Faculteit Industriële Ingenieurswetenschappen

master in de industriële wetenschappen: informatica

Masterthesis

Gebaarherkenning van de SMOG-gebarentaal voor VR-belevingen

Yara Mijnendonckx

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: informatica

PROMOTOR :
Prof. dr. Nick MICHIELS

BEGELEIDER :
De heer Kristof OVERDULVE

Gezamenlijke opleiding UHasselt en KU Leuven

Universiteit Hasselt | Campus Diepenbeek | Faculteit Industriële Ingenieurswetenschappen | Agoralaan Gebouw H - Gebouw B | BE 3590 Diepenbeek

Universiteit Hasselt | Campus Diepenbeek | Agoralaan Gebouw D | BE 3590 Diepenbeek
Universiteit Hasselt | Campus Hasselt | Martelarenlaan 42 | BE 3500 Hasselt

2023
2024

Faculteit Industriële Ingenieurswetenschappen

master in de industriële wetenschappen: informatica

Masterthesis

Gebaarherkenning van de SMOG-gebarentaal voor VR-belevingen

Yara Mijnendonckx

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: informatica

PROMOTOR :

Prof. dr. Nick MICHIELS

BEGELEIDER :

De heer Kristof OVERDULVE

UHASSELT KU LEUVEN

Woord vooraf

Met het schrijven van deze masterproef rond ik mijn opleiding 'Master in de Industriële Wetenschappen - Informatica' af. Het afronden van de opleiding en de masterproef is gelukt dankzij de motivatie en steun van verschillende mensen die ik hier graag wil bedanken.

Als eerste wil ik mijn promotor, Prof. dr. Nick Michiels, en mijn begeleider, Kristof Overdulve, bedanken voor de goede begeleiding tijdens deze masterproef en om altijd klaar te staan wanneer ik vragen had. Dankzij jullie kennis heb ik veel geleerd, zowel over het onderwerp als over algemene onderzoekscompetenties. Daarnaast wil ik ook Yasmine Wauthier bedanken voor de achtergrondinformatie over de SMOG-gebarentaal.

Verder wil ik mijn kotgenoten en medestudenten bedanken voor hun tijd om de video's op te nemen voor mijn dataset. Zonder deze dataset had ik deze masterproef niet kunnen afronden.

Ten slotte wil ik mijn familie en vrienden bedanken voor hun constante steun en voor de ontspanning en afleiding wanneer dat nodig was.

Inhoudsopgave

Lijst van tabellen

Lijst van figuren

Abstract

Spreken Met Ondersteuning van Gebaren (SMOG) is een vorm van communicatie die gebaren gebruikt voor de kernbegrippen van een gesproken zin. Hierdoor kunnen personen met een communicatieve beperking zich beter uitdrukken. Het aanleren en oefenen van SMOG is echter arbeidsintensief, aangezien één begeleider telkens slechts één persoon kan bijstaan. *Virtual reality* (VR) heeft potentieel om SMOG-gebaren effectiever en speelser in te oefenen en tegelijkertijd de werklast van begeleiders te verminderen. Daarom onderzoekt deze masterproef in welke mate het mogelijk is om een accuraat en kostenefficiënt gebaarherkenningssysteem voor VR te ontwikkelen met behulp van *machine learning*. Eerst werden geschikte sensoren geselecteerd om gebaren op te nemen: de dieptecamera Intel RealSense D435f, een laptopwebcam en een smartphonecamera. Daarna werd een dataset van zeven SMOG-gebaren, uitgevoerd door 13 verschillende personen, samengesteld. Deze data werd vervolgens gebruikt om een *machine learning*-model te trainen, dat met slechts een paar voorbeelden kan leren. Bij gebruik van een smartphonecamera - de sensor die het best scoort op het vlak van gebruiksvriendelijkheid en kostenefficiëntie - vertoonde *transfer learning* met MoViNets bij 4 epochs de hoogste nauwkeurigheid voor het correct classificeren van gebaren, namelijk 99,3%, vergeleken met 95,0% bij de webcam. Dit toont aan dat het mogelijk is om gebruikers in een VR-omgeving te filmen, terwijl zij gebaren uitvoeren en deze gebaren met een hoge nauwkeurigheid correct te classificeren.

Abstract in English

Spreken Met Ondersteuning van Gebaren (SMOG) is a form of communication that utilizes gestures for the core concepts of a spoken sentence, enabling individuals with communicative disabilities to express themselves more effectively. However, teaching and practicing SMOG is labor-intensive, as one instructor can only assist one individual at a time. Virtual reality (VR) has the potential to make practicing SMOG gestures more effective and playful while simultaneously reducing the workload of instructors. Therefore, this master's thesis investigates the feasibility of developing an accurate and cost-efficient gesture recognition system for VR using machine learning. First, appropriate sensors were selected to capture gestures: the Intel RealSense D435f depth camera, a laptop webcam, and a smartphone camera. Subsequently, a dataset of seven SMOG gestures, performed by 13 different individuals, was assembled. This data was then used to train a machine learning model capable of learning with just a few examples. When using a smartphone camera - the sensor that scored best in terms of user-friendliness and cost-efficiency - transfer learning with MoViNets at 4 epochs showed the highest accuracy for correctly classifying gestures, namely 99.3%, compared to 95.0% with the webcam. This demonstrates that it is possible to film users in a VR environment while they perform gestures and correctly classify these gestures with high accuracy.

Hoofdstuk 1

Inleiding

1.1 Situering

Het onderzoek richt zich op het oefenen van Spreken Met Ondersteuning van Gebaren, of kortweg SMOG. SMOG is een communicatiestijl die speciaal ontwikkeld is voor personen met een communicatieve beperking, bijvoorbeeld mensen met het syndroom van Down. Bij SMOG wordt de gesproken taal ondersteund door gebaren, terwijl traditionele gebarentaal op zichzelf staat zonder een gesproken boodschap. Bovendien gebruikt SMOG enkel gebaren voor de kernbegrippen van een zin. Om de gebaren aan te leren maakt men gebruik gemaakt van lijntekeningen [1], [2]. Figuur 1.1 toont de lijntekening voor het SMOG-gebaar "toilet".

