25/26

Eindevaluatie

SOLARBOT MEESTERPROEF

KEVIN KANG 6V (MW. ROOS, MR. DE GRAAF)

Technasium

Inhoud

Voorwoord

Dit document bevat de reflectie van fase 2 en de eindevaluatie van ons project SolarBot. Het project liep van 14 september 2025 tot en met 18 februari 2026.

Ons team bestond uit Xinxin Wang, Jono Wink, Jivraj Singh en Kevin Kang. Binnen het project lag de focus van twee teamleden vooral op de hardware en van de andere twee meer op de software, maar we hebben gedurende het hele traject nauw samengewerkt om tot één gezamenlijk eindresultaat te komen. Deze evaluatie vormt de afsluiting van dat proces.

Inleiding

In dit document blik ik terug op mijn persoonlijke ontwikkeling en op de samenwerking binnen het team tijdens de meesterproef. Het project duurde ongeveer vijf maanden. In deze periode hebben we gewerkt aan het ontwikkelen van een systeem dat vervuiling op zonnepanelen kan detecteren met behulp van machine learning en een fysiek, werkende prototype van een robot.

In deze evaluatie bespreek ik wat goed ging, wat beter kon en in hoeverre de gestelde doelen zijn behaald.

TOP

Het doel

Ons teamdoel is om dit schooljaar beter te worden in tijdsbeheer. We willen leren hoe we onze taken goed kunnen plannen en ons aan de afspraken houden. Dit doen we door een gezamenlijke planning te maken, wekelijkse check-ins te houden en samen te kijken of we ons werk op tijd af hebben. Deze keuze is gemaakt omdat in onze betawereld voor dit project energie en natuur, omdat het energie gedeelte van de betawereld veel programmeren zal bevatten. Hiervoor zullen er gestructureerd gewerkt moeten worden, en goed afspraken na kunnen komen, ook als er in verschillende delen van de robot gewerkt wordt en als de teamgenoten niet direct samenwerken aan hetzelfde stuk code. Hiervoor is het doel voor dit project om als team een planning te hebben met afspraken waar iedereen zich aan kan houden, zodat het team gestructureerd werkt en productief is, zelfs als er aan andere delen van het project gewerkt wordt.

Uitwerking

Specifiek: We willen als team beter leren plannen en zorgen dat we ons aan deadlines houden. Deze deadlines kunnen specifiek gesteld worden in Trello en het is dus duidelijk wat er moet gebeuren.

Meetbaar: We vinden dat we dit doel hebben gehaald als minstens 80% van onze taken op tijd af zijn, Dit kunnen wij altijd in onze Trello geschiedenis terugzien, om te zien hoe wij hierin vooruitgaan tijdens het project.

Acceptabel: Iedereen in het team vindt dit een belangrijk doel en staat erachter. Niemand anders wordt door dit doel benadeeld.

Realistisch: Met een gezamenlijke planning en wekelijkse check-ins is dit haalbaar als iedereen in het team zijn best ervoor doet, en zich aan afspraken houdt.

Tijdgebonden: We werken hier het hele schooljaar aan en evalueren elke maand of we op schema liggen, en kunnen in Trello zien of wij onze tijdsgebonden deadlines af krijgen op tijd. Zo niet dan zien wij bij onze wekelijkse check ins hoe het gegaan is en of we goed genoeg bezig zijn.

POP

Het doel

Mijn doel dit jaar is om zoveel mogelijk te leren over machine learning in combinatie met mechanical engineering. Sinds organisaties zoals OpenAI of Deepseek ontwikkelt machine learning zich sneller en sneller, en steeds meer mensen leren hoe ze dit kunnen toepassen in hun eigen projecten. Ik heb zelf al ervaring door een eigen machine learning programma te maken, maar dat was alleen de eerste stap. Door machine learning te combineren met robotica ontstaan er mogelijkheden die eerder ondenkbaar waren. Bedrijven en universiteiten hebben hier al mooie resultaten mee behaald, maar ik wil binnen dit project mijn eerste stap zetten in het toepassen van machine learning op robotica.

Uitwerking

Specifiek: Ik wil leren hoe ik machine learning kan combineren met robotica door een eenvoudig model te ontwikkelen voor een robot.

Meetbaar: De doel is bereikt wanneer ik een getrainde model heeft geschrijven, en ook mee werkt in de robot.

Acceptabel: Dit doel is ook een doel van ons project en sluit aan bij mijn rol als programmeur.

