met de opdrachtgever waarbij de requirements concreet gemaakt worden. Hierbij is het belangrijk is dat de opdrachtgever tevreden is met wat er opgeleverd gaat worden en dit haalbaar is binnen de afstudeerperiode.

De verzamelde en gevalideerde requirements dienen als input voor het opstellen en evalueren van verschillende kandidaat oplossingen.

3.2 Ontwerpen

Met de geverifieerde requirements kan gericht gezocht worden naar kandidaat oplossingen. Hierbij zal eerst door middel van deskresearch gezocht worden naar bestaande oplossingen voor vergelijkbare problemen. Met deze available product analysis (ictresearchmethods.nl, 2018b) wordt gezocht naar relevante bestaande producten die een vergelijkbaar probleem oplossen, hierbij zal gekeken worden naar bestaande oplossingen binnen de fiets- en auto-industrie en bestaande valdetectie systemen gericht op ouderen. Om te bepalen of een bestaande oplossing relevant is zal deze getoetst worden aan de opgestelde requirements, waarbij de 'must haves' leidend zijn. Door onderzoek te doen naar de bestaande oplossingen kan veel tijd bespaard worden omdat het wiel niet opnieuw uitgevonden hoeft te worden. De bestaande oplossingen kunnen niet direct gekopieerd worden maar kunnen wel dienen als belangrijke input voor het ontwerpen van een eigen oplossing.

Om het aantal kandidaat oplossingen uit te breiden zal ook gebruik gemaakt worden van de kennis binnen Conneqtech. Tijdens het verzamelen van de requirements en gesprekken met collega's zijn er waarschijnlijk al mogelijke oplossingen naar voren gekomen, hierbij is het belangrijk dat de serieuze ideeën opgeschreven worden en verder besproken met de betreffende personen. Als dit nodig wordt geacht zal een extra interviewmoment met de opdrachtgever worden gebruikt om mogelijke oplossingen te vinden en te valideren.

Met de combinatie van eigen onderzoek en de ideeën vanuit Conneqtech zal een lijst met kandidaat oplossingen naar voren komen. Het doel hier is om de twee of drie beste opties te kiezen welke verder uitgewerkt kunnen worden. Om het aantal kandidaat oplossingen terug te brengen naar twee of drie zal gebruik gemaakt worden van een multi-criteria analyse waarbij de oplossingen worden beoordeeld op nader te bepalen criteria. Hiermee kan onderbouwd een selectie gemaakt worden voor de top drie mogelijkheden.

Om een onderbouwde keuze te maken uit de geselecteerde twee of drie mogelijke oplossingen zullen deze eerst gepresenteerd worden aan de opdrachtgever. Door de mogelijkheden met de opdrachtgever kritisch te bespreken kan er mogelijk al een definitieve keuze gemaakt worden. Om de gekozen oplossing verder te onderbouwen zal gebruik gemaakt worden van prototyping waarbij een concept testversie wordt gerealiseerd voor de mogelijke oplossingen met een focus op de verschillen tussen de kandidaten. Om op basis van de prototypes een onderbouwde keuze te maken zal een testplan geschreven worden met als doel de prototypes zo te testen dat de verschillen duidelijk worden en er een uitspraak kan worden gedaan over welke optie het beste aansluit op de beschreven requirements.

De uiteindelijke keuze zal verder uitgewerkt worden met een functioneel en technisch ontwerp waarin de werking van de oplossing en de implementatie zullen worden vastgelegd. Voor deze uitwerking zal eerst een schets van een entiteit relatie diagram gemaakt worden die de relaties tussen de bestaande en nieuw te bouwen entiteiten visualiseert. Deze schets zal gevalideerd worden bij de opdrachtgever en vervolgens verder uitgewerkt worden voor de documentatie. De te realiseren oplossing zal verder uitgewerkt worden met behulp van een data-flow diagram waarin de relevante processen en datastromen worden opgenomen. Het doel van het functioneel en technisch ontwerp is om een gedocumenteerd plan te hebben welke gevalideerd kan worden door de opdrachtgever en gebruikt kan worden als handleiding om de oplossing te realiseren.

3.3 Implementatie

Om de prototypes en het uiteindelijke ontwerp te realiseren en te implementeren binnen de bestaande omgeving zal eerst een eigen werkomgeving opgezet worden met een IDE (code editor) en versiebeheersysteem. Hierbij zal zoveel mogelijk gebruik gemaakt worden van de bestaande systemen een tools vanuit Conneqtech, waarbij de hulp van collega's zal worden gebruikt om alles werkend te krijgen. Om de kwaliteit van de software te waarborgen zal regelmatig gebruik gemaakt worden van voortgangsgesprekken met een senior ontwikkelaar binnen Conneqtech welke de software inhoudelijk kan beoordelen. Om de werking van de software te testen zal gebruik gemaakt worden van praktijktests zoals het simuleren van een crash met een fiets zodat getest kan worden hoe het system hierop reageert. Ook zal gebruik gemaakt worden van dummy crashdata waarvan is gevalideerd of het wel of niet als een echte crash aangemerkt zou moeten worden. Deze data wordt dan gebruikt als input waarbij gevalideerd wordt hoe de oplossing presteert. Als een eerste werkende versie van het eindproduct is gerealiseerd zal een testplan geschreven worden waarbij het doel is om te valideren of de gerealiseerde oplossing voldoet aan de gestelde requirements. Belangrijk hierbij is dat er op basis van kwantitatieve data een percentage genoemd kan worden die uitdrukt hoe accuraat de oplossing is in het correct detecteren van een crash.

3.4 Advies

Na het realiseren van een werkende oplossing zullen er nog vragen en mogelijkheden openblijven waarbij extra tijd nodig is voor bijvoorbeeld implementatie, tests of onderzoek. Hiervoor zal een adviesrapport worden opgesteld met advies over een passend vervolgtraject om de gerealiseerde oplossing verder te verbeteren en te beheren. Om het adviesrapport een passende invulling te geven zal een evaluatiemoment met de opdrachtgever gehouden worden waarbij zal worden gesproken over de kwaliteit van de gerealiseerde oplossing en de verwachtingen vanuit Conneqtech. Op basis van het gesprek zal worden bepaald op welke aspecten van het probleem dieper zal worden ingegaan in het adviesrapport. Uiteindelijk dient dit adviesrapport als overdrachtsdocument waarbij het onderzoek kan worden afgesloten en Conneqtech genoeg informatie heeft om de gerealiseerde oplossing zelf te kunnen uitbreiden en beheren zonder verdere hulp van de afstudeerder.

Naast het eindadvies zal tussentijds ook gebruik gemaakt worden van adviesmomenten zoals bij het bepalen van requirements en het kiezen van de beste oplossing op basis van de kandidaat oplossingen.

Hoofdstuk 4: Resultaten

4.1 Analyse huidige situatie

De eerste stap voor dit onderzoek was om de huidige situatie in kaart te brengen zodat er inzicht is in de context omtrent het probleem. De analyse van de huidige situatie is gemaakt op basis van gesprekken met developers (zie bijlage notulen 1 t/m 8 +13) en de bestaande documentatie op Confluence.

In de huidige situatie bestaat er al een crashdetectie functionaliteit, deze werkt globaal met de volgende logica:

- De fiets bevat een tracker (Zie Figuur 4) welke onder andere g-krachten kan meten en data kan verzenden. Wanneer een drempelwaarde aan g-krachten wordt overschreden stuurt de tracker een ‘crashed’ event uit naar de IoT server van Conneqtech.
- De IoT server ontvangt er verwerkt de rauwe data.
- In de data processing wordt het crashed event geëvalueerd. Hierbij wordt bepaald of het om een echte crash zou gaan. De evaluatie heeft de volgende stappen:
    - Was er een rit gestart op het moment van de crash? Een rit is gestart als het apparaat meer dan 100 meter is bewogen sinds registratie van beweging.
    - Stond het apparaat aan ten tijde van de crash? Als het apparaat niet aan stond wordt aangenomen dat de fiets niet werd gebruikt.
    - Is de binnengekomen data van na het crashmoment? Dit is een check omdat er oude gebufferde data verstuurd kan worden na tijdelijk verlies van connectie.
    - Is de snelheid van de fiets na de crash meer dan 5km/u? Dit om te bepalen of de gebruiker na de vermeende crash doorgaat met fietsen, wat zou betekenen dat het niet om een serieuze crash gaat. Deze beweging wordt voor 60 seconden na de crash geëvalueerd.
    - Als de crash in bovenstaande checks niet wordt geannuleerd, wordt er een 'crashdetected' event aangemaakt. Dit zou betekenen dat het om een echte crash gaat.
- Als er een crashdetected event wordt gemaakt wordt deze ook weer verwerkt, waarbij er berichten naar ingestelde emergency contacts van de gebruiker van de fiets verstuurd worden.

Zie bijlagedocument 2.1.2 voor een uitgebreidere uitleg en een flowchart met de werking van de huidige crashdetectie.

De tracker

De tracker (Zie Figuur 4 voor CK300) is het IoT apparaat welke aan of in de e-bikes wordt gemonteerd zodat deze met internet kan worden verbonden. Het apparaat zelf bevat verschillende sensoren zodat data als locatie en batterijgegevens verstuurd en ontvangen kunnen worden via GSM. Voor dit onderzoek zal gericht worden op de CK300, deze is in staat om crashes te detecteren en crashdata te versturen.

Figuur 4: Voorbeeld tracker (CK300)

Crash data

Zoals uit de initiële probleembeschrijving vanuit Conneqtech al naar voren kwam wordt nog niet alle beschikbare crash data gebruikt om een crash te evalueren. Naast het crashed event is de tracker in staat om crash data reports te versturen met informatie van een paar seconden vóór en na de crash. Dit report bevat frame data met acceleratiegegevens als een reeks X, Y en Z waarden. Een frame is hierbij een momentopname van één seconde. Met de standaard instellingen zitten er 100 acceleratie metingen in elke frame. Na vertaling van de rauwe ASCII-waarden kan de data er op de volgende manier uit zien:

{"X":-10853,"Y":-445,"Z":-2574},{"X":-13976,"Y":5149,"Z":-23546},{"X":-15941,"Y":-610,"Z":156},…

(De waarden kunnen naar g-krachten vertaald worden door deze te delen door 2048.)

Met de standaard instellingen bestaat het totale crash data report uit zeven delen, 2 seconden aan acceleratiedata van vóór de crash en 5 seconden van na de crash. De documentatie van de tracker (CK300 @Track Air Interface Protocol [interne documentatie]) beschrijft dat er maximaal 5 seconden vóór en 10 seconden na de crash aan acceleratiedata kan worden verstuurd.