Figuur 1.1: SMOG-gebaar voor toilet [3]

Om de volledige boodschap te begrijpen, moet men zowel naar de boodschap luisteren als naar de gebaren kijken. Er zijn drie hoofdparameters die een gebaar bepalen: de plaats (waar ergens bij het lichaam), de vorm en de beweging van de handen. De combinatie van gebaren en gesproken taal kan mensen helpen bij het begrijpen van gesproken taal alsook om zichzelf beter uit te drukken. Een verklaring voor deze positieve effecten van SMOG is dat veel mensen gemakkelijker visuele informatie verwerken dan informatie langs auditieve weg [1], [2].

Het aanleren en oefenen van SMOG is echter een arbeidsintensief proces, waarbij één begeleider meestal slechts een kleine groep of zelfs maar één persoon tegelijkertijd kan bijstaan. Het potentieel van *virtual reality* (VR) in dit domein is aanzienlijk. Het aanleren van de gebaren blijft de taak van de SMOG-begeleider, terwijl de VR-ondersteuning in de oefenfase wordt ingezet. Het gebruik van VR kan op die manier de effectiviteit van trainingen vergroten en tegelijkertijd de werklast van de begeleider verminderen. VR kan worden ingezet om de basisgebaren interactief

te oefenen, waardoor de personen worden gestimuleerd om actief aan de slag te gaan met de gebaren. Dit kan leiden tot een efficiëntere verwerving en diepgaandere kennis van de gebaren, wat op zijn beurt de taalontwikkeling en zelfstandigheid ten goede kan komen.

Het project, waarbinnen deze masterproef kadert, is gestart door het kenniscentrum Onderzoek Levenslang Leren & Innoveren (OLLI) en de graduaatsopleiding Orthopedagogische Begeleiding van de AP Hogeschool Antwerpen. Deze opleiding streeft ernaar om een SMOG-opleiding in hun curriculum te integreren naar aanleiding van vragen uit het werkveld. Daarnaast sluit dit project aan bij een lopend onderzoek binnen OLLI, genaamd GLaDoS, waarbij men nagaat hoe spelgebaseerd leren ingezet kan worden in het onderwijs [4]. In de toekomst is er een mogelijkheid dat de AP Hogeschool Antwerpen en de Universiteit Hasselt gezamenlijk verder zullen werken aan het SMOG-project.

1.2 Probleemstelling

Het probleem dat zich voordoet, is de uitdaging van het automatisch identificeren en classificeren van gebaren in een VR-omgeving. Om dit te verwezenlijken is er nood aan een nauwkeurig gebaarherkenningssysteem dat in VR toegepast kan worden. Dit systeem moet echter ook kostenefficiënt zijn en vereist bij voorkeur minimale extra infrastructuur. In het ideale geval is enkel een VR-bril met ingebouwde camera's nodig om dit te realiseren. Daarnaast moet men onderzoeken welk *machine learning*-algoritme het meest geschikt is om een accuraat classificatiemodel te creëren voor het onderscheiden van de verschillende gebaren.

De hoofdonderzoeksvraag is: in welke mate is het mogelijk om een nauwkeurig en kostenefficiënt gebaarherkenningssysteem voor VR te ontwikkelen met behulp van *machine learning*, met als doel het oefenen van SMOG-gebaren?

Om de hoofdonderzoeksvraag te beantwoorden, zullen verschillende deelvragen worden onderzocht. Ten eerste zal er worden geëvalueerd welke sensoren het meest geschikt zijn voor het detecteren van gebaren binnen een VR-omgeving. Hiervoor wordt gebruikt gemaakt van recente inzichten uit de literatuur [5], [6]. Vervolgens zal er worden nagegaan op welke manier een dataset kan worden samengesteld en verwerkt voor het trainen van een classificatiemodel. Verder zal er worden onderzocht welk *machine learning*-algoritme het meest geschikt is om een accuraat classificatiemodel te trainen om de verschillende gebaren realtime te onderscheiden. Ten slotte zullen de ontwikkelde prototypes geëvalueerd worden op basis van allerhande criteria, waaronder nauwkeurigheid en realtime prestaties, waarbij verwezen wordt naar naar de tijd die een algoritme nodig heeft om een gebaar correct te herkennen.

1.3 Doelstellingen

De hoofddoelstelling is de volgende: het ontwikkelen van een geautomatiseerd systeem tegen begin juni 2024, dat sensorgegevens (bijvoorbeeld uit de ingebouwde camera's van een VR-bril) verwerkt en deze gegevens realtime classificeert met een hoge nauwkeurigheid.

De hoofddoelstelling kan opgesplitst worden in onderstaande deeldoelstellingen.

- Geschikte sensoren en hun eigenschappen voor het vastleggen van de gebaren in een VR-omgeving identificeren;

- Verschillende algoritmen of *Software Development Kits* (SDK's) voor gebaarherkenning in een VR-omgeving onderzoeken;
- Een beoordelingsmatrix opstellen om verschillende sensoren te evalueren op basis van criteria zoals haalbaarheid binnen het tijdsbestek, kostenefficiëntie en nauwkeurigheid;
- Video's van elk gebaar opnemen met behulp van de geselecteerde sensoren;
- Uitgebreide dataset creëren en alle data correct labelen voor het trainen van het classificatiemodel;
- Het classificatiemodel met behulp van *machine learning*-algoritmen trainen, met een focus op nauwkeurigheid en realtime prestaties;
- De prototypes evalueren op basis van criteria zoals nauwkeurigheid, realtime prestaties en gebruiksvriendelijkheid, om het beste prototype te bepalen.

1.4 Methode

Het onderzoek is opgedeeld in vijf opeenvolgende werkpakketten om een gestructureerde aanpak te bieden. Dit zorgt voor een duidelijke organisatie van taken en maakt het mogelijk om de voortgang nauwkeurig bij te houden.

WP1: Literatuurstudie
In het eerste werkpakket zal een uitgebreide literatuurstudie in de state-of-the-art van gebarentaalherkenning via *machine learning* worden uitgevoerd om met het resultaat hiervan mogelijke geschikte sensoren te identificeren voor het vastleggen van gebaren in een VR-omgeving. De literatuurstudie zal zich richten op recente ontwikkelingen in VR-technologieën en gebaarherkenningssystemen.