Realistish: Ik heb al programmeer ervaring en basiskennis van machine learning, waardoor het haalbaar is om binnen dit project een werkend prototype te ontwikkelen en te prototypen.

Tijdgebonden: Ik wil dit doel bereiken voor het einde van de meesterproef, en volledige integratie in de robot voor de eindpresentatie.

Reflectie Fase 1

Team Reflectie Fase 1

In fase 1 hadden we als team het doel om beter om te gaan met tijdsbeheer. We wisten vanaf het begin dat dit belangrijk zou worden, omdat het project technisch complex is en uit veel verschillende onderdelen bestaat. Daarom hadden we aan het start van het project een concreet leerpunt opgesteld: we wilden een planning hebben die haalbaar was, overzicht gaf en die we als team echt zouden gebruiken.

Wij hadden onderzoek gedaan naar hoe timemanagement stress vermindert in een groot project zoals de meesterproef. Wij kwamen op de conclusie dat als je goed je time managet dat je veel minder stress zou hebben, en ook daardoor veel minder snel geïrriteerd raakt en vervolgens minder snel opgeeft. (Shah, 2016)

De gevolgen van stress zijn slecht voor je gezondheid en werktempo, je kan slaapproblemen krijgen en makkelijker moe worden. Dat wil je absoluut niet in een project waar elk les/ opdracht cruciaal is voor een goed eindproduct opleveren. Daarom is tijd managem ent ook zo belangrijk (Vieveen, 2025)

Wij hadden onderzoek gedaan naar wat de beste manieren zijn om je tijd goed te managen, wij kwamen op Trello, een gratis webapp waar je borden kan maken met taken en die ook verdelen tussen de leden van je team. Het gaf hetzelfde gevoel als de scrumboards waar wij allemaal mee bekend waren, wij wist en al dat die goed werkten waren wij van overtuigd om een Trellobord aan te maken voor ons project.

Maar we moesten wel weten hoe we het het best kon gebruiken. Daar deden wij ook veel onderzoek op, er zijn een paar punten waar wij extra goed op moeten letten zodat het gebruik van de trellobord ook echt werkt. Wij moesten goed flexibel zijn en van tevoren plannen.

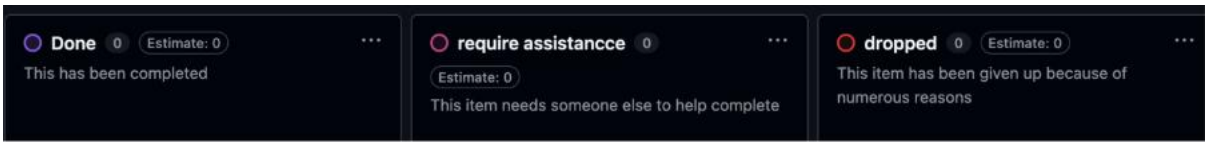
Ook heeft een goed Trellobord minimaal deze vier tabben:

1. Backlog
2. To Do
3. In Progress
4. Done

Dit lijkt dan ook heel erg op de scrumboards die wij vroeger veel gebruikte.

(Radulovic, 2023)

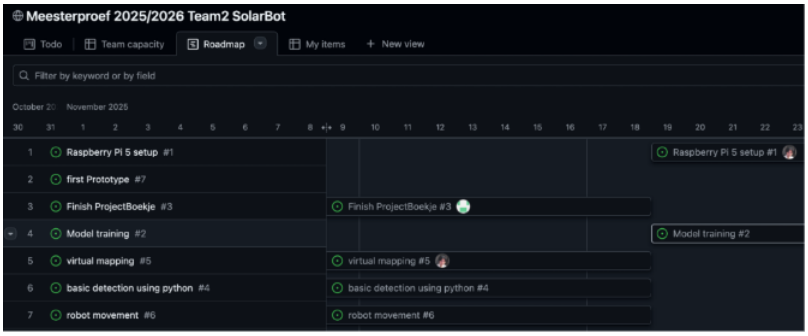
We begonnen met GitHub voor de code en technische bestanden en daar werkten we vanaf het begin prettig mee. Voor de planning gebruikten we eerst Trello, maar na een paar weken merkten we dat dit niet logisch was. Veel taken hadden direct te maken met issues op GitHub, waardoor we vaak tussen twee systemen heen en weer moesten schakelen. Dat vertraagde het proces. Als team hebben we toen besloten om de hele planning naar GitHub Projects te verplaatsen. Dit zorgde voor één duidelijk systeem waarin taken, deadlines en technische onderdelen bij elkaar kwamen. Achteraf gezien was dit een goede keuze, omdat het meer structuur gaf en beter paste bij de manier waarop we aan dit project werken.