Het probleem

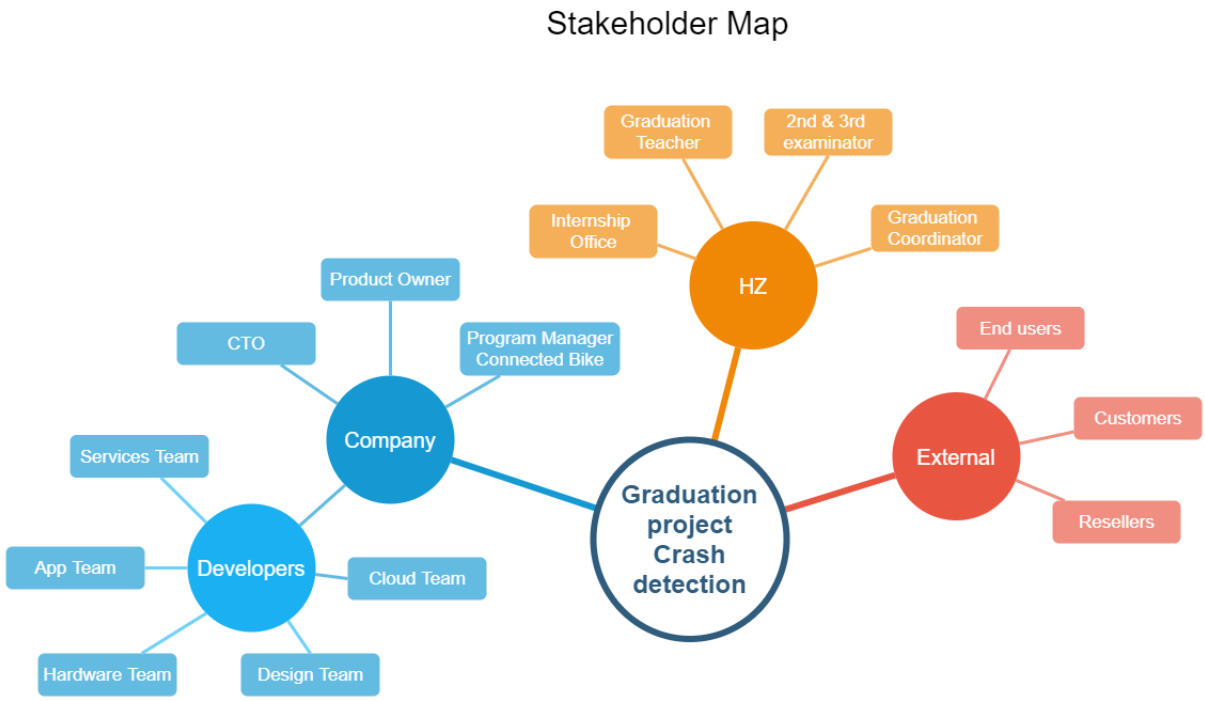
Op basis van de gesprekken met verschillende belanghebbenden en het bestuderen van de bestaande documentatie zijn de volgende problemen naar voren gekomen:

- Er is geen validatie uitgevoerd waarmee is gecontroleerd in hoeverre de huidige opzet in staat is om correct crashes te herkennen. Uit een gesprek met Rene Bolt (zie bijlage notulen 6) en documentatie die hij heeft gedeeld is duidelijk geworden dat de huidige opzet en ingestelde gevoeligheid zijn bepaald op basis van een paar fietsritten, waarbij is gekeken wat de maximale gemeten g-krachten waren.

- Er is geen geclassificeerde data met wat wel of niet een echte crash zou moeten zijn. Dit laat ook zien dat de prestaties van de huidige opzet niet objectief geëvalueerd zijn.
- Uit de binnenkomende data blijkt dat een kleine groep fietsen een groot aantal crashdetected events genereert. (Zie bijlage notulen 2)
- Door het gebrek aan inzicht en het aantal 'foutieve' crashmeldingen is er geen vertrouwen in de huidige werking van de crashdetectie.

Stakeholders

Voor dit project is het belangrijk om rekening te houden met zowel interne als externe stakeholders die mogelijk verschillende belangen en prioriteiten hebben met betrekking tot het afstudeerproject. Door de belangrijkste stakeholders te identificeren en mee te nemen in het onderzoeksproces kunnen conflicten worden vermeden en kan gebruik gemaakt worden van kennis en inzichten uit meerdere perspectieven. Onderstaande Figuur 5 toont een stakeholder map gemaakt op basis van een gesprek met de product owner (zie bijlage notulen 3) en eigen inzicht.



Figuur 5: Stakeholder map

Een groot deel van de geïdentificeerde stakeholders heeft alleen indirect invloed op het project, maar er zijn een paar belangrijke belanghebbenden die extra aandacht vereisen:

- **Opdrachtgever/ Product owner (L. Bij de Vaate):** Heeft baat bij een werkende en betrouwbare crashdetectie die door klanten gebruikt kan worden.
- **CTO (R. Bolt):** Is technisch eindverantwoordelijk en heeft baat bij een verbeterde crashdetectie die aansluit op de gebruikte hardware.
- **Het services team:** Het team waarbinnen de backend werking van de crashdetectie valt. Zij hebben baat bij goed geschreven software die functioneel en leesbaar is zodat deze eenvoudig beheert kan worden. Hulp en vragen tijdens het ontwerpen en implementeren zullen aan dit team gesteld worden.

4.2 Requirements

Met de kennis over de huidige situatie en de belangrijkste stakeholders kwam de stap om te bepalen waar de oplossing van het probleem aan moest voldoen. De requirements, scope en beperkingen van de opdracht zijn gedocumenteerd op Confluence (zie bijlagedocument 2.3). Voor het verzamelen van deze eisen en wensen is een elicitatie plan (Bijlage 1: Requirements Elicitatie Plan) opgesteld en gevolgd.

De requirements, inclusief non-functionals, scope en beperkingen zijn vastgesteld over meerdere iteraties aan gesprekken met stakeholders (zie bijlage notulen 3 + 6 t/m 11). De product owner heeft hierbij de regie gehad over de prioriteiten en scope van de opdracht. Met de product owner is ook beslist om verschillende crash scenario's te definiëren naast de requirements, waarbij is gespecificeerd of de oplossing dit wel of niet als echte crash zou moeten kunnen valideren. Door deze scenario's te definiëren werd het makkelijker om de kwaliteit en prestaties van de oplossing te evalueren.

4.3 Onderzoek naar oplossingen

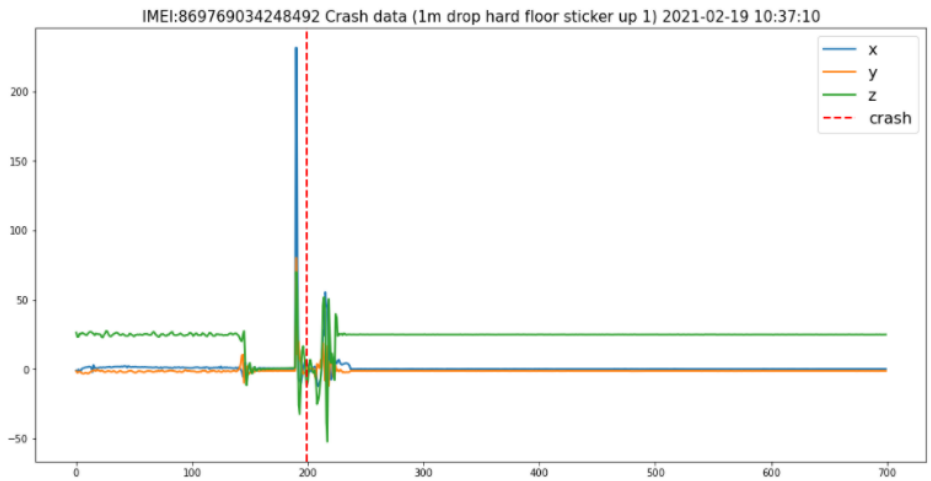
Met de kennis van de huidige situatie, het probleem en de requirements voor de oplossing is gestart met het onderzoek richting een oplossing. Hierbij is als eerste stap data-analyse uitgevoerd op de beschikbare data zodat kon worden geïnventariseerd welke gegevens relevant kunnen zijn, wat die data precies betekent en of deze van voldoende kwaliteit is.

Data-analyse

De volledige documentatie van deze data understanding is terug te vinden in bijlagedocument 2.4. De data understanding is uitgevoerd door gebruik te maken van de bestaande documentatie binnen Conneqtech en data aangeleverd door developers. Hierbij is inzicht verkregen in de werking van de tracker in hoe potentiële crashes gedetecteerd worden en welke informatie hierbij gebruikt kan worden voor verdere validatie van de crash.

Na de initiële data understanding is er nog een verdiepende verkenning (bijlagedocument 2.5) uitgevoerd op de crash data reports die een belangrijke rol zullen gaan spelen binnen de oplossing. Hierbij is op basis van een testplan (Zie Bijlage 2: Tracker Crash Data test plan) crashdata verzameld met een tracker waarmee het gedrag en de data geanalyseerd konden worden.

Onderstaande Figuur 6 laat een visualisatie van de crash data zien, met hierbij accelerometer data van een paar seconden vóór en na het crash moment.



Figuur 6: Voorbeeld visualisatie crash data

Naast validatie op basis van visualisaties zijn er verschillende tests uitgevoerd om de kwaliteit van de data te bepalen. Initieel kwam hierbij naar voren dat er soms data mist binnen de crashdata reports. Na communicatie met een developer bleek dit een simpel configuratie probleem die gelijk kon worden verholpen. Na implementatie van deze oplossing waren er geen reports meer met missende waarden.

Met het analyseren van de resultaten uit de test is geconcludeerd dat de tracker zich naar verwachting gedraagt en dat de crashdata van voldoende kwaliteit is, en daarmee bruikbaar als onderdeel van de oplossing.

Bestaande oplossingen

Om inspiratie op te doen en te zorgen dat het wiel niet opnieuw wordt uitgevonden is literatuuronderzoek gedaan naar bestaande oplossingen met betrekking tot crash/ongeval detectie. Hierbij is gezocht naar oplossingen binnen de auto-industrie, (motor)fietsenbranche en technieken die gebruik maken van accelerometer data.

Relevante gevonden oplossingen zijn gedocumenteerd op Confluence (Zie bijlagedocument 3.2.4)

Geen van de gevonden bestaande oplossingen was direct bruikbaar als oplossing voor het eigen probleem. Oplossingen binnen de auto-industrie maken gebruik van sensoren en gegevens welke niet beschikbaar zijn op e-bikes. Daarnaast maken veel crashdetectie systemen gebruik van validatie bij de gebruiker waarbij deze bijvoorbeeld 60 seconden de tijd krijgt om via de smartphone aan te geven dat hij/zij ok is. Voor dit onderzoek is al vastgesteld dat dit geen mogelijkheid is omdat niet kan worden gegarandeerd dat de gebruiker de smartphone meebrengt.

Uit het onderzoek naar bestaande oplossingen komt wel naar voren dat voor het valideren van crashes vaak gebruik gemaakt wordt van accelerometer data. Wat bevestigd dat de crash data reports met deze accelerometer data die de tracker kan versturen belangrijk kunnen zijn als onderdeel van een oplossing.