WP2: Onderzoek naar prototypes
Dit werkpakket richt zich op het onderzoeken van verschillende sensoren (bijvoorbeeld camera, dieptesensoren, ...) voor gebaarherkenning in een VR-omgeving. Eerst worden de bestaande sensoren onderzocht. De focus ligt op het evalueren van de mogelijkheden en beperkingen van elke optie, aan de hand van criteria zoals nauwkeurigheid en realtime prestaties. Daarna wordt deze informatie verzameld in een beoordelingsmatrix om tot slot de best beoordeelde sensoren te bepalen.

WP3: Data verzamelen
Dit werkpakket richt zich op het verzamelen van data van elk gebaar met behulp van de geselecteerde sensoren om een uitgebreide dataset te creëren. De deeltaken omvatten het opzetten van een opname-opstelling om gebaren vast te leggen, code schrijven om de opnames te automatiseren, en het verwerken en labelen van de verzamelde data. Hierna zal nog een kwaliteitscontrole uitgevoerd worden om ervoor te zorgen dat de data bruikbaar is voor het trainen van het model.

WP4: Model trainen
In dit werkpakket wordt de uitgebreide dataset gebruikt om het classificatiemodel te trainen. Dit omvat het onderzoeken van verschillende *machine learning*-algoritmes en het trainen van een model. Na initiële training en validatie wordt het model geoptimaliseerd en verfijnd. Tot slot wordt het model geëvalueerd met behulp van de testset om de uiteindelijke prestaties te meten.

WP5: Evaluatie prototypes
Het laatste werkpakket omvat de evaluatie van de ontwikkelde prototypes. De prototypes worden

beoordeeld op basis van criteria zoals nauwkeurigheid, realtime prestaties en gebruiksvriendelijkheid om op deze manier het beste prototype te bepalen.

Tijdens het onderzoek wordt er gewerkt aan het schrijven van de scriptie, dat na de afronding van het onderzoek wordt voortgezet.

1.5 Vooruitblik

Deze vooruitblik geeft een overzicht van de komende hoofdstukken. Het volgende deel is de literatuurstudie die gebarenherkenning, inclusief toepassingen, onderverdelingen van gebarentaalherkenning en uitdagingen bij handtracking onderzoekt. Vervolgens is er de methodologie van het onderzoek, met de selectie van de hardware, de creatie van een dataset, en de ontwikkeling van het *machine learning*-model. Verder, de resultaten die tonen hoe goed het model presteert in het herkennen van de gebaren, gevolgd door de discussie van deze resultaten. Tot slot geeft de conclusie een samenvatting van de belangrijkste bevindingen en aanbevelingen voor toekomstig onderzoek.

Hoofdstuk 2

Literatuurstudie

Deze literatuurstudie onderzoekt gebarenherkenning, inclusief de toepassingen, onderverdelingen van gebarentaalherkenning, en de uitdagingen die zich hierbij voordoen. Dit onderzoek is uitgevoerd om een uitgebreide kennis te creëren rond gebarenherkenning en handtracking, aangezien deze onderwerpen centraal staan in deze masterproef.

Verder worden in de volgende sectie verschillende soorten sensoren, en daarmee de hardware voor handtracking, geanalyseerd. Dit is cruciaal om een rationale te voorzien voor de gemaakte keuze van hardware voor de uitvoering van deze masterproef. Tot slot worden *machine learning*-technieken besproken die effectief zijn bij het trainen met beperkte data. *Transfer learning* wordt hierbij als een veelbelovende oplossing geïdentificeerd en daarom nader toegelicht.

2.1 Gebarenherkenning

Om gebaren te kunnen detecteren en interpreteren, is het noodzakelijk om gebruik te maken van technologieën zoals handtracking, die de beweging van handen kunnen digitaliseren. Handtracking omvat het detecteren en volgen van handbewegingen, waarbij sommige technieken gebruikmaken van segmentatie om de handen van de omgeving te scheiden, terwijl andere technieken direct op de videobeelden werken zonder expliciete segmentatie [7]. Binnen het domein van *virtual reality* wordt handtracking eveneens toegepast om traditionele controllers te vervangen. Het beoogde doel daarbij is om interacties die typisch zijn voor de fysieke wereld vlot over te brengen naar de virtuele omgeving [8].

Er zijn verschillende methoden om handbewegingen te observeren en te registreren via sensoren. Twee veelgebruikte benaderingen zijn methoden gebaseerd op *data gloves* en methoden gebaseerd op computervisie. Beide benaderingen hebben hun eigen voordelen en toepassingen die in Sectie 2.2 verder toegelicht worden.

2.1.1 Toepassingen van gebarenherkenning

Gebarenherkenning kent een breed scala aan toepassingen in diverse domeinen:

- detecteren van leugens;
- medisch monitoren van stressniveaus;
- manipuleren van virtuele omgevingen;

- monitoren van de alertheid van autobestuurders;
- herkennen van gebarentaal [9].

De vele toepassingen van gebarenherkenning tonen zijn multidisciplinaire karakter en impact op menselijke communicatie en technologie.

2.1.2 Onderverdeling van gebarentaalherkenning

Het herkennen van gebarentaal kan worden onderverdeeld in twee verschillende categorieën op basis van de aard van de invoergegevens: statisch en dynamisch. Statisch verwijst naar een enkel beeld dat een gebaar impliceert, dat wil zeggen een gebaar zonder beweging, terwijl dynamisch gebaren bewegingen omvatten [6]. Een studie uitgevoerd door Wadhawan en Kumar [10] onderzocht de verdeling van gepubliceerde literatuur over deze categorieën gedurende de periode 2007-2017. De resultaten brachten aan het licht dat 45% van de gepubliceerde onderzoeken zich richtte op statische gebaren, terwijl 40% zich concentreerde op dynamische gebaren, en nog eens 15% gewijd aan een combinatie van beide aspecten.