Figuur 1 Github Projects

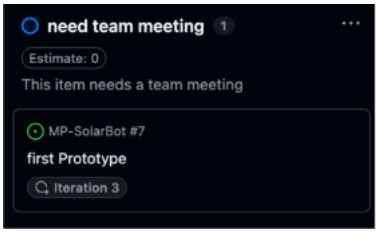
Tijdens deze fase kwamen we er ook achter dat we sommige technische onderdelen hadden onderschat. Machine learning was nieuw voor ons en het kostte tijd om te begrijpen hoe we datasets moesten voorbereiden en hoe TensorFlow precies werkte. Bij het programmeren in C++ voor de besturing van de robot merkten we dat hardware en software niet altijd meteen samenwerkten, waardoor we extra tijd kwijt waren aan testen en aanpassen. Het kiezen van geschikte hardwareonderdelen duurde soms langer dan verwacht, omdat we steeds moesten afwegen wat wel en niet compatibel was. Dit alles had invloed op onze planning en maakte duidelijk dat we onze tijd realistischer moesten inschatten.

Naast de technische uitdagingen waren er ook onderbrekingen in het schoolprogramma. In oktober hadden we een toetsweek en daarna was er een reis naar Parijs voor een deel van ons team, waarna de herfstvakantie volgde. Hierdoor lag het project regelmatig stil. We hadden dit wel ingepland, maar we merkten dat het effect groter was dan gedacht. Vooral het weer opstarten na deze periodes kostte tijd. Dit zorgde ervoor dat we het teamleerdoel opnieuw moesten toepassen: afspraken maken, planning bijstellen en taken opnieuw verdelen.



Figuur 2 Github roadmap

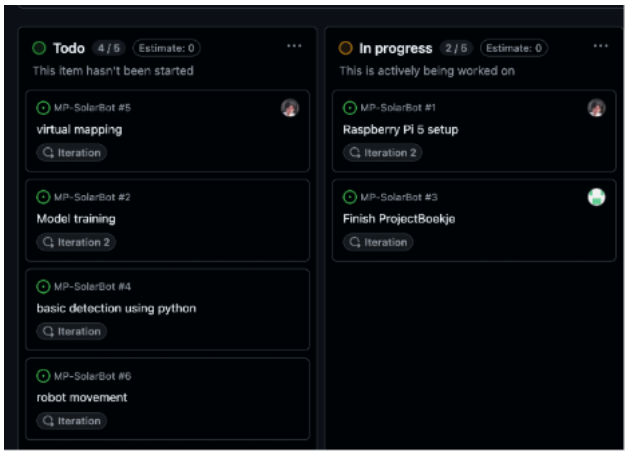
Onze wekelijkse check-ins werden in de loop van fase 1 steeds beter. In het begin waren de gesprekken nog vrij los van structuur, maar later gebruikten we een vaste volgorde: wat is af, waar loopt iemand vast, wat moet er worden aangepast en wat is haalbaar voor de komende week. Hierdoor hadden we meer overzicht en konden we sneller reageren op problemen. Dit sloot goed aan bij ons leerdoel, omdat we merkten dat planning pas werkt als je het team er actief bij betrekt en er regelmatig op terugkomt.



Figuur 3 Github todo

In deze fase hebben we ook gebruikgemaakt van basiskennis uit de theorie die bij ons leerdoel past. We keken bijvoorbeeld naar het opdelen van taken in kleinere stappen en naar het maken van realistische inschattingen. Dit hielp ons om beter te begrijpen waarom onze planning soms niet werkte en wat we konden doen om dat te verbeteren.

Als we terugkijken naar fase 1, dan hebben we ons doel niet volledig gehaald, maar we hebben wel duidelijke vooruitgang geboekt. We zijn beter geworden in het bespreekbaar maken van problemen, het verdelen van taken en het aanpassen van onze planning op basis van wat haalbaar is. De overstap naar GitHub Projects en de gestructureerde check-ins zorgden voor een samenhangend systeem waarin iedereen kon zien wat er moest gebeuren en hoe ver we waren. Dat hielp ons ook om onze eigen bijdrage beter te begrijpen en te zien hoe ons werk waarde heeft binnen het geheel.