Kandidaat oplossingen

Op basis van inzichten verkregen uit de analyse van de huidige situatie, de requirements en de bestaande oplossingen zijn twee kandidaat oplossingen naar voren gekomen:

1. Het ontwikkelen van een applicatie met een algoritme dat gebruik maakt van de crash data reports en andere beschikbare informatie welke crash events kan evalueren.
2. Het ontwikkelen van een machine learning model welke op basis van crash data kan bepalen of het wel of niet om een echte crash gaat.

De kandidaten met voor- en nadelen zijn gedocumenteerd op Confluence (bijlagedocument 3.2)

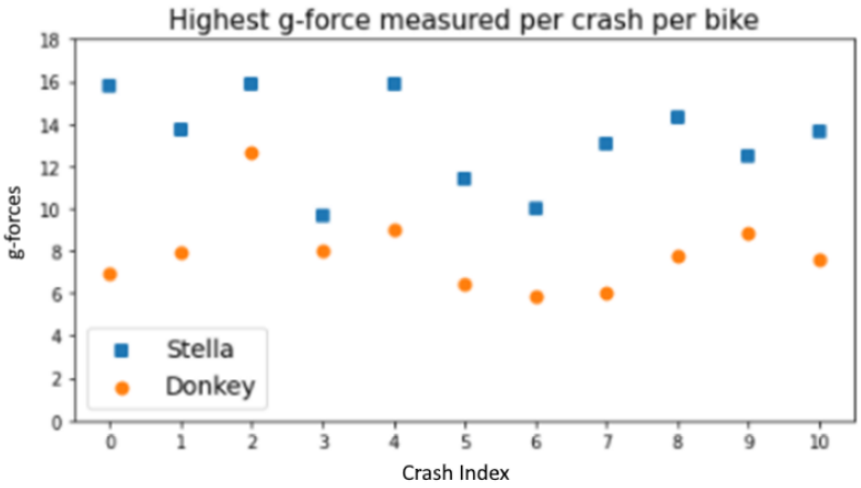
Voor de aanpak was het plan om een multi-criteria analyse toe te passen en zo het aantal kandidaat oplossingen terug te brengen tot twee of drie. Aangezien er maar twee oplossingen naar voren zijn gekomen is de multi-criteria analyse daarom overgeslagen. De twee kandidaten zijn voorgelegd aan de product owner en het services team die goedkeuring hebben gegeven om voor beide kandidaten een prototype te maken en daarmee de beste oplossing te bepalen. Voor het maken van een prototype applicatie en algoritme werd aangeraden dit te doen binnen de bestaande infrastructuur in programmeertaal Go.

Tracker positie en oriëntatie

Binnen de kandidaat oplossingen zijn er nog problemen die een oplossing nodig hadden. Een probleem wat relevant was voor beide kandidaten is dat de tracker positie en oriëntatie verschilt per fiets, wat invloed heeft op de crash data. Om te bepalen hoe dit impact heeft op de data is onderzoek uitgevoerd waarbij data van twee verschillende fietsen is vergeleken.

De documentatie van dit onderzoek is terug te vinden in bijlagedocument 3.3.

Het resultaat van dit onderzoek was constatering dat de gemeten piek in g-krachten meer dan 50% kan verschillen in bijna identieke scenario's. Daarnaast is vastgesteld dat variatie in de tracker oriëntatie betekent dat er niet vanuit gegaan kan worden dat bijvoorbeeld de Y-as van de accelerometer data altijd naar voren is gericht. Onderstaande Figuur 7 laat een visualisatie zien met de verschillen in gemeten piek g-krachten van twee verschillende fietsen in vergelijkbare situaties.



Figuur 7: Vergelijking twee fietsen in gemeten piek g-krachten

4.4 Prototypes

Om een keuze te maken uit de twee kandidaat oplossingen is voor beide een prototype gemaakt welke getoetst kon worden op prestaties en aansluiting op de requirements. Om te zorgen dat de prototypes getest en de prestaties geëvalueerd konden worden, was het nodig om zelf crash data te verzamelen door ongelukken te simuleren.

Data verzameling

Om te zorgen dat de prototypes getest en vergeleken konden worden was er data nodig van crashes waarbij bekend was of dit wel of niet als echte crash geclassificeerd zou moeten worden. Deze data bestond nog niet en daarom moest deze gecreëerd worden.

Om bruikbare test data te genereren is eerst een plan geschreven met hierin stappen om voor verschillende scenario's crash events te simuleren. Dit plan is terug te vinden in bijlagedocument 3.1. Voor de uitvoering van dit plan is met goedkeuring van de product owner een testfiets aangeschaft welke bij het simuleren van serieuze crashes dienst deed als substituut voor de dure elektrische fietsen.

Voor het verzamelen van de crashdata is gebruik gemaakt van een Python script welke de data ophaalt uit de database via de IoT API en deze vervolgens opslaat in JSON-formaat. Met het verzamelen van de data was er nu geclassificeerde crash data beschikbaar die gebruikt kon worden voor het testen, evalueren en vergelijken van de prototypes.

Machine learning prototype

Voor het maken van een machine learning prototype is gebruik gemaakt van Python en Jupyter Notebook. Hier was al ervaring mee en al toegepast voor het verzamelen van de data waardoor het eenvoudig was om deze data te gebruiken in machine learning.

Documentatie van de gehele opzet van het machine learning prototype is te vinden in bijlagedocument 3.5.1.

Voor het machine learning prototype is gebruik gemaakt van de verzamelde crash data. Hierbij is als eerste stap een standaard voorbereidingsproces uitgevoerd waarbij de data wordt aangepast en gefilterd zodat de data als input voor het train- en testproces gebruikt kan worden. Gekozen is om een eerste test uit te voeren met een Random Forest Classifier model omdat deze out-of-the-box goed presteert zonder dat er intensievere voorbereiding nodig is zoals het schalen of normaliseren van waarden. Daarnaast is het voordeel van een model gebaseerd op een beslisboom dat deze gevisualiseerd kan worden en daarmee inzicht in hoe beslissingen worden gemaakt.

Om de prestaties van het model te meten is gebruik gemaakt van een confusion matrix en een classification report. De confusion matrix (Figuur 8) vergelijkt de voorspellingen van het model met de verwachtte (Actual) waarde. Hierbij is “1” een echte crash en “0” een false positive. Uit deze confusion matrix is af te lezen dat het model 30 keer een correcte voorspelling doet en 5 keer een fout maakt.

	Actual 0	Actual 1
Predicted 0	24	3
Predicted 1	2	6

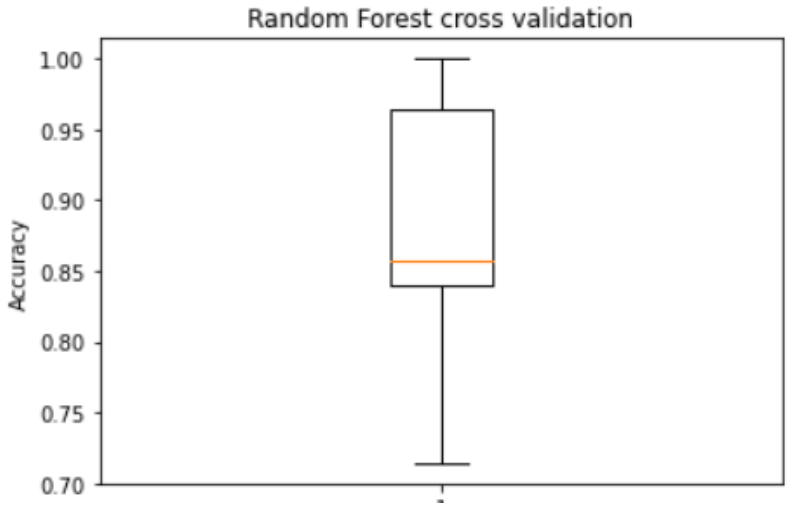
Figuur 8: Confusion matrix Random Forest Classifier

Het classification report (Figuur 9) toont de resultaten van het model met verschillende statistische gegevens. Belangrijk hierbij is de accuracy van 0.857, wat betekent dat 85,7% van de voorspellingen correct waren. Daarnaast is de recall score voor “1” (ook wel de true positive rate) belangrijk, omdat dit aangeeft dat ‘slechts’ 0.667 of 66,7% van de echte crashes correct geclassificeerd werden.

	precision	recall	f1-score	support
0	0.8889	0.9231	0.9057	26
1	0.7500	0.6667	0.7059	9
accuracy			0.8571	35
macro avg	0.8194	0.7949	0.8058	35
weighted avg	0.8532	0.8571	0.8543	35

Figuur 9: Classification report Random Forest Classifier

Omdat er gebruik gemaakt is van een willekeurige splitsing in train en test data is er een mogelijk probleem dat de resultaten erg afhankelijk zijn van welke data er in de train of test set terecht komt. Om dit te controleren is gebruik gemaakt van Cross validation (scikit-learn, z.d.) waarmee de resultaten van 10 verschillende train-test splits zijn vergeleken. Figuur 10 toont een boxplot welke de verdeling van de gemeten accuracy waarden laat zien en hiermee hoe de verschillende splits invloed hebben op de accuracy van het model. Hierbij varieert het resultaat van een perfect model met 100% accuracy tot een model wat 70% van de crashes correct kon classificeren.



Figuur 10: Cross validation boxplot

Ter validatie, om te voorkomen dat alle resultaten gebaseerd werden op een enkel model wat mogelijk niet goed presteert, is het train- en evaluatieproces herhaald met drie andere machine learning modellen. De uitwerking hiervan is terug te vinden in bijlagedocument 3.5.1.4. Deze modellen presteerden vergelijkbaar of iets minder dan het Random Forest model.

Als tweede test is gebruik gemaakt van een data normalisatieproces waarbij de 'magnitude', ook wel de lengte of kracht van de vectors, zijn gebruikt als input voor het trainproces. Hierbij is het Basic threshold monitoring algoritme (zie hoofdstuk 2: Algoritmes voor crashdetectie) toegepast.

Figuur 11 laat de resultaten van deze test zien in een confusion matrix en classification report. Het model heeft een accuracy van 74%, maar een true positive rate van slechts 11% waarbij negen van de tien echte crashes niet correct geclassificeerd worden.

	Actual 0	Actual 1
Predicted 0	25	8
Predicted 1	1	1

	precision	recall	f1-score	support
0	0.7576	0.9615	0.8475	26
1	0.5000	0.1111	0.1818	9
accuracy			0.7429	35
macro avg	0.6288	0.5363	0.5146	35
weighted avg	0.6913	0.7429	0.6763	35

Figuur 11: Confusion matrix & classification report van machine learning test met magnitudes

Bij het evalueren van de machine learning resultaten kwam een mogelijk probleem naar voren welke ontstond omdat alle data van echte crashes is gegenereerd met dezelfde testfiets. Mogelijk probleem hierbij is dat de machine learning modellen een bias (voorkeur) creëren voor de oriëntatie van de tracker op deze testfiets, welke anders is dan gebruikte fietsen voor het genereren van false positives. Om te zorgen dat het model niet de testfiets gaat herkennen op basis van de vectordata is een test uitgevoerd waarbij de labels van de vector assen door elkaar zijn gehaald. Bijkomend voordeel is dat door meerdere combinaties met verwisselde as-labels te gebruiken er meer data gegenereerd is om mee te trainen en testen.