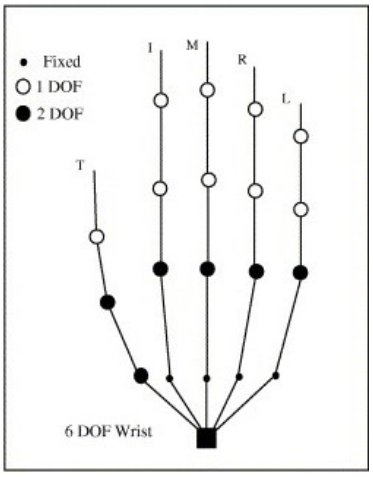
Dynamische invoergegevens kunnen verder worden onderverdeeld in geïsoleerde dynamische invoer en continue dynamische invoer. Geïsoleerde dynamische invoer betreft afzonderlijke woorden, terwijl continue dynamische invoer betrekking heeft op zinnen, waarbij de volledige beweging moet worden opgesplitst in verschillende woorden en dus gebaren [6]. Uit het onderzoek van Wadhawan en Kumar [10] is gebleken dat een aanzienlijk deel van de gepubliceerde literatuur zich concentreert op geïsoleerde invoer, namelijk 83%, terwijl 12% gericht is op continue invoer, en nog eens 5% op een combinatie van beide. Het is belangrijk op te merken dat deze verhoudingen niet exclusief van toepassing zijn op dynamische invoergegevens, maar een algemeen beeld geven.

2.1.3 Uitdagingen bij handtracking

Uiteraard gaat elke technologie gepaard met verschillende problemen. Het vastleggen van driedimensionale handbewegingen wordt beschouwd als een complex probleem [11]. In wat volgt worden enkele van de uitdagingen die zich voordoen vermeld.

Als eerste zijn handen lastig te herkennen voor computersystemen vanwege zelfocclusie. Regelmatig verbergen delen van de hand andere delen, wat het moeilijk maakt om de vorm te identificeren. Dit probleem wordt extra lastig doordat handen draaien en plooien, waardoor vingers elkaar langdurig kunnen bedekken, zelfs bij een gunstig perspectief [12], [13].
Bovendien spelen ongecontroleerde omgevingen ook een rol. In dergelijke omgevingen, met wisselende achtergronden en lichtinvallen, is het voor veel systemen lastig om objecten zoals handen te herkennen. Vooral complexe achtergronden met huidtinten, structuren of bewegende objecten maken het lastig om de contouren van handen nauwkeurig te onderscheiden. Veranderingen in licht spelen hierin ook een rol [12], [14].
Verder vormen de variatie in grootte en vorm van menselijke handen een uitdaging bij handtracking, omdat standaardalgoritmen moeite kunnen hebben om deze diversiteit nauwkeurig vast te leggen. Dit kan leiden tot inconsistente trackingprestaties tussen verschillende gebruikers [12].
Ten slotte stelt het hoge dimensionale karakter van handen een uitdaging, gezien de menselijke hand een complex object is met meer dan 20 vrijheidsgraden, zie Figuur 2.1. Ondanks dat natuurlijke handbewegingen niet altijd alle 20 vrijheidsgraden gebruiken vanwege de onderlinge afhankelijkheden tussen vingers, is uit onderzoek gebleken dat het niet mogelijk is om minder

dan zes dimensies te gebruiken. Dit resulteert in een groot zoekgebied met een enorm aantal mogelijke handposities [12], [14].

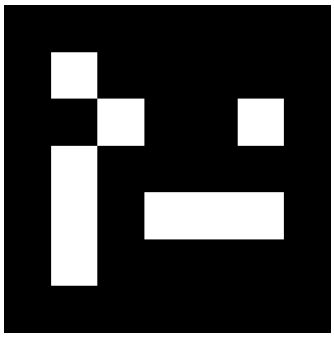


Figuur 2.1: Hand met 27 vrijheidsgraden (*degrees of freedom*, of kortweg DOF) [14]

2.2 Soorten hardware voor handtracking

Handtracking, essentieel voor het herkennen van gebaren, vereist het gebruik van sensoren of andere hardware om de handbewegingen waar te nemen. Hoewel onderzoek naar dit domein al in de jaren 1960 begon, werd het pas in 1992 publiekelijk bekend met de aankondiging van de Apple Newton, een persoonlijke digitale assistent [15].

Het bepalen van de positie van objecten in de ruimte, en daarmee ook van handbewegingen, kan over het algemeen op twee manieren worden uitgevoerd: met behulp van markers en zonder markers. Een prominente vertegenwoordiger van *marker-based* tracking is de ArUco marker (zie Figuur 2.2). Dergelijke vierkante binaire markers worden vaak ingezet om de positie van bijvoorbeeld een camera te bepalen vanwege hun vermogen om dit op een snelle en robuuste wijze te realiseren [16]. Dit hoeft niet noodzakelijk beperkt te blijven tot externe beelden, maar het kan ook een paar handschoenen met vaste kleurcodering impliceren [17]. Daarentegen maakt *markerless* tracking geen gebruik van expliciete markers, maar gebruikt het alternatieve kenmerken of past het algoritmen toe om handbewegingen te identificeren en te volgen.



Figuur 2.2: Voorbeeld van een ArUco marker

Er zijn twee gangbare benaderingen voor het interpreteren van gebaren. De eerste benadering is gebaseerd op het gebruik van *data gloves*, die ofwel draagbaar zijn of rechtstreeks contact maken

met de handen. De tweede benadering maakt gebruik van computervisietechnieken, waarbij geen sensoren op het lichaam gedragen hoeven te worden [18], [19]. In wat volgt, worden deze benaderingen uitvoeriger toegelicht aan de hand van voorbeelden, waarbij opgemerkt dient te worden dat dit geen volledige lijst is van alle mogelijke methoden.

2.2.1 Gebaseerd op data gloves

De *data gloves* kunnen in twee types worden ingedeeld: actief en passief. Actieve systemen omvatten handschoenen met ingebouwde sensoren of een versnellingsmeter. Eerder werden deze handschoenen via kabels met de computer verbonden, maar door de vooruitgang van draadloze technologieën is dit nu niet langer nodig. Anderzijds verwijst de term passief naar handschoenen die geen elektronische onderdelen bevatten, maar gebruikmaken van kleuren om beeldherkenning te vergemakkelijken [5].