Figuur 4 Github todo

Voor fase 2 willen we dit verder verbeteren door taken nog consequenter bij te houden, onze planning beter af te stemmen op drukke periodes en grotere onderdelen vaker op te splitsen. We hebben in fase 1 een basis gelegd en we willen die in de volgende fases sterker maken.

Persoonlijk Reflectie Fase 1

Uitwerking

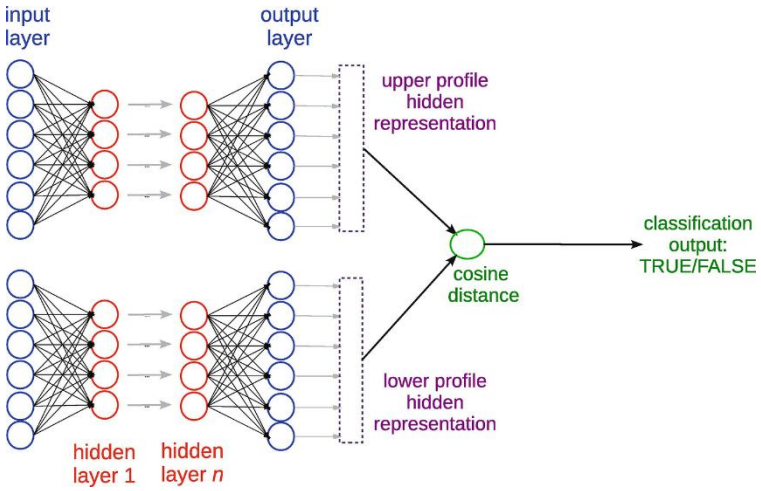
Voor de POP had ik veel gelezen en gewerkt met kunstmatig inteligentie, en daarbij met samenwerking met mijn team een prototype netwerk geschreven. We hebben veel onderzocht, over welke soorten netwerk het beste is voor onze usecase en welke programeer taal het beste is om de netwerk in te schrijven.

Kunstmatig neural netwerk

Een neural netwerk is een groep van verbonden biologische neuronen (zenuwcellen) in onze breinen. Voor een kunstmatige netwerk is het aangestuurt door een computerprogramma, met eigenschappen van een echte neuron. Er zijn veel verschillende kunstmatige netwerken, van de simpelste Perceptron, gebruikt om een keuze temaken, tot GPT(Generative Pre-trained Transformer), een model dat text en zelfs afbeeldingen kan genereeren. (Lee)

Voor onze usecase hebben we gevonden dat de Siamese-netwerk het beste bij ons past. Een Siamese netwerk is een deep learning architectuur die twee of meer identieke neurale netwerken gebruikt om twee inputs te vergelijken en hun gelijkenis of ongelijkheid te bepalen. Of in andere woorden, het vergelijkt twee foto's. (Jane Bromley, 1994)

Dit is namelijk perfect voor onze project, waar we de live camera foto vergelijkt met een foto dat schoon is, en daarmee te bepalen of het schoon word gemaakt of niet.



Figuur 5 Neural netwerk

Programeer taal

Er zijn verschillende programeer talen dat goed werken met AI, maar voor onze usecase vonden we dat python het beste bij ons past. Python is namelijk een energiezuinige programmeer taal, en omdat het een object-oriented (organiseert softwareontwerp rond data of objecten) programeer taal is, is het ook makkelijk te leren. Python is ook een van het meest gebruikte talen ooit, en er word ook veel gebruikt voor machinelearning en AI. (Think Python)

Voor python hebben we een package (een map met modules en mogelijk andere mappen die zelf weer meer mappen en modules kunnen bevatten) gevonden waar we onze neurale netwerken kan trainen. (usage, n.d.)

TensorFlow is een open source machine learning platform, ontwikkeld door Google, waarmee ontwikkelaars machine learning modellen kunnen bouwen en trainen. Het maakt gebruik van dataflowgrafieken, waarbij knooppunten wiskundige bewerkingen vertegenwoordigen en randen multidimensionale data arrays. (Chen, n.d.)

```
def train():
    encoder = build_encoder()
    siamese = build_siamese(encoder)
    siamese.compile(optimizer=Adam(1e-4), loss=contrastive_loss)
    steps_per_epoch = max(100, (len(clean_imgs) + len(dirty_imgs)) // BATCH_SIZE)
    gen = pair_generator(clean_imgs, dirty_imgs, batch_size=BATCH_SIZE)
    print("Training siamese network. Steps per epoch:", steps_per_epoch)
    history = siamese.fit(gen, steps_per_epoch=steps_per_epoch, epochs=EPOCHS,
verbose=2)
    print("Saving encoder and siamese model...")
    encoder.save(ENCODER_SAVE)
    siamese.save(SIAMESE_SAVE)
    print("Done.")
```

Dit is een stukje code waar we onze neurale netwerk trainen.