Figuur 12 laat de resultaten van deze test zien in een confusion matrix en classification report. Het model heeft een accuracy van 92,75% en een true positive rate van 78,4%. Voor dit model is ook een cross validation uitgevoerd, hierbij laten de prestaties een spreiding in accuracy zien tussen 90% en 98%.

	Actual 0	Actual 1
Predicted 0	152	11
Predicted 1	4	40

	precision	recall	f1-score	support
0	0.9325	0.9744	0.9530	156
1	0.9091	0.7843	0.8421	51
accuracy			0.9275	207
macro avg	0.9208	0.8793	0.8975	207
weighted avg	0.9267	0.9275	0.9257	207

Figuur 12: Confusion matrix & classification report van machine learning test met verwisselde vector as-labels

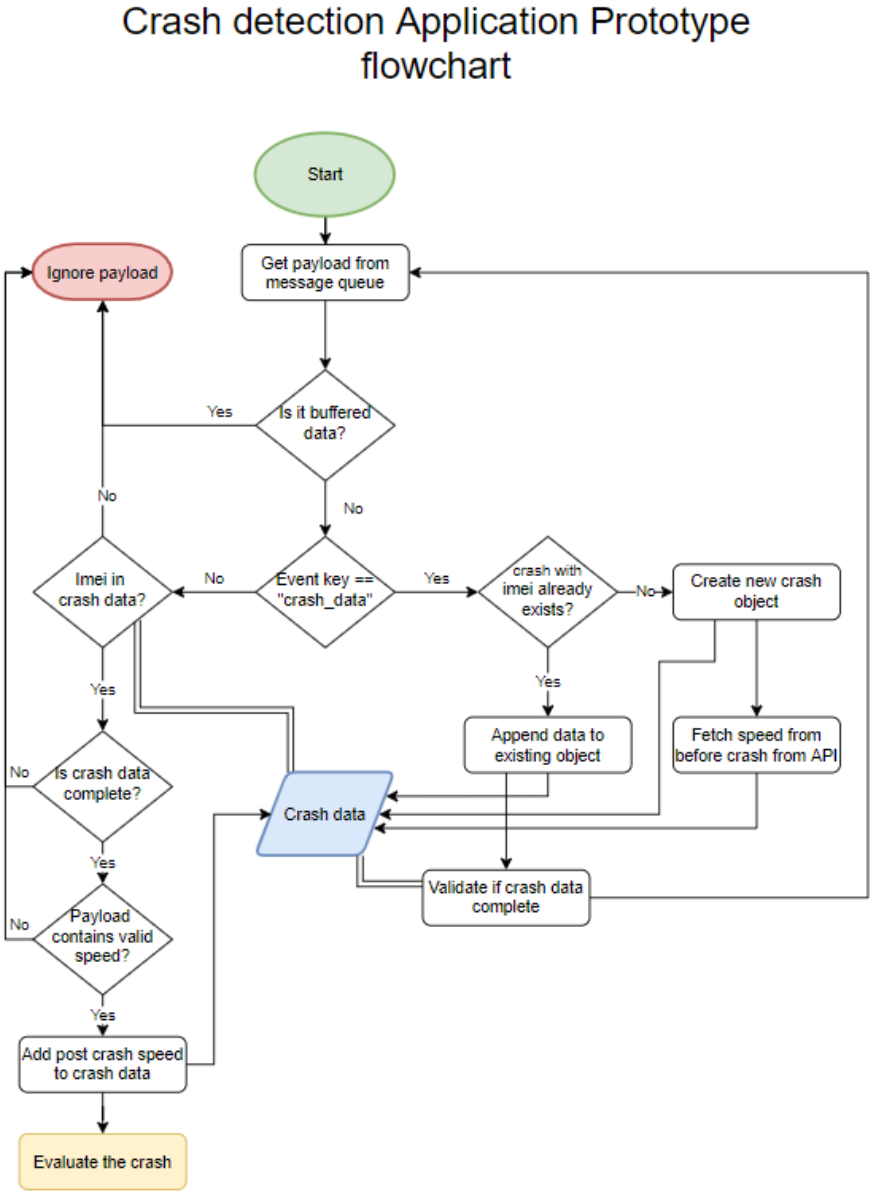
Kanttekening bij deze resultaten is wel dat met het door elkaar halen van de as-labels weliswaar geen kans meer is op bias voor een bepaalde oriëntatie, maar dat dit mogelijk ook zorgt dat het model minder betrouwbaar en representatief is voor alle e-bikes met verschillende tracker oriëntaties.

Crash detectie applicatie in Go

Voor het ontwikkelen van een crash detectie prototype in Go is eerst een werkomgeving opgezet (zie 4.5 Werkomgeving). Hierbij is met behulp van een senior developer een opzet van een Go applicatie gemaakt binnen de bestaande infrastructuur welke in staat was tracker data op te halen en deze te verwerken.

Na meerdere tests en experimenten is er een test applicatie gemaakt welke in staat was crash data te ontvangen en deze te evalueren en hierbij een voorspelling te doen of het wel of niet om een echte crash lijkt te gaan. De volledige opzet en evaluatie is beschreven op Confluence (Bijlagedocument 3.5.2)

Figuur 13 laat een flowchart zien met de opzet van het prototype en de manier waarop data wordt verwerkt.



Figuur 13: Flowchart crash detectie applicatie prototype

Voor het crash evaluatieproces zijn verschillende experimenten uitgevoerd. Hierbij is voor het prototype het volgende evaluatieproces gebruikt:

- De snelheid vóór de crash moet hoger dan 0 zijn.
- De snelheid minimaal 5 seconden na de crash moet 0 zijn.
- De fiets moet op zijn kant eindigen.

Hoewel dit evaluatieproces niet waterdicht is en in bepaalde situaties voor false positives of false negatives kan zorgen was dit voldoende om een uitspraak te kunnen doen over de prestaties van het prototype.

Om de prestaties te testen is gebruik gemaakt van de verzamelde crash data. Dit is dezelfde data als gebruikt voor het machine learning prototype, zodat de resultaten objectief vergeleken kunnen worden. Voor de tests is een script geschreven welke alle crashes door het prototype laat evalueren, waarna de voorspellingen zijn vergeleken met de gewenste classificatie door de waarden voor een confusion matrix te berekenen. Figuur 14 toont de confusion matrix, uit deze resultaten van de test is berekend dat het prototype een accuracy van bijna 90% behaald, en een true positive rate van 76%.

	Actual 0	**Actual 1**
Predicted 0	49	4
Predicted 1	3	13

Figuur 14: Confusion matrix resultaten applicatie prototype

Keuze voor oplossing

Uit de resultaten van het machine learning prototype is gebleken dat dit primair door een gebrek aan geclassificeerde crash data geen haalbare oplossing is. Zelfs met een trucje om meer data te genereren kwam het machine learning model niet verder dan een 78% true positive rate, wat betekent dat het bijna een kwart van de echte crashes niet correct zou herkennen.

Op basis van een toetsing aan de requirements (zie bijlagedocument 3.5.1.7 en 3.5.2.6) komt de Go applicatie beter naar voren. Hoewel het prototype op zichzelf nog niet betrouwbaar genoeg is met een true positive rate van 76%, toont het wel veel potentie en was duidelijk wat er gedaan moest worden om een applicatie te realiseren welke voldoende presteert en een daadwerkelijke oplossing zou worden voor het probleem.

De keuze voor een oplossing is als advies gepresenteerd tijdens een services meeting en daarbij voorgelegd aan de product owner. (Zie bijlage notulen 19.)

4.5 Werkomgeving

Om de prototypes en de gekozen oplossing te realiseren was er een werkomgeving nodig waarbij code geschreven, getest en beheerd kon worden op een manier dat deze leesbaar, deelbaar en functioneel was.

Conneqtech maakt gebruik van Atlassian software waaronder Bitbucket, Jira en Confluence. Om de Go code deelbaar en beheersbaar te houden is gebruik gemaakt van versiebeheer met Bitbucket. Hierbij is het branching model gebruikt zoals Conneqtech deze hanteert. Het prototype en de verdere ontwikkeling zijn hierbij uitgevoerd op zogenaamde feature branches, waarbij de aanpassingen en toevoegingen middels een pull request samengevoegd kunnen worden met de hoofdbranch.

Op aanraden van de ontwikkelaars is voor het programmeren met Go gebruik gemaakt van GoLand, een IDE (integrated development environment) van Jetbrains specifiek voor programmeertaal Go.

Om te kwaliteit van de code te garanderen is ten eerste gebruik gemaakt van meerdere codereview momenten met een senior ontwikkelaar waarbij de opzet is besproken en eventuele problemen zijn verholpen (zie bijlage notulen 16, 17, 18, 21).

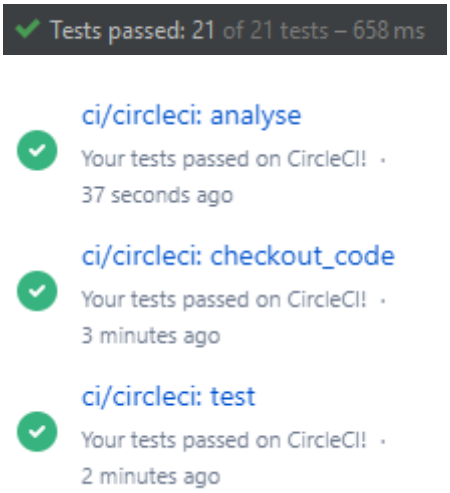
Naast de codereviews is in de requirements opgenomen dat de code voorzien moet zijn van unit tests voor de gehele applicatie. Hiervoor is gebruik gemaakt van de Testing package in Go, waarmee code kan worden uitgevoerd en gecontroleerd of de output van functies naar verwachting is. Om de kwaliteit van de code te testen zijn meerdere unit tests geschreven die controleren of de applicatie zich naar verwachting gedraagt en de crash data op een correcte manier verwerkt.

Figuur 15 laat een voorbeeld zien van een test waarbij een berekening wordt uitgevoerd en middels een test.Assert functie wordt gecontroleerd of de output correct is.

```
func Test_vectorMagnitude(t *testing.T) {
    vectorA := data.CrashVector{
        X: 0.745439,
        Y: 0.065434,
        Z: 0.674523,
    }

    mag := vectorMagnitude(vectorA)

    test.Assert(t,  name: "magnitude", mag,  expectation: 1.0074433844399964)

}
```

Figuur 15: Voorbeeld unit test in Go

De tests kunnen allemaal worden uitgevoerd met een positief resultaat. Na het maken van een pull request wordt de code gecontroleerd door CircleCi. Figuur 16 laat zien dat alle tests slagen. Dit laat zien dat de code in ieder geval geen grove functionele fouten bevat. De pull requests zijn ook gecontroleerd door minimaal twee andere ontwikkelaars die goedkeuring moeten verlenen, dit zorgt dat de kwaliteit van de code gewaarborgd is.