Passieve data gloves

Wang en Popović [20] introduceerden een realtime handtrackingmethode met een gekleurde handschoen. De handschoen heeft een uniek patroon van 20 patches in tien kleuren (zie Figuur 2.3), wat leidt tot een efficiëntere vergelijking met een uitgebreide database. Door de afbeelding van de handschoen te matchen met dieptewaarden, wordt nauwkeurig de positionering en oriëntatie van de hand bepaald. Deze aanpak vereenvoudigt het handtrackingproces aanzienlijk door de houding en positie van de hand te combineren.

Figuur 2.3: Gekleurde handschoenen ontwikkeld door Wang en Popović [20]

Actieve data gloves

Zoals eerder aangegeven, maken actieve *data gloves* gebruik van een reeks sensoren, waaronder wellicht versnellingsmeters, om de bewegingen van de vingers nauwkeurig vast te leggen. In 2021 waren er volgens Caeiro-Rodríguez *et al.* 24 verschillende modellen van *data gloves* commercieel beschikbaar wereldwijd [21]. Een voorbeeld van dergelijke technologie is de Nova van SenseGlove (zie Figuur 2.4), een Nederlands bedrijf. De Nova is uitgerust met meerdere sensoren die de buiging en strekking van alle vingers, behalve de pink, meten. De verlenging van de draden, aanwezig in de Nova, draagt bij aan de meting van de voorgaande parameters. Het prijskaartje voor één paar van deze *data gloves* bedraagt €4499 (exclusief btw) [22].

2.2.2 Gebaseerd op computervisie

De methoden gebaseerd op computervisie maken gebruik van visuele informatie om handbewegingen te interpreteren. Het voornaamste onderscheid met *data glove* methoden is dat er geen nood is aan het dragen van fysieke sensoren op het lichaam, wat een meer natuurlijke interactie

Figuur 2.4: SenseGlove Nova [22]

mogelijk maakt en een kostenefficiënter alternatief biedt. Daarentegen, vereist deze aanpak uitgebreide algoritmen en technieken om de handbewegingen accuraat vast te leggen [18], [23]. De verschillende relevante methoden zullen in de volgende secties nader besproken worden.

RGB-camera

De meest voorkomende apparaten voor het vastleggen van beelden zijn monoculaire RGB-camera's zoals die te vinden zijn in laptops of smartphones. Een onderzoek uitgevoerd door Wadhawan en Kumar [10] illustreert dat 55% van de gepubliceerde literatuur camera's gebruikt, aangezien veel studies afhankelijk zijn van gefotografeerde datasets. Hedendaagse digitale camera's registreren licht via drie hoofdkanalen: rood, groen en blauw [5]. De resulterende afbeelding wordt georganiseerd in een *pixelarray* met RGB-waarden, wat een essentiële basis vormt voor computervisie-algoritmen om objecten te identificeren [5].

Dieptecamera

Dieptecamera's vormen een variant op de traditionele RGB-camera's door toevoeging van een extra kanaal voor diepte-informatie, naast de drie kleurenkanalen. Dit extra kanaal biedt inzicht in de afstand tussen de lens en het object [5]. Van dergelijke camera's is de RealSense-serie van Intel (zie Figuur 2.5a), die een scala aan modellen aanbiedt tussen €250 en €500, elk met unieke eigenschappen, een voorbeeld. Liao *et al.* hebben een nauwkeurigheid voor het classificeren van gebaren van 99,4% weten te behalen op hun eigen Amerikaanse Gebarentaal dataset met behulp van een dieptecamera van Intel RealSense [24].

Microsoft Kinect

De Microsoft Kinect (zie Figuur 2.5b), met een prijs van €18,99 [25], heeft de interesse in gebarentaalherkenning vergroot door zijn kostenefficiënte dieptecamerafunctie. Deze technologie omvat zowel een dieptecamera als een videocamera, waardoor het mogelijk is om handbewegingen nauwkeurig vast te leggen [26]. In een spelomgeving betekent dit dat de Kinect handbewegingen kan detecteren en interpreteren, wat leidt tot interactieve en meeslepende spelervaringen. 20% van het onderzoek naar gebarentaalherkenning maakt immers gebruik van de Kinect volgens Wadhawan en Kumar [10]. Bovendien heeft deze benadering geleid tot verbeteringen in de kosteneffectiviteit van gebarentaalherkenningssystemen [27], [28].

Leap Motion Controller

De Leap Motion Controller 2 (zie Figuur 2.5c) biedt nauwkeurige handtracking met een focus op handpositionering. In tegenstelling tot de Kinect-sensor, die het volledige lichaam vastlegt, richt de Leap Motion zich op hand- en vingerniveau [29]. Met twee camera's en drie infrarood-LED's [30] en dankzij een nauwkeurigheid van ongeveer 0,01 mm kan de Leap Motion de 3D-positie van handen bepalen, waardoor het geschikt is voor gebarentaalclassificatie. Deze betaalbare sensor heeft een prijs van ongeveer €130 [31]. Volgens onderzoek uit 2021 gebruikt ongeveer 6% van de gebarentaalherkenningssystemen de Leap Motion Controller [10].

(a) Intel RealSense D435f [32] (b) Microsoft Kinect [25] (c) Leap Motion Controller 2 [31]

Figuur 2.5: Verschillende dieptecamera's

2.3 Machine learning

Machine learning is een vorm van artificiële intelligentie waarbij de focus ligt op het creëren van systemen die gegevens kunnen leren door patroonherkenning. Dit kan op twee manieren plaatsvinden: *supervised learning* en *unsupervised learning*, respectievelijk leren met en zonder toezicht. *Supervised learning* gebruikt gelabelde data om te bepalen of de voorspelling correct is. Om voorspellingen te maken bij *unsupervised learning* moet het model, in tegenstelling tot *supervised learning*, zelf patronen en structuren in ongelabelde data achterhalen [33].