Reflectie Fase 2

Team Reflectie Fase 2

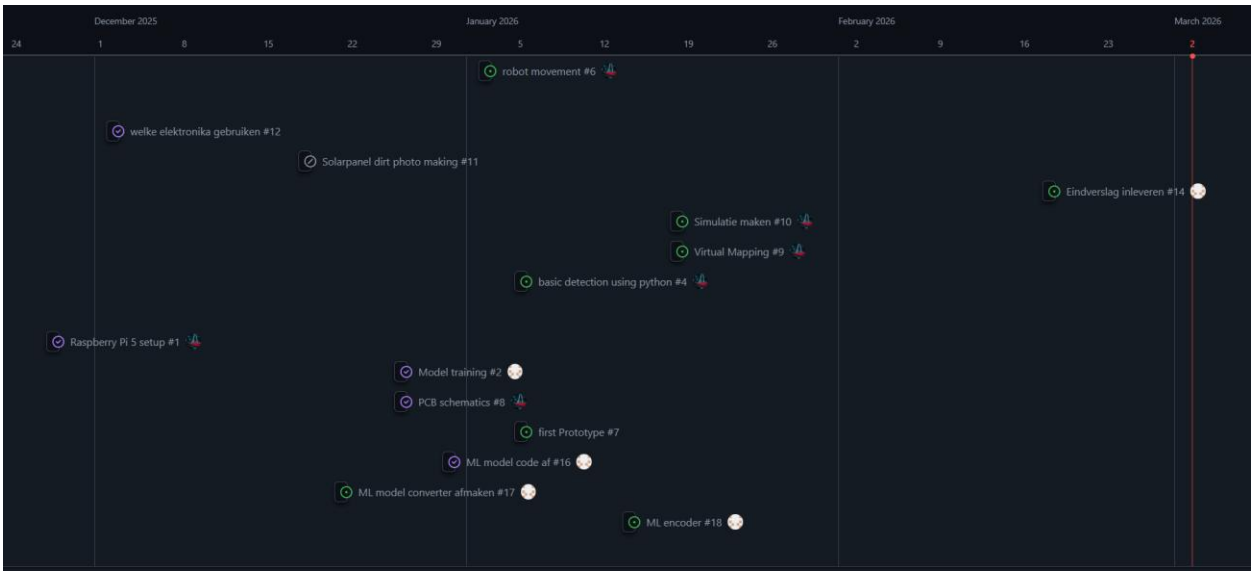
Aan het begin van het schooljaar hebben wij als team in ons TOP afgesproken dat we beter wilden worden in tijdsbeheer. We wilden leren om realistisch te plannen, deadlines serieus te nemen en gestructureerd te werken, vooral omdat het project veel programmeerwerk en technische integratie zou bevatten.

In vergelijking met fase 1 verliep fase 2 merkbaar beter. In het begin van het project liepen we soms achter de feiten aan en werkten we meer reactief dan gepland. De eerste deadline bij het inleveren van het projectboekje is zelfs misgegaan.