Figuur 16: Succesvolle tests

Python

Voor het toepassen van machine learning en het verkennen van de data, zoals het uitvoeren van tests en het maken van visualisaties, is gebruik gemaakt van programmeertaal Python en Jupyter Notebook. Hiermee kunnen eenvoudig blokken code uitgevoerd worden en de output gecontroleerd.

Resultaten en inzichten uit de data-analyse met Python zijn gedocumenteerd op Confluence (Bijlagedocument 2.5 en 3.4).

4.6 Iteratief ontwerpen

Het originele plan was om na het kiezen van een oplossing een ontwerp te maken, welke dan gebruikt kon worden voor de implementatie. Hier is gekozen om van het plan af te wijken en een meer iteratief verbeterproces toe te passen. Dit sluit aan bij de werkwijze van Conneqtech waarbij wordt gewerkt met Scrum en sprints van twee weken. Ook is na het evalueren van de prototypes geconcludeerd dat de prototype Go applicatie op zichzelf al een goede opzet heeft waar verder op gebouwd kan worden en verbeteringen toegevoegd kunnen worden door specifieke problemen te onderzoeken.

Met het iteratieve proces zijn er meerdere verbeterstappen gerealiseerd waarbij het prototype is uitgebouwd tot een volwaardige oplossing. Elke iteratie zijn er code reviews en tests uitgevoerd om de prestaties en kwaliteit van de applicatie te controleren.

De belangrijkste verbeterstappen en onderzoek tijdens het iteratieve verbeterproces waren:

- Het herkennen van patronen in de vectordata zodat kan worden bepaald of de fiets stilstaat, normaal beweegt of zich in een chaotische situatie bevindt.
- Het herkennen van beweging vóór en na de crash. Zodat kan worden bepaald of er een normale rit bezig was en of de fiets wel of niet verder rijdt na een vermeende crash.
- Het bepalen van de oriëntatie van de fiets op basis van de vectordata. Hiermee kan worden bepaald of de fiets op zijn kant ligt na de vermeende crash.

Herkennen van patronen in de crash data

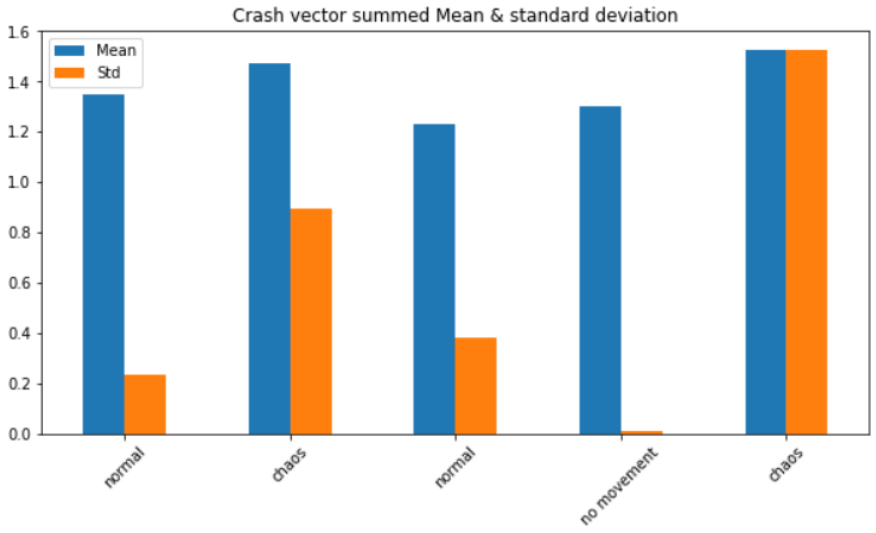
Uit analyse van de prestaties van de prototype Go applicatie kwam naar voren dat de classificatie meerdere malen fout ging omdat het evaluatieproces geen rekening hield met de context van de situatie. Hierbij ontstonden er problemen met het bepalen van de oriëntatie als de vectordata chaotische patronen liet zien die suggereerden dat de fiets op dat moment een crash ervaart. De oriëntatie berekening was dan niet betrouwbaar en daarom is besloten te onderzoeken of en hoe deze patronen in de vectordata herkent kunnen worden zodat fouten kunnen worden voorkomen.

Voor dit onderzoek is vectordata verzameld van verschillende scenario’s vóór en na de crash en zijn deze vergeleken met visualisaties en wiskundige berekeningen. Vervolgens is gezocht naar de beste manier om deze scenario’s van elkaar te onderscheiden.

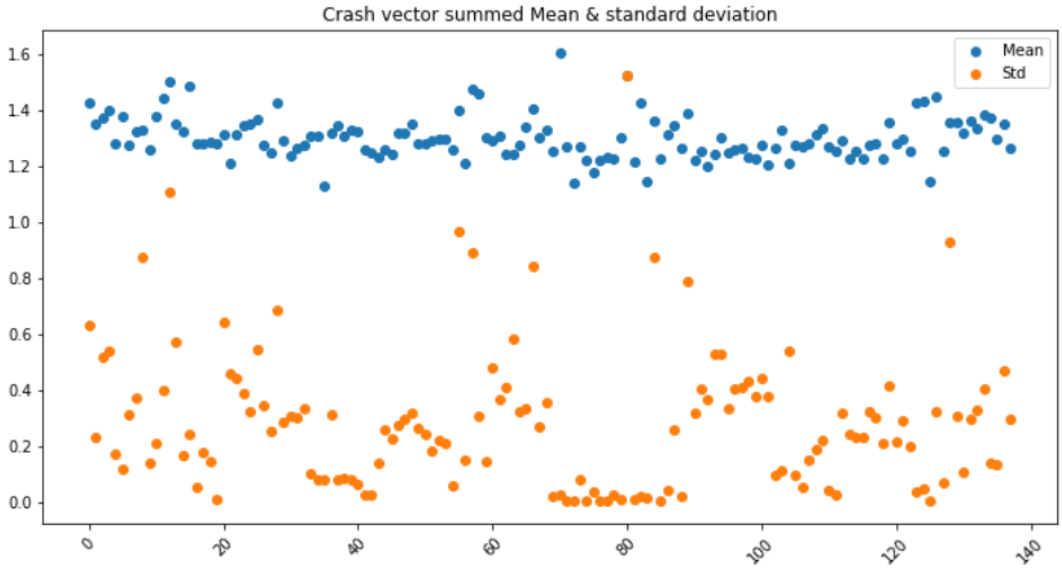
De volledige uitwerking van het onderzoek is te vinden op Confluence (Bijlagedocument 4.2)

Uit het onderzoek kwam naar voren dat de standaarddeviatie van de vectordata een goede graadmeter is om de situatie te bepalen. Hierbij kan onderscheid gemaakt worden tussen stilstaan, normaal rijgedrag en chaotische situaties. Figuur 17 laat een berekend gemiddelde en standaarddeviatie zien voor verschillende scenario’s.

Om deze logica te toetsen is de berekening toegepast op de verzamelde crashdata (Figuur 18), waarna is geëvalueerd welke situaties een hoge standaarddeviatie hebben. Hiermee is bepaald dat een standaarddeviatie boven 0.8 altijd situaties waarbij de fiets geen normaal rijgedrag vertoont. Een standaarddeviatie onder 0.1 betekend dat de fiets niet in (noemenswaardige) beweging is. Waarden tussen 0.1 en 0.8 suggereren een situatie van normaal rijgedrag.



Figuur 17: Vergelijking gemiddelde en standaarddeviatie verschillende scenario's



Figuur 18: Vergelijking standaarddeviatie van test data

Herkennen van beweging vóór en na de crash

De vectordata uit de crash data reports zijn heel waardevol gebleken om een crash mee te kunnen classificeren. Echter is deze data beperkt tot twee seconden vóór en vijf seconden na de crash. De situatie buiten deze zeven seconden is wat betreft de vectordata onbekend, terwijl deze wel belangrijk kan zijn om de context van de crash te bepalen. Twee belangrijke metrics hierbij zijn:

- Was er een normale rit aan de gang vóór de vermeende crash?
  Als er geen beweging was vóór de crash suggereert dit dat niemand op de fiets aan het rijden was en de crash is veroorzaakt omdat deze bijvoorbeeld omvalt of omdat iemand tegen de geparkeerde fiets aanstoot.
- Blijft de fiets normaal doorrijden na een vermeende crash?
  Als de fiets gewoon blijft doorrijden na een crash event is het eenvoudig deze te classificeren als false positive, waarschijnlijk veroorzaakt door het raken van een put of ander object. Of de gebruiker herpakt zich direct na een lichte val.

Om uit te vinden hoe beweging vóór en na de crash kan worden bepaald en gebruikt in het classificatieproces is een onderzoek uitgevoerd. De volledige uitwerking van dit onderzoek is te vinden op Confluence (Bijlagedocument 4.3).

Het resultaat van dit onderzoek is een oplossing waarbij beweging vóór de crash wordt bepaald door te kijken naar het laatste ‘rest_to_motion’ event, wat kan worden vergeleken met de tijd van de crash om te bepalen hoe lang de fiets al in beweging was.

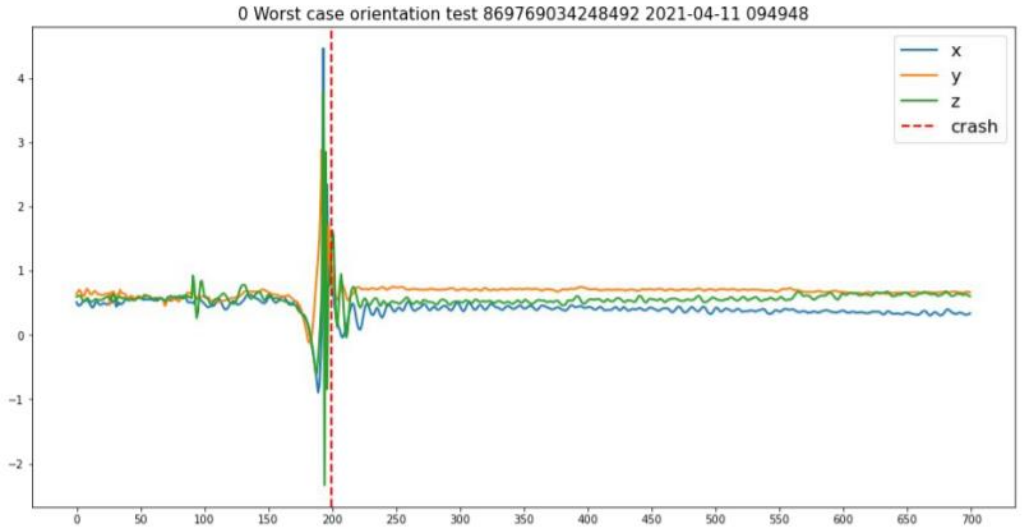
Om beweging na de crash te bepalen wordt gebruik gemaakt van de gemeten snelheid. Hierbij wacht de applicatie tot 30 seconden na het crash moment waarbij voor nieuwe data wordt gekeken of hier een valide snelheid in wordt gerapporteerd. Is een snelheid boven de 3km/u wordt gemeten tussen 10 en 30 seconden na de crash wordt dit in het crash evaluatieproces meegenomen als indicatie dat de gebruiker gewoon doorfietst en het niet om een echte crash gaat.