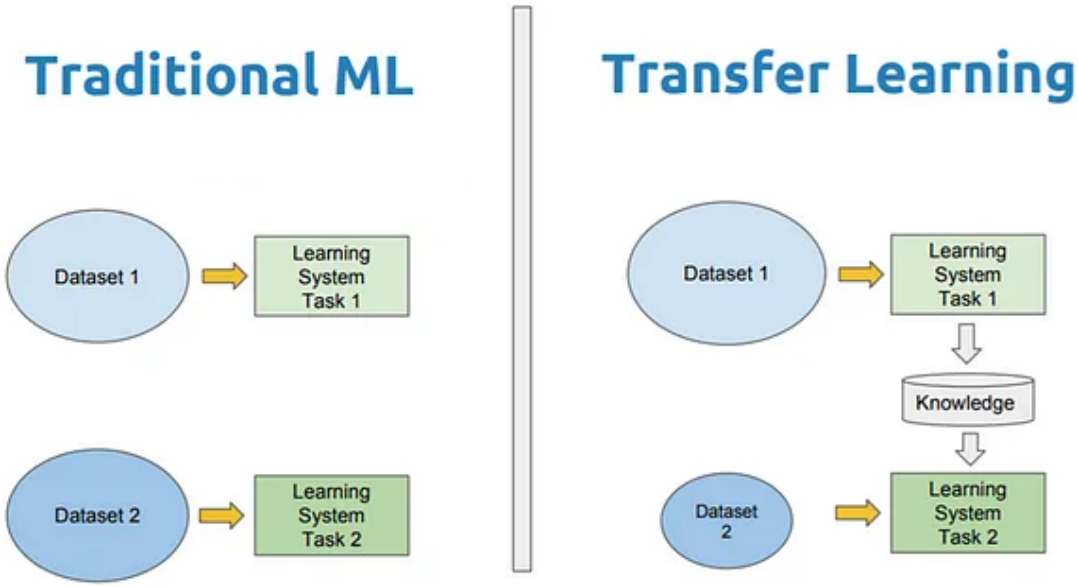
2.3.1 Machine learning met beperkte data

Een van de grootste uitdagingen bij het gebruik van *machine learning*-modellen voor classificatieproblemen is de noodzaak aan grote hoeveelheden gelabelde data om de modellen te trainen. Het verzamelen en labelen van dergelijke data blijkt in veel gevallen praktisch onmogelijk te zijn omdat het zeer tijdrovend is [34].

Semi-supervised learning biedt een gedeeltelijke oplossing voor dit probleem doordat het niet dezelfde grote hoeveelheden gelabelde data vereist. In plaats daarvan kan *semi-supervised learning* met een beperkte hoeveelheid gelabelde data, aangevuld met een grotere hoeveelheid ongelabelde data, een nauwkeurig model genereren. Echter, het blijft in sommige gevallen moeilijk om zelfs deze grotere hoeveelheid ongelabelde data te verzamelen [34]. *Semi-supervised learning* tracht representaties uit de beschikbare data te halen, welke vervolgens ingezet en verfijnd kunnen worden om specifieke problemen met meer efficiëntie op te lossen. Een andere oplossing is *transfer learning*, dat in de volgende sectie nader wordt toegelicht.

2.3.2 Transfer learning

Transfer learning, een veelbelovende techniek binnen *machine learning*, biedt een oplossing voor het bovenstaande probleem door zich te richten op het overdragen van kennis tussen verschillende domeinen. Dit proces wordt geïllustreerd in Figuur 2.6. In een traditioneel *machine learning*-model wordt er telkens gebruik gemaakt van een aparte dataset om een specifieke taak uit te voeren. In *semi-supervised learning* wordt aanvankelijk een voorgetraind model geïnitialiseerd op basis van de beschikbare data, waarna dit model wordt aangepast door middel van finetuning voor een specifieke taak. *Transfer learning* daarentegen begint met het trainen van een model om taak X uit te voeren met behulp van een grote dataset. Vervolgens wordt de verworven kennis getransfereerd om een ander model voor taak Y te trainen met de kleinere beschikbare dataset. Door gebruik te maken van het voorgetrainde model, kan dit nieuwe model al gebruikmaken van relevante features van taak X voor taak Y zonder het van nul te moeten leren, wat resulteert in hoge nauwkeurigheden ondanks de beperkte hoeveelheid beschikbare data [35], [36].



Figuur 2.6: Traditionele *machine learning* versus *transfer learning* [35]

Hoofdstuk 3

Methode

De methode bestaat uit drie fasen. De eerste fase omvat de selectie en evaluatie van hardware voor de herkenning van SMOG-gebarentaal binnen een VR-omgeving. Hierbij worden diverse hardware-opties vergeleken en een weloverwogen selectie gemaakt. In de tweede fase wordt een dataset van SMOG-gebaren samengesteld door video's op te nemen van verschillende personen die de gebaren uitvoeren. De laatste fase richt zich op het trainen van een *machine learning*-model door middel van *transfer learning*. Hierbij worden verschillende parameters geoptimaliseerd om de nauwkeurigheid van het model te verbeteren, met als uiteindelijke doel de effectieve implementatie van gebarentaalherkenning in VR.

3.1 Keuze hardware

De keuze van geschikte hardware vormt de eerste cruciale fase in het ontwikkelingsproces van een gebarentaalherkenningssysteem. Deze fase bepaalt niet alleen de mogelijkheid om gebaren op te nemen en te herkennen, maar heeft ook een directe invloed op de nauwkeurigheid en dus betrouwbaarheid van het uiteindelijke model. Daarom is het essentieel om zorgvuldig te evalueren welke hardware het meest geschikt is voor het vastleggen van handbewegingen.

3.1.1 Opties hardware

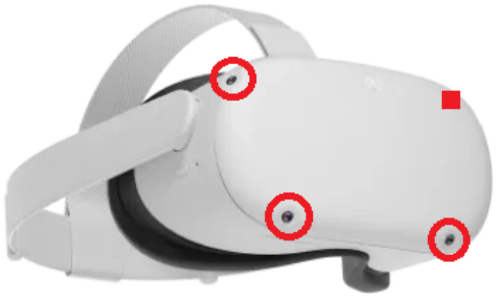
Uit de literatuurstudie bleek dat er verschillende opties waren om als mogelijke hardware te dienen. Deze opties zijn:

- passieve *data gloves*,
- actieve *data gloves*,
- RGB-camera,
- dieptecamera,
- Microsoft Kinect,
- Leap Motion Controller.