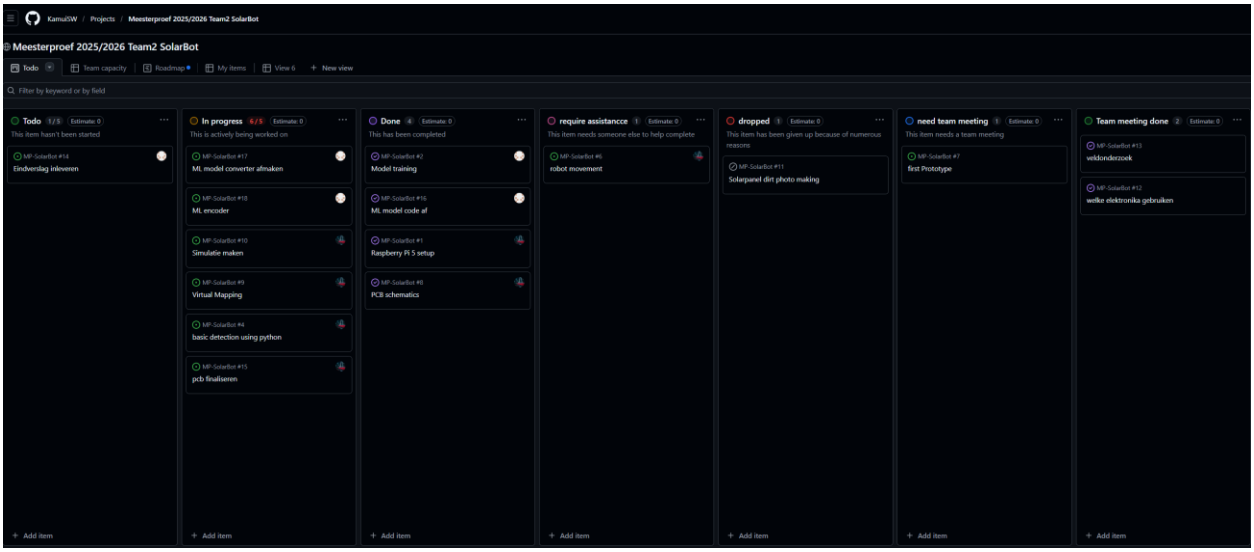
In fase 2 was dit anders. We maakten concretere planningen richting de eindpresentatie en eindverslag. We hielden elkaar vaker verantwoordelijk voor afgesproken deadlines. Hierdoor was er meer duidelijkheid over wie wat moest doen en wanneer het af moest zijn.

We gebruikte GitHub Projects veel meer. Taken werden niet alleen aangemaakt, maar ook gekoppeld aan specifieke “issues” en “commits”.

Hierdoor konden we precies zien wie waaraan werkte en in welke fase een taak zit. We voegden ook deadlines toe aan grotere taken, zodat we beter konden plannen richting de eindpresentatie.



Figuur 6 Deadlines



Figuur 7 Github todo op 1 januari 2026

Binnen het team lag de focus van twee teamleden meer op hardware en van twee meer op software. Hoewel iedereen zijn eigen verantwoordelijkheden had, werd de samenwerking juist belangrijk bij de integratie van het volledige systeem. Hier merkten we dat goed tijdsbeheer erg belangrijk was. Wanneer één onderdeel vertraging opliep, had dat direct invloed op de rest. Dit maakte duidelijk waarom ons TOP-doel zo belangrijk was.

Wat in fase 2 beter ging, was de communicatie. Wanneer iemand vastliep, werd dit sneller besproken in plaats van er te lang zelfstandig mee te blijven worstelen. Dit zorgde ervoor dat technische problemen sneller werden opgelost. Ook planden we gerichter richting de einddeadline, waardoor we meer overzicht hielden dan in fase 1.

Toch hebben we ons doel niet volledig perfect behaald. Richting het einde ontstond opnieuw tijdsdruk. Niet alle afspraken werden altijd even strak nagekomen en sommige integratieproblemen tussen hardware en software kostten meer tijd dan we hadden ingeschat. Dit laat zien dat plannen niet alleen betekent dat je een planning maakt, maar ook dat je realistisch moet inschatten hoeveel tijd iets daadwerkelijk kost.

Als ik terugkijk op ons TOP-doel, denk ik dat we duidelijke vooruitgang hebben gemaakt. We hebben geleerd om gestructureerder te werken en om elkaar aan afspraken te houden. Tegelijkertijd hebben we geleerd dat goed tijdsbeheer discipline vraagt van iedereen in het team, vooral wanneer de druk toeneemt. Fase 2 heeft laten zien dat we als team beter zijn geworden in plannen, maar ook dat dit een vaardigheid is die continu aandacht nodig heeft.

Uiteindelijk hebben we samen een werkend eindproduct kunnen opleveren. Dat is voor mij het bewijs dat onze samenwerking en onze groei in tijdsbeheer daadwerkelijk resultaat hebben gehad.

Persoonlijk Reflectie Fase 2

Uitwerking

In fase 2 ben ik begonnen van theorie in te zetten naar het praktijk. In fase 1 was ik vooral bezig met het begrijpen van neurale netwerken en het trainen van een model, ben ik in fase 2 begonnen met het integratie van het getrainde TensorFlow model in de robot.