Het bepalen van de oriëntatie

Een belangrijke metric om de crashdata mee te classificeren is om te bepalen of de fiets op zijn kant ligt na de crash. De vectordata met acceleratiegegevens geeft een indicatie van de oriëntatie van de fiets. Echter verschilt de oriëntatie van de tracker per fiets, waarbij deze in alle mogelijke posities gemonteerd kan zijn. Voor het prototype was al een berekening geïmplementeerd, maar deze hield geen rekening met alle mogelijke tracker oriëntaties waardoor deze niet altijd betrouwbaar was om te bepalen of een fiets op zijn kant ligt.

Om te bepalen hoe de oriëntatie van de fiets correct bepaald kan worden ongeacht hoe de tracker is gemonteerd is een onderzoek uitgevoerd. De volledige uitwerking van dit onderzoek is te vinden op Confluence (Bijlagedocument 4.4).

Voor dit onderzoek is uitgegaan van een ‘worst case scenario’ waarbij de tracker is georiënteerd op een manier waarbij de zwaartekracht evenredig is verdeeld over alle drie de assen. In deze situatie er geen mogelijkheid om te oriëntatie te bepalen op basis van de richting van de zwaartekracht. Figuur 19 toont hoe de vector data er uit kan zien in dit scenario waarbij er geen onderscheid tussen de assen gemaakt kan worden.



Figuur 19: Voorbeeld visualisatie worst case oriëntatie scenario test

Om een oplossing te vinden die bruikbaar was met alle mogelijke tracker oriëntaties is literatuuronderzoek verricht en zijn verschillende experimenten uitgevoerd. Hierbij is een oplossing gevonden die de hoek kan berekenen tussen twee vectoren. Door gebruik te maken van een gemiddelde vector van vóór en na de crash en voor deze de hoek te berekenen is er een bruikbare oplossing gevonden waarbij de verandering in aantal graden kan worden geëvalueerd.

Het berekenen van de verandering in aantal graden werkt het best in combinatie met de berekening voor de standaarddeviatie. Hierbij is bepaald dat een hoge standaarddeviatie (>0.8) zorgt dat de hoekberekening niet betrouwbaar is. Door dit mee te nemen in het crash evaluatieproces wordt ervoor gezorgd dat de berekende hoek alleen wordt gebruikt als de standaarddeviatie binnen de grenswaarden valt en daarmee betrouwbaar.

Resultaat

Na meerdere iteraties aan verbeteringen waarbij de applicatie is geëvalueerd en het evaluatieproces geoptimaliseerd, haalt de applicatie een accuracy van 95.7% en een true positive rate van 94.1%. Figuur 20 laat de confusion matrix zien waar deze resultaten op zijn gebaseerd. Kanttekening hierbij is dat de fouten die de applicatie maakte deels te verwijten waren aan mogelijk onrealistische situaties in de test data, welke ontstaan zijn door beperkingen aan het kunnen simuleren van de verschillende crash scenario’s. Na implementatie is het dan ook belangrijk dat de prestaties van de applicatie worden geëvalueerd op crashdata anders dan de gesimuleerde crashes.

	Actual 0	**Actual 1**
Predicted 0	50	1
Predicted 1	2	16

Figuur 20: Confusion matrix resultaten applicatie na verbeterproces

4.7 Evaluatie

Na implementatie was de taak om te evalueren hoe de oplossing presteert en in hoeverre deze voldoet aan de gestelde requirements.

Evaluatie werking

Om de geïmplementeerde applicatie te evalueren is een testplan opgesteld met hierin een beschrijving en vragen voor verschillende aspecten van de applicatie die getest kunnen worden. Met het beantwoorden van de vragen kan een uitspraak gedaan worden over de kwaliteit van de software. Het testplan is terug te vinden in Bijlagedocument 4.5.

Voor het uitvoeren van het testplan is de applicatie uitgerold op de staging (test) ontwikkelomgeving. De volledige uitwerking van de testresultaten is terug te vinden in Bijlagedocument 4.5.5. Om het systeem te testen is ervoor gezorgd dat 694 fietsen een ingeschakelde crashdetectie hadden en de data rapporteerden naar de staging omgeving.

Voor het analyseren van de resultaten is gebruik gemaakt van een periode van 24 uur aan data. De belangrijkste resultaten en conclusies uit de analyse van deze data:

De standaarddeviatie tijdens normaal rijgedrag kan hoger zijn dan gedacht. De huidige implementatie gebruikt een grenswaarde van 0.8 waarboven de situatie als ‘chaotisch’ wordt gezien. Echter blijkt uit de data dat sommige fietsen vóór en na bijvoorbeeld over een put te rijden er een standaarddeviatie kan zijn van 1.5. Om te zorgen dat de crashdetectie niet te snel situaties als chaotisch gaat bestempelen is het nodig om de grenswaarde te verhogen. Een waarde van 1.5 lijkt hier een goede grens om onderscheid te maken tussen normaal rijgedrag en abnormale situaties.

Er zit gemiddeld 24 seconden tussen het ontvangen van de eerste crashdata payload en het publishen van het crashdetection event, met een spreiding van tussen de 17 en 35 seconden. Hiermee is de applicatie sneller dan de oude crashdetectie die minimaal 60 seconden nodig heeft.

Het aantal crashdetection events is met de huidige gevoeligheid erg hoog. Uit de data blijkt dat meer dan 90% van de crash events wordt veroorzaakt door duidelijke false positives waarbij de fiets niet omvalt en gewoon blijft doorrijden. Met de huidige gevoeligheid wordt er onnodig veel data verstuurd, om dit te reduceren kan de gevoeligheid van de tracker aangepast worden zodat g-krachten hoger moeten zijn voordat er een crash wordt geregistreerd.

A/B sensitivity test

Om te bepalen wat een goede sensitivity is voor de crashdetectie is een A/B test (ictresearchmethods.nl, 2018b) opgezet, waarbij een deel van de apparaten een andere sensitivity heeft. Door de twee groepen te vergelijken kan een uitspraak gedaan worden over het effect van een aangepaste gevoeligheid.

Het resultaat van deze test laat zien dat een aanpassing van de sensitivity van 25 naar 50 (hoger is minder gevoelig) zorgt voor een reductie in crashdetection events van ongeveer 90%. Op basis van de testdata is geconcludeerd dat de echte crashes in vrijwel alle gevallen een hoge piek in g-krachten hebben, waardoor de verwachting is dat de lagere gevoeligheid vrijwel alleen non-crashes zal wegfilteren. Een sensitivity van 50 lijkt hier een goede balans in dataverbruik en kans dat echte crashes genegeerd worden.

Requirements evaluatie

Requirements evaluatie gebaseerd op testresultaten en validatie met een senior ontwikkelaar (Zie bijlage notulen 32).

Tabel 1: Requirements evaluatie

	Titel	Prioriteit	Evaluatie
1	Crash detection	MUST	Als oplossing voor het probleem is er een applicatie gerealiseerd welke op basis van de crash data kan bepalen of het om een valide crash gaat. Op basis van de testresultaten is gebleken dat de applicatie ongeveer 95% accuraat is. De oplossing voldoet aan alle must have requirements en de gestelde kwaliteitseisen.
2	Bike types	MUST	De applicatie werkt onafhankelijk van het type fiets en kan overweg met variatie in de data veroorzaakt door verschillende typen fietsen en tracker posities. Daarmee voldoet de oplossing aan deze requirement.
3	Tracker orientation	MUST	Het crash evaluatieproces werkt met elke mogelijke tracker oriëntatie. Daarmee voldoet de oplossing aan deze requirement.
4	Crash severity rating	SHOULD	De applicatie geeft een ‘probability’ waarde voor elke crash welke een indicatie geeft hoe serieus de crash lijkt te zijn.
5	Sensitivity by user	WON'T	Niet uitgevoerd.
6	Insights	COULD	Na evaluatie wordt een ‘crashdetection’ event aangemaakt met hierin metadata over de evaluatie zoals de gebruikte parameters en de berekende waarden. Met behulp van de metadata kan de crash ook gereproduceerd worden.
7	Cancelling emergency messages	WON'T	Niet uitgevoerd.
8	Unit tests	MUST	Coverage d.m.v. unit tests komt volgens Sonarcloud uit op 90.9%, Conneqtech hanteert een minimum coverage van 60%.
9	Performance	MUST	Analyse van de implementatie (Zie Bijlagedocument 4.5.5) laat zien dat er gemiddeld 24 seconden zit tussen het ontvangen van de eerste crashdata payload en het publishen van het crashdetection event.
10	Expandability	SHOULD	De uitbreidbaarheid is beschreven in het adviesdocument (bijlage). De applicatie heeft geen technische beperkingen die het onmogelijk zouden maken om nieuwe variabelen of andere hardware toe te voegen.
11	Scalability	SHOULD	De applicatie is geïmplementeerd binnen de event worker architectuur. Hierbij wordt automatisch load balancing toegepast waarbij het aantal workers kan schalen met de workload.
12	Data usage	SHOULD	Met een crash sensitivity van 25 was het aantal false positives en data verbruik erg hoog. Met een sensitivity van 50 is het dataverbruik acceptabel.

Hoofdstuk 5: Conclusie

Het doel van dit onderzoek was om een verbetering te realiseren van de crashdetectie waarbij Conneqtech voldoende vertrouwen heeft om deze functionaliteit aan te prijzen.

Uit de initiële opdrachtbeschrijving kwam al naar voren dat er geen vertrouwen was in de oude crashdetectie door het aantal fouten en het gebrek aan inzicht in de detectie. Uit verdere analyse van het probleem kwam hierbij naar boven dat het gebrek aan vertrouwen mede werd veroorzaakt omdat de opzet nooit objectief geëvalueerd was, er geen testdata beschikbaar was en daarom ook geen kwantitatieve resultaten bestonden over de prestaties.

Het resultaat van dit onderzoek is een gerealiseerde verbetering van de crashdetectie welke ongeveer 95% accuraat is in het correct classificeren van de crashes. Dit resultaat is bereikt na uitgebreid onderzoek naar mogelijke oplossingen door een combinatie van literatuurstudie, experimenten, tests en prototypes welke zijn gedocumenteerd en beschikbaar voor Conneqtech, zodat er inzicht is in de gemaakte keuzes en de werking van de oplossing. Dit moet Conneqtech het vertrouwen geven dat de gerealiseerde verbetering van de crashdetectie van voldoende kwaliteit is.