Deze hardware-opties zijn geselecteerd mede omdat ze beschikbaar gesteld konden worden voor deze masterproef.

De ingebouwde camera's van de VR-bril werden aanvankelijk overwogen als hardware, aangezien dit voornamelijk de meest handige, maar ook de meest kostenefficiënte methode zou zijn. Het

doel is namelijk om het getrainde model later in VR te implementeren, waardoor er bij het gebruik van de ingebouwde camera's geen extra externe hardware nodig zou zijn. Echter, deze optie werd verworpen omdat ontwikkelaars geen toegang hebben tot de ruwe visuele data van deze camera's. Bovendien is het gezichtsveld (*field-of-view*) mogelijk te beperkt voor sommige gebaren, aangezien deze gebaren zich vaak ter hoogte van de schouders afspelen, wat buiten het bereik van de ingebouwde camera's van welke VR-bril dan ook valt. De Meta Quest 2, een wijdverspreid model van VR-bril, heeft een *field-of-view* van 90° horizontaal en verticaal [37]. De drie zichtbare camera's van de Meta Quest 2 zijn in rood omcirkeld en de vierde camera, die niet zichtbaar is, is als een vierkant aangeduid in Figuur 3.1.



Figuur 3.1: Meta Quest 2 [37]

De optie om de ingebouwde camera's van een VR-bril te gebruiken werd daarom verworpen voordat een vergelijking tussen de verschillende hardware-opties kon worden gemaakt.

Daarnaast werden de actieve *data gloves*, met een prijskaartje van €4499 exclusief btw, ook afgewezen vanwege hun gebrek aan kostenefficiëntie. Hoewel deze handschoenen zeer nauwkeurige resultaten kunnen leveren, wegen de hoge kosten niet op tegen de voordelen. Aangezien de SMOG-gebarentaal een niche betreft met beperkte financiering, zijn de beschikbare middelen beperkt en is de prijs een belangrijke factor om rekening mee te houden. De prijs is te hoog in vergelijking met andere beschikbare hardware die een betere balans biedt tussen kosten en nauwkeurigheid.

Hierdoor werd de focus verlegd naar de andere beschikbare hardware-opties die mogelijk een betere balans bieden tussen kosten en nauwkeurigheid. Dit gaf de mogelijkheid om een meer gedetailleerde evaluatie uit te voeren van de passieve *data gloves*, de RGB- en dieptecamera's, de Microsoft Kinect en de Leap Motion Controller, om te bepalen welke optie het meest geschikt zou zijn voor deze masterproef.

3.1.2 Vergelijking hardware

De overgebleven alternatieven uit sectie 3.1.1 worden onderworpen aan een vergelijking op verschillende criteria. Deze criteria, in willekeurige volgorde van belangrijkheid, omvatten: geschiktheid voor VR, kostenefficiëntie, gebruiksvriendelijkheid, realtime verwerking en haalbaarheid van implementatie binnen de beperkte tijdsperiode.

Voor een helder begrip van de beoordelingsmatrix in Tabel 3.1, volgt eerst een nadere toelichting van de genoemde criteria. 'Geschiktheid voor VR' verwijst naar de mate waarin de hardware kan worden geïntegreerd in een VR-omgeving, gezien het doel is om deze technologie aan te wenden in een interactief spel in VR. Bij gebrek aan compatibiliteit met VR zou het verdere onderzoek

onbruikbaar zijn voor dit doeleinde. 'Kostenefficiëntie' spreekt voor zichzelf en wijst op de kostprijs van de hardware. 'Gebruiksvriendelijkheid' geeft aan in welke mate de hardware intuïtief te gebruiken is, zelfs voor gebruikers met beperkte technische kennis en of het noodzakelijk is om speciale apparatuur aan te kopen. 'Realtime' verwijst naar de snelheid waarmee de inkomende data wordt verwerkt. Snelle en realtime feedback over de nauwkeurigheid van uitgevoerde handelingen is van essentieel belang. Tot slot, 'haalbaarheid' is een subjectieve evaluatie gebaseerd op de eerder genoemde criteria, inclusief de beschikbaarheid van voldoende onderzoeksinteresse en de bestaande open source-projecten of SDK's.

	geschikt voor VR	**kostprijs**	**gebruiksvriendelijkheid**	**realtime**	**haalbaarheid**
passieve *data gloves*	in combinatie met camera	goedkoop	ja	ja	-
smartphone-camera (RGB-camera)	niet rechtstreeks	geen extra kost	ja	ja	+
webcam laptop (RGB-camera)	kabel naar VR-bril	geen extra kost	ja	ja	+
dieptecamera	kabel naar VR-bril	≈ €350	ja	ja	+
Microsoft Kinect	minder	€18,99	ja	ja	-
Leap Motion Controller	ja	≈ €130	ja	ja	+

Tabel 3.1: Beoordelingsmatrix van de hardware

De gegevens uit Tabel 3.1 worden hieronder nader toegelicht. Passieve *data gloves* vereisen een externe camera voor gebruik, aangezien de ingebouwde camera's van VR-brillen niet kunnen worden benut. Hoewel dit alternatief kostenefficiënt is vanwege de lage aanschafkosten van handschoenen, vereist het een extra inspanning voor implementatie, gezien de noodzaak van aangepaste algoritmen omdat het finetunen van modellen voor dergelijke handschoenen een grotere aanpassing vereist en een grotere overdrachtskloof heeft dan andere modellen. Moderne implementaties gebruiken skeletmodellen, maar hedendaagse *machine learning*-algoritmen presteren beter met ruwe camerabeelden van handbewegingen dan op skeletmodellen. Bijgevolg wordt dit als een minder haalbare optie beschouwd.

De toepassing van een smartphonecamera in VR vereist een tussenliggend apparaat voor gegevensoverdracht, zoals draadloze verbindingen via Wi-Fi of bluetooth. Aangezien smartphones alomtegenwoordig zijn, brengt dit doorgaans geen extra kosten met zich mee, waardoor de implementatie haalbaar en praktisch mogelijk is.