Mijn teamgenoot heeft het Siamese netwerk getraind en code voor geschreven met TensorFlow. Mijn taak was om dit model toepasbaar te maken in de robotomgeving. Hiervoor heb ik gebruik gemaakt van C++ en Python op de Raspberry Pi.

Op de Raspberry pi heb ik een Python script geschreven dat camerabeelden kan verwerken, en daarmee de getrainde model oproept om te detecteren of er viezigheden zijn op de zonnepaneel.

Hiermee heb ik een “brug” gemaakt tussen machine learning en mechanische beweging.

Hiernaast heb ik ook geleerd hoe ik schematics kan tekenen en hoe ik de schematic kan omzetten naar een werkende PCB.

Integratie en simulatie

Omdat er risico bestaat dat fouten in de code direct invloed kan hebben op de hardware, heb ik eerst een simulatie omgeving gemaakt.

De eerste simulatie richtte zich op spiraalnavigatie en coordinatenberekening in C++. Hier testte ik wiskundige formules.

In de tweede simulatie heb ik het bewegingsmodel uitgewerkt in Unity. Hierbij werd randdetectie gesimuleerd en werden hoekcoordinaten opgeslagen in een JSON bestand. Dit maakte het mogelijk om een schoonmaakpad te berekenen zonder fysieke robot.

In de eindversie heb ik alles samengebracht in Python, inclusief het TensorFlow Lite model. Hier kon ik viezigheden simuleren, geplande pad en het werkelijke pad visualiseren en het nearest neighbour algoritme testen.

Wat ik hier heb geleerd, is dat een model pas waarde krijgt wanneer het word geintegreerd in een systeem. De simulatie hielp mij om te testen en fouten te isoleren. Ik leerde om eerst virtueel te valideren voordat ik hardware gebruik.

PCB ontwerp

Naast software ben ik begonnen met het ontwerpen van een PCB. Dit begon vanuit een praktisch probleem, de bedrading werd te rommelig en instabiel.

Het eerste ontwerp was zeer compact. Technisch klopte het schema, maar in de praktijk ontstond instabiliteit.

Hier ontdekte ik iets belangrijks. Een schema dat logisch lijkt op papier, werkt niet automatisch in de realiteit.

De geintegreerde buck converter veroorzaakte schakelruis (EMI). Daarnaast waren de koperbanen te lang en te dun voor de stroom die de Raspberry Pi gebruikt. Hierdoor ontstond spanningsverlies en instabiliteit.

In versie twee verbeterde ik de routing en scheidde ik power en signaal beter. Toch bleef de geintegreerde buck converter een probleem.

Na feedback van expert Alec Li heb ik geleerd hoe belangrijk trace width is bij hogere stromen. Met behulp van de formule voor stroomcapaciteit van kopertraces berekende ik dat mijn oorspronkelijke ontwerp te dun was.

$$I = K * \Delta T^{0.44} * (W * H)^{0.725}$$

In versie drie heb ik een groot ontwerpkeuze gemaakt. En externe buck converter module gebruiken inplaats van geintegreerde buck converter, en power planes toevoegen voor beter stabiliteit.

Hier heb ik geleerd dat engineering iteratief is. Je eerste oplossing is meestal niet de beste.

Wat ik heb geleerd

In deze fase heb ik drie belangrijke dingen geleerd:

Ten eerste, machine learning op zichzelf is niet voldoende. Een getraind model is maar een wiskundige functie. Om dit echt te gebruiken moet er veel meer werk achter de laden van het model, optimaliseren, verwerking van data en het vertalen van modeloutput naar gebruikbare mechanische instructies.

Ten tweede, robotica stelt veel striktere eisen dan gewone AI omgevingen. Latency, rekenkracht en geheugen spelen een veel grotere rol dan bij het trainen op een krachtige computer.

Ten derde, hardware ontwerp heeft een andere manier van denken dan software. Elektrische ruis, trace width en spanningsverlies zijn variabelen die je niet kan negeren. Theorie en praktijk verschillen vaak meer dan verwacht.

Conclusie

In fase 1 begreep ik hoe machine learning werkt.

In fase 2 heb ik geleerd hoe moeilijk het is om machinelearning betrouwbaar te laten samenwerken met echte hardware.

Hiermee heb ik mijn POP en doelen gedeeltelijk bereikt. Deze fase heeft mij dichter gebracht bij mijn doel om machine learning prakktisch toepasbaar te maken in mechanische systemen. Ik heb nu niet alleen theoretische kennis, maar ook ervaring met integratie en implementatie.