De oplossing is gebouwd op een manier waarbij er inzicht is in hoe de crashes zijn geëvalueerd met behulp van metadata. Door deze metadata terug te halen kan de crash gereproduceerd worden en kan worden ingezien met welke variabelen de crash geclassificeerd is. Dit moet niet alleen het vertrouwen in de oplossing verder vergroten maar geeft ook de mogelijkheid om de werking te evalueren en aanpassingen te maken aan het crashevaluatieproces, als dit op basis van de inzichten nodig wordt geacht. De applicatie werkt met een aantal thresholds (grenswaarden) zoals wanneer de ‘crash probability’ hoog genoeg is om dit als echte crash te classificeren. Deze waarden kunnen eenvoudig worden aangepast zodat het evaluatieproces gefinetuned kan worden en het aantal fouten geminimaliseerd.

De evaluatie heeft laten zien dat de applicatie werkt en in staat is om crashes correct te evalueren. In vergelijking met de oude crashdetectie is de kans op false positives kleiner en door de inzichtelijkheid kunnen aanpassingen gemaakt worden om het aantal fouten verder te reduceren. De requirements evaluatie liet ook zien dat de oplossing voldoet aan de gestelde eisen, waarmee geconcludeerd kan worden dat er een betrouwbare verbetering van de crashdetectie is gerealiseerd.

Hoofdstuk 6: Discussie

Bij de uitvoering van dit onderzoek is de initieel geschreven aanpak (Zie Hoofdstuk 3: Aanpak) grotendeels correct gevolgd. Op twee momenten is afgeweken van het plan, het eerste moment bij het bepalen van de mogelijke oplossingen. Hier was volgens de aanpak het plan om een multi-criteria analyse toe te passen en zo het aantal kandidaat oplossingen terug te brengen tot twee of drie. Er waren echter maar twee realistische oplossingen naar voren gekomen, waardoor het onnodig was om nog een multi-criteria analyse uit te voeren. Met behulp van prototyping kon toen een keuze gemaakt worden uit de twee kandidaten.

Het tweede moment waarbij is afgeweken van het initiële plan was bij het ontwerpen van een implementatie nadat er een oplossing was gekozen. Hier is gekozen om een meer iteratief verbeterproces toe te passen. Dit sloot aan bij de werkwijze van Conneqtech waarbij wordt gewerkt met Scrum en sprints van twee weken. Verder is geconcludeerd dat de prototype Go applicatie al een goede opzet had waar verder op gebouwd kon worden. Hier was al een ontwerp voor gemaakt welke ook bruikbaar was voor de realisatie van de oplossing.

Naast deze twee momenten is het plan correct opgevolgd. Hierbij is zoveel mogelijk gebruik gemaakt van de bestaande werkwijze en technieken vanuit Conneqtech zodat het eindproduct gerealiseerd en overgedragen kon worden op een manier die effectief was voor Conneqtech. Dit betreft niet alleen de gerealiseerde crashdetectie applicatie maar ook de achterliggende documentatie met onderbouwing voor gemaakte keuzes op de Conneqtech Confluence documentatie. Hiermee was documentatie van het onderzoek direct beschikbaar en deelbaar binnen de organisatie.

Beperkingen van de test crashdata

Bij de evaluatie van de prototypes en het gerealiseerde product is gebruik gemaakt van crashdata welke is gecreëerd door verschillende crash scenario's te simuleren. Het simuleren van deze scenario's had echter een aantal beperkingen, zo was het niet mogelijk om ernstige ongelukken zoals een aanrijding door een auto of een val met hoge snelheid te simuleren. Het risico op verwondingen en schade was hiervoor te groot. Door deze beperkingen aan beschikbare testdata zijn niet alle mogelijke scenario's objectief meegenomen in de evaluatie van de producten en is hiervoor meer gebruik gemaakt van gezond verstand om het crash evaluatieproces te optimaliseren.

Naast deze missende scenario's in de test data is er nog het probleem dat de verhoudingen binnen de testdata in het aantal wel/niet echte crashes en de scenario's niet representatief is voor de werkelijkheid. In werkelijkheid zal waarschijnlijk meer dan 90% van de crash events als non-crash geclassificeerd moeten worden, waarbij een aanzienlijk deel veroorzaakt zal worden door scenario's zoals stoepranden en geparkeerde fietsen die omvallen. De gerealiseerde oplossing zal met het leeuwendeel van deze data geen moeite hebben om deze correct te classificeren waardoor puur door de verhoudingen een accuraatheid van >99% kan worden behaald. Gevaar hierbij is dat snel de aanname wordt gedaan dat de applicatie 'goed' werkt, terwijl deze accuraatheid een verkeerd beeld zal geven in hoeverre de echte crashes correct geclassificeerd worden. Aanbeveling hierbij is dan ook om bij het evalueren van de prestaties verder te kijken dan alleen een totaal percentage correcte classificaties en ook te kijken naar bijvoorbeeld een 'false negative rate' welke een indicatie geeft over hoe vaak de applicatie echte ongelukken niet correct herkent. Dit laatste is belangrijk, want de echte crashes zullen relatief weinig voorvallen maar zijn wel cruciaal om correct te classificeren.

Toegevoegde waarde van het beroepsproduct

De gerealiseerde oplossing is een duidelijke verbetering van de crashdetectie. Deze verbetering is echter niet te kwantificeren, simpelweg omdat er geen kwantitatieve gegevens beschikbaar zijn over de prestaties van de oude crashdetectie. Dit op zichzelf is al een belangrijke indicatie dat de gerealiseerde oplossing betrouwbaarder is, want de gerealiseerde applicatie is gevalideerd zodat de prestaties kwantitatief benoemd kunnen worden. De oplossing is ongeveer 95% in het correct classificeren van crashes. Daarnaast voldoet de oplossing aan alle gestelde requirements zodat het totaalplaatje een betrouwbare oplossing is die aansluit bij de wensen van Conneqtech.

Voor Conneqtech betekent dit dat de crashdetectie kan worden aangeraden aan klanten welke dit weer kunnen gebruiken in de marketing als een feature van de elektrische fietsen. Met het verbeteren van de aangeboden producten kan Conneqtech financieel ook meer druk uitoefenen en zichzelf in een betere positie op de markt positioneren. Op zijn beurt kan dit weer nieuwe klanten met zich meebrengen zodat Conneqtech kan groeien.

Adviesrapport

Om een onderbouwd voorstel te geven voor een passend vervolgtraject is een adviesrapport geschreven. (Dit adviesrapport is een aparte bijlage). In dit adviesrapport is eerst beschreven wat de gerealiseerde oplossing is en waarom dit een verbetering is ten opzichte van de oude crashdetectie. Daarna is beschreven hoe de applicatie onderhouden kan worden, met advies om een dashboard met dataverzameling en monitoring op te zetten zodat inzicht verkregen kan worden in de werking en prestaties van de applicatie. Vervolgens is beschreven hoe de applicatie uitgebreid kan worden voor gebruik met nieuwe hardware en hoe de crashdetectie gebruikt zou kunnen worden voor andere typen fietsen of andere soorten vervoersmiddelen.

Voor het vervolgtraject is advies geschreven over hoe de prestaties geëvalueerd en verbeterd kunnen worden door een functionaliteit te bouwen waarmee feedback van de gebruiker kan worden gevraagd na een vermeende crash. Verder is in het adviesrapport beschreven dat er verder onderzoek gedaan kan worden naar de mogelijkheden om verschillende acties te gebruiken afhankelijk van de crash probability. Tot slot is er advies gegeven voor vervolgonderzoek naar de mogelijkheden om de crash evaluatie deels in de tracker hardware te implementeren zodat dataverbruik gereduceerd kan worden.

Literatuurlijst

3d Accelerometer calculate the orientation. (2010, 20 september). Stack Overflow. https://stackoverflow.com/questions/3755059/3d-accelerometer-calculate-the-orientation

AXA Bike Security. (z.d.). Connect. Geraadpleegd op 2 januari 2021, van https://www.axa-in.com/connect/

Candefjord, S., Sandsjö, L., Andersson, R., Carlborg, N., Szakal, A., & Westlund, J. (2014). Using Smartphones to Monitor Cycling and Automatically Detect Accidents - Towards eCall Functionality for Cyclists. Proceedings, International Cycling Safety Conference 2014, 1–9. http://publications.lib.chalmers.se/records/fulltext/211381/local_211381.pdf

Care – Conneqtech. (z.d.). Conneqtech. Geraadpleegd op 30 december 2020, van https://www.conneqtech.com/care/

Conneqtech | De Connected E-Bike. (z.d.). Geraadpleegd van https://www.conneqtech.com/nl/de-connected-e-bike/

Conneqtech | My GPS Tracker – NL. (z.d.). Geraadpleegd van https://www.conneqtech.com/nl/my-gps-tracker/

How to find the magnitude and direction of a 3D vector. (2018, 15 november). [Video]. YouTube. https://www.youtube.com/watch?v=MYuSkJhJlvw

ictresearchmethods.nl. (2018a). A/B testing - ICT research methods. https://ictresearchmethods.nl/A/B_testing

ictresearchmethods.nl. (2018b). Available product analysis - ICT research methods. https://ictresearchmethods.nl/Available_product_analysis

Khan, J. Y., & Yuce, M. R. (2019). Internet of Things (IoT). https://books.google.nl/books?id=DRGwDwAAQBAJ&pg=PA231&dq=ISBN+978-981-4800-29-7&hl=en&sa=X&ved=2ahUKEwi63M7NzonuAhWBH-wKHaYeDSoQ6AEwAHoECAYQAg#v=onepage&q=ISBN%20978-981-4800-29-7&f=false

Ministerie van Algemene Zaken. (2019, 6 juni). Welke regels gelden voor mijn elektrische fiets (e-bike)? Rijksoverheid.nl. https://www.rijksoverheid.nl/onderwerpen/fiets/vraag-en-antwoord/welke-regels-gelden-voor-mijn-elektrische-fiets-e-bike

MoSCoW – projectmanagementsite. (z.d.). Projectmanagementsite. Geraadpleegd op 31 december 2020, van https://projectmanagementsite.nl/moscow/

Pedley, M. (2013, maart). Tilt Sensing Using a Three-Axis Accelerometer (Nr. AN3461). https://www.nxp.com/docs/en/application-note/AN3461.pdf

Salmeron-Manzano, E., & Manzano-Agugliaro, F. (2018). The Electric Bicycle: Worldwide Research Trends. Energies, 11(7), 1894. https://doi.org/10.3390/en11071894

Santos, M. (2020, 6 juni). False Positives vs. False Negatives - Towards Data Science. Geraadpleegd op 2 januari 2021, van https://towardsdatascience.com/false-positives-vs-false-negatives-4184c2ff941a

scikit-learn. (z.d.). 3.1. Cross-validation: evaluating estimator performance — scikit-learn 0.24.2 documentation. Geraadpleegd op 6 april 2021, van https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation

TechTarget. (z.d.). Example of an IoT system [Illustratie]. Geraadpleegd van https://cdn.ttgtmedia.com/rms/onlineimages/iota-iot_system.png

TechTarget Contributors. (2020, 11 februari). Internet of things (IoT). Geraadpleegd op 1 januari 2021, van https://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT

TME. (2020, 9 oktober). Hoe werkt een versnellingsmeter en waar dient hij voor? Geraadpleegd op 2 januari 2021, van https://www.tme.eu/nl/news/library-articles/page/22568/hoe-werkt-een-versnellingsmeter-en-waar-dient-hij-voor/

Valori. (z.d.). ISO9126 kwaliteitsmodel [Figuur]. http://www.smartest.nl/inc/upload/ibrowser/ISO9126%20model.gif

Wikipedia contributors. (2021, 5 april). Rotation formalisms in three dimensions. Wikipedia. https://en.wikipedia.org/wiki/Rotation_formalisms_in_three_dimensions

Williams, B. (2018). BICYCLE CRASH DETECTION: USING A VOICE-ASSISTANT FOR MORE ACCURATE REPORTING. Geraadpleegd van https://www.cs.uoregon.edu/Reports/MS-201806-Williams.pdf

Wirth, R., & Hipp, J. (2000). CRISP-DM: Towards a Standard Process Model for Data Mining (Nr. 1). London, UK: Springer-Verlag. http://www.cs.unibo.it/~danilo.montesi/CBD/Beatriz/10.1.1.198.5133.pdf

Yoshida, T., Mizuno, F., Hayasaka, T., Tsubota, K., Wada, S., & Yamaguchi, T. (2005). A wearable computer system for a detection and prevention of elderly users from falling. Proc ICBM, 179–82.

Bijlagen

Bijlage 1: Requirements Elicitatie Plan

Doel

Met het verzamelen van requirements kan worden bepaald waar de oplossing voor het verbeteren van de crashdetectie aan moet voldoen, welke wensen er zijn en wat niet gedaan zal worden. Met behulp van dit plan kunnen de requirements op methodische wijze verzameld en gedocumenteerd worden. Voor dit onderzoek is het belangrijk om te bepalen waar de oplossing functioneel aan moet voldoen, welke kwaliteitseisen er zijn en met welke andere voorwaarden rekening moet worden gehouden.

Voorbereiding

Om het verzamelen van de requirements op een effectieve manier uit te voeren zijn voorafgaand twee stappen uitgevoerd: De huidige situatie binnen het probleem is beschreven en de stakeholders zijn in kaart gebracht. Met deze kennis is de context omtrent het probleem verduidelijkt zodat er gericht vragen gesteld kunnen worden aan de juiste personen.

Aanpak

Om de requirements te verzamelen zal een interview gehouden worden met de belangrijkste stakeholders die input kunnen geven over waar de oplossing aan moet voldoen.

De belangrijkste stakeholders zijn:

- De product owner
- Developers van het service team
- De CTO

Met de product owner, CTO en een ervaren ontwikkelaar van het service team zal een interview gehouden worden gericht op het verzamelen van requirements. De interviews zullen gericht worden op het definiëren van meerdere typen requirements:

- Functionele requirements
- Kwaliteitseisen (ook wel non functional requirements)
- Randvoorwaarden

De interviews zullen een los karakter hebben waarbij zal worden ingegaan op de expertise van de persoon. Deze personen zijn al op de hoogte van de opdracht zodat er voor het interview geen tijd nodig is om deze uit te leggen. Wel is het belangrijk om aan te geven dat het doel van het interview is om requirements te bepalen waar de oplossing aan moet voldoen. Om het interview richting te geven kunnen de volgende vragen en informatie gebruikt worden:

Functioneel

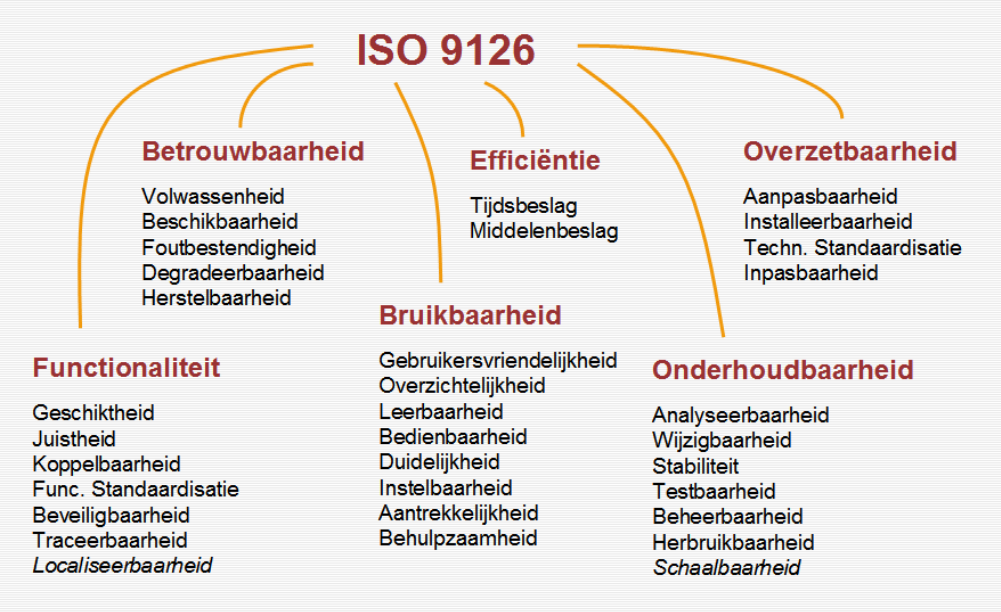
Voor de functionele requirements is het belangrijk om te bepalen wanneer een crash gedetecteerd moet worden. Hierbij kunnen de volgende vragen helpen:

- Wanneer zou een crash als 'valide' aangemerkt moeten worden? Is geen beweging na 60 seconden voldoende? Hoe zou kunnen worden bepaald wat voldoende is?
- Welke grenzen kunnen er gesteld worden aan de crashdetectie? Als iemand bijvoorbeeld een sloot in rijdt moet dit dan ook een crash zijn?
- Wanneer is de crashdetectie goed genoeg? Denk hierbij aan een percentage voor hoe vaak het goed moet zijn/fout mag gaan. Is het bijvoorbeeld voldoende als de crashdetectie 90% van de tijd correct is?

Kwaliteit

Om kwaliteitseisen te bepalen kan onderstaande afbeelding getoond/gebruikt worden waarna specifieke vragen gesteld kunnen worden zoals:

- (Hoe) moet ik rekening houden met de aanpasbaarheid van mijn oplossing? Is het de bedoeling dat elke andere ontwikkelaar grote aanpassingen moet kunnen maken?
- Waar moet rekening mee gehouden met betrekking tot de testbaarheid? Moeten er unit tests toegevoegd worden?
- Zijn er bepaalde tools waar rekening mee moet worden gehouden?



Figuur 21: Valori. (z.d.). ISO 9126 kwaliteitsmodel [Illustratie]. Geraadpleegd van http://www.smartest.nl/inc/upload/ibrowser/ISO9126%20model.gif

Randvoorwaarden

- Met welke beperkingen van de hardware (CK300) moet rekening worden gehouden?
- Is het idee dat er bijvoorbeeld een callcenter gebruikt gaat worden om de gebruiker hulp te bieden nadat er een crash is gedetecteerd?
- Moet er rekening houden met aansluiting op nieuwe hardware die mogelijk op een andere manier crashdetectie heeft?

Documentatie

Het documenteren van de requirements zal gebeuren op basis van user-stories waarbij het volgende format gebruikt zal worden:

	Title	User Story	Importance	Source	Notes
1	Korte titel van de requirement	Wat wil de user bereiken? Beschreven als een user story	Prioriteit op basis van MoSCoW	Waar komt de requirement vandaan	Opmerking om rekening mee te houden
2					

Om het project beheersbaar te houden zullen de requirements voorzien worden van een prioriteit met behulp van de MoSCoW-methode (MoSCoW – projectmanagementsite, z.d.) waarbij wordt bepaald of een requirement een must, should, could of won't betreft. Op basis van deze labels krijgt het onderzoek betere richting waarbij duidelijk is waar de oplossing aan moet voldoen en wat eventueel kan worden onderzocht of uitgevoerd als hier ruimte voor is.

Validatie

Na het uitvoeren van de interviews zullen de requirements gevalideerd worden zodat gecontroleerd is of de gevonden requirements ook echt aansluiten op de eisen en wensen vanuit de belangrijkste stakeholders. De validatie zal uitgevoerd worden door de gedocumenteerde requirements te delen met de product owner, CTO en developers van het service team. Hierbij zal worden gevraagd om feedback en of zij het eens zijn met de opgestelde requirements. Met de product owner zal een extra moment ingepland worden waarbij elke requirement zal worden nagelopen en middels discussie eventueel aanpassingen gemaakt kunnen worden, zodat de product owner uiteindelijk tevreden is.

Bronnen

MoSCoW – projectmanagementsite. (z.d.). Projectmanagementsite. Geraadpleegd op 31 december 2020, van https://projectmanagementsite.nl/moscow/

Valori. (z.d.). ISO9126 kwaliteitsmodel [Figuur]. http://www.smartest.nl/inc/upload/ibrowser/ISO9126%20model.gif

Bijlage 2: Tracker Crash Data test plan

Dit plan heeft als doel om te omschrijven hoe op een betrouwbare en reproduceerbare manier data kan worden vergaard omtrent de crash detectie van de e-bikes.

Tracker test

Om te bepalen of de tracker betrouwbaar is en hoe deze presteert zullen tests uitgevoerd worden waarbij de tracker valt en 'crashed' op een specifieke ondergrond met verschillende oriëntaties. De volgende scenario's zullen worden getest:

#	Hoogte	Vloer	Oriëntatie
1	1 meter	Harde vloer	Stikker naar boven
2	1 meter	Harde vloer	Bovenkant (stekkers) naar boven
3	1 meter	Harde vloer	Zijkant naar boven

Testopstelling:

- CK300 tracker: crashdetectie & data_report aan in configuratie. Sensitivity 25, Sample rate 100hz, 2 samples before crash. 5 samples after crash.
- Meetlat van 1 meter. Hoogte voor het laten vallen gemeten vanaf de onderkant van de tracker.
- Harde vloer = laminaat (Zie onderstaande figuur)
- Tracker zo vrij mogelijk laten vallen. (Dus niet omhoog/omlaag gooien.)
- Een paar seconden vóór het laten vallen even de tracker schudden zodat deze actief wordt en een rest_to_motion event verstuurd.

Na het uitvoeren van een test worden de volgende gegevens gedocumenteerd:

- Crash event JSON
- Crash data reports in JSON
- *Last location report before & first location report after the crash event. (Niet interessant voor tracker test maar wel belangrijk voor volledige data van verschillende scenario's)*

Resultaten = rauwe data + visualisatie + conclusie

Figuur 22: Gebruikte vloer voor tests