Team Conclusie

Terugkijkend op SolarBot kunnen we als team zeggen dat we een technisch ambitieus project hebben uitgevoerd. Het combineren van machine learning met hardware-integratie bleek complexer dan vooraf gedacht, vooral bij het samenbrengen van alle onderdelen tot één werkend systeem. Toch hebben we ondanks tegenslagen en momenten van tijdsdruk een functionerend eindproduct gerealiseerd.

Als team zijn we gedurende het project gegroeid. In het begin moesten we nog zoeken naar een effectieve manier van plannen en samenwerken, maar in fase 2 werd duidelijk dat we hierin stappen hadden gezet. De communicatie werd directer, verantwoordelijkheden waren duidelijker en we durfden sneller problemen te bespreken wanneer iets niet werkte. Dit heeft ervoor gezorgd dat we uiteindelijk als één geheel naar het eind konden toewerken.

Op individueel niveau heeft dit project ons niet alleen technisch sterker gemaakt, maar ook bewuster van het belang van structuur, realistische planning en samenwerking. Tegelijkertijd hebben we ervaren dat tijdsbeheer en integratie in technische projecten altijd kritisch blijven. Als we het project opnieuw zouden uitvoeren, zouden we eerder beginnen met integratietesten en strakker sturen op deadlines.

Al met al zien wij de meesterproef als een intensieve maar waardevolle afsluiting van ons zesde jaar. Het project heeft niet alleen geleid tot een concreet eindproduct, maar ook tot duidelijke groei in samenwerking, verantwoordelijkheid en technische ontwikkeling.

Persoonlijk Conclusie

Wanneer ik terug kijk op de twee fases van dit project, zie ik vooral een verandering in mijn manier van denken. In het begin van de meesterproef was ik vooral gefoccust op machine learning als losstaand onderwerp. Ik wilde begrijpen hoe een model werkt en hoe je het kunt trainen. In fase twee heb ik geleerd dat een model pas waarde heeft wanneer het betrouwbaar geintegreerd word in een fysiek systeem.

Ook het ontwerpen van de PCB heeft mijn manier van denken veranderd. ik heb ervaren dat een schema wat op papier correct lijkt, in de praktijk instabiel kan zijn. Problemen zoals spanningsverlies, EMI en trace width dwong mij om verder te gaan dan alleen functionele logica. Engineering is geen lineair proces, maar een iteratieve zoektocht naar stabiliteit en betrouwbaarheid.

Mijn persoonlijk ontwikkelingsdoel was om machine learning te combineren met robotica. Binnen deze twee fases is dat in mijn opinie bereikt. Ik heb niet alleen een model gebruikt, maar het geintegreerd in een embedded syteem en gekoppeld aan mechanische delen.

SolarBot was voor mij niet een eindproduct. Het is mijn eerste serieuze stap richting het werken met intelligente mechanische systemen.

Dank u wel voor het lezen.

Kevin Kang

Literatuurlijst

Chen, J. (sd). *TensorFlow: A System for Large-Scale*. Opgehaald van usenix.org: https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf

Jane Bromley, I. G. (1994). *Signature Verification using a "Siamese" Time Delay Neural Network*. Opgehaald van papers.neurips.cc: https://papers.neurips.cc/paper_files/paper/1993/file/288cc0ff022877bd3df94bc9360b9c5d-Paper.pdf

Lee, F. (sd). *What is a neural network?* Opgehaald van ibm: https://www.ibm.com/think/topics/neural-networks

Radulovic, A. (2023, 2 8). *How I Use Trello To Be More Productive at Work And Manage My Daily Life*. Opgehaald van medium.com: https://medium.com/@angelina.radulovic/how-i-use-trello-to-be-more-productive-at-work-and-manage-my-daily-life-5397b51c0914

Shah, N. (2016, 2 2). *Why is time management key*. Opgehaald van stress.org.uk: https://www.stress.org.uk/why-is-time-management-key/

Think Python. (sd). Opgehaald van allendowney: https://allendowney.github.io/ThinkPython/

usage. (sd). Opgehaald van pip-installer.org: https://web.archive.org/web/20120502155303/http://www.pip-installer.org/en/latest/usage.html

Vieveen, J. (2025, 7 21). *Gevolgen van stress*. Opgehaald van ipracticle.nl: https://ipractice.nl/klachten/stress/gevolgen/