Een laptop kan via een kabel verbonden worden met een VR-bril, waardoor webcamgegevens kunnen worden overgedragen en deze optie geschikt is voor VR-toepassingen. Net als bij smartphonecamera's brengt de wijdverbreide beschikbaarheid van laptops doorgaans geen extra kosten met zich mee, waardoor dit eveneens een praktisch en haalbaar alternatief is.

Intel's dieptecamera's kunnen worden verbonden met een VR-bril via een kabel voor gegevensoverdracht, wat een geschikte optie biedt voor VR-gebruik. Hoewel de initiële investering in deze camera's hoog kan zijn, kan deze dieptecamera gerechtvaardigd worden door de kwaliteit van de resultaten, waardoor het een haalbare optie blijft.

De Microsoft Kinect wordt beschouwd als minder geschikt voor VR-toepassingen, voornamelijk vanwege de bekendmaking op de officiële website van Microsoft dat de productie ervan is gestaakt, waarbij de opvolgende generatie dieptesensoren vertegenwoordigd wordt door de Azure Kinect DK [38]. De Azure Kinect DK is echter momenteel niet beschikbaar, waardoor deze sensor niet als een optie kan worden beschouwd. Bovendien richt de functionaliteit van de Microsoft Kinect zich hoofdzakelijk op het tracken van het volledige lichaam, wat resulteert in beperkte precisie bij het detecteren van specifieke hand- en vingerbewegingen. Hoewel de Kinect goedkoop is, wordt dit voordeel tenietgedaan door het feit dat dit alternatief niet erg geschikt is voor VR-toepassingen. Dit creëert een uitdaging bij de implementatie ervan, wat de haalbaarheid van dit alternatief bemoeilijkt.

Tot slot is de Leap Motion Controller, met uitgebreide documentatie voor integratie met VR-toepassingen, een veelbelovende optie. Hoewel de kosten gerechtvaardigd kunnen worden door goede resultaten, dient de efficiëntie van deze controller zorgvuldig te worden overwogen voor implementatie in VR-systemen.

3.1.3 Gekozen hardware

Door een samenvatting van de eigenschappen van alle hardware in Tabel 3.1 te maken, kan worden vastgesteld welke opties niet langer in overweging worden genomen en welke verder worden onderzocht. De passieve *data gloves* worden uitgesloten en eveneens wordt de Microsoft Kinect niet gebruikt vanwege de mogelijke beperkte nauwkeurigheid bij het volgen van handbewegingen aangezien deze sensor voor het volledige lichaam is geoptimaliseerd. Bijgevolg worden alle opties verworpen die een uitdaging met zich meebrengen om te implementeren.

De geselecteerde hardware om verder te onderzoeken omvat een smartphonecamera, de webcam van een laptop, een dieptecamera en de Leap Motion Controller. Deze alternatieven worden gekozen vanwege hun goede haalbaarheid, gebruiksvriendelijkheid en realtime functionaliteit. Ondanks de hogere kosten van de dieptecamera, wordt besloten om ermee verder te gaan vanwege het mogelijke hoge niveau van nauwkeurigheid, wat de investering rechtvaardigt.

3.2 Dataset creëren

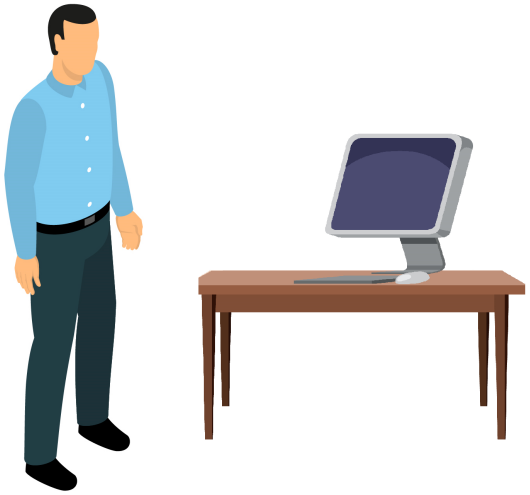
De volgende fase omvat de creatie van een dataset om een *machine learning*-model te trainen voor gebarentaalherkenning, aangezien er momenteel geen bestaande dataset beschikbaar is voor de SMOG-gebarentaal. Dit vormt een significante uitdaging binnen het gebied van gebarentaalherkenning, niet alleen vanwege de beperkte beschikbaarheid van data, maar ook vanwege de verscheidenheid aan dialecten. Niet alle dialecten hebben immers een uitgebreide dataset, wat

de uitdaging nog groter maakt. In het kader van dit onderzoek zullen zeven woorden worden geselecteerd binnen de SMOG-gebarentaal. Deze woorden zijn gekozen binnen het thema 'slaap-routine' en omvatten: aankleden, avond, knuffelbeer, slapen, tandenpoetsen, toilet en wassen. Het SMOG-gebaar voor 'toilet' is reeds geïllustreerd in Sectie 1.1, zoals weergegeven in Figuur 1.1.

3.2.1 Voorbereiding en opstelling

Aangezien de betekenis van de gebaren bepaald wordt door de aard van de beweging, eerder dan door een statische houding, was het noodzakelijk om video's op te nemen in plaats van foto's. In dit onderzoek is ervoor gekozen om met geïsoleerde invoer te werken, waarbij elk gebaar als een afzonderlijke video wordt vastgelegd, in plaats van een video waarin meerdere gebaren voorkomen. Voor de opnames is een opstelling ontwikkeld waarbij alle sensoren gelijktijdig de opgenomen gebaren kunnen registreren, zodat de uitvoerders deze gebaren zo min mogelijk moeten herhalen. Voordat deze opstelling in gebruik kon worden genomen, is er code geschreven om de data snel en eenvoudig te kunnen opnemen en achteraf te verwerken. Dit geldt voor de webcam van de laptop, de Intel RealSense dieptecamera D435f en de Leap Motion Controller 1.

Gezien de gebaren zich op schouderhoogte afspelen, zowel voor het gezicht als voor het lichaam, is besloten om de opname-opstelling voor de uitvoerder te plaatsen en niet ter hoogte van bijvoorbeeld waar een VR-bril zich bevindt. Dit is geïllustreerd in Figuur 3.2, waarin de computer de volledige opnameapparatuur vertegenwoordigt, bestaande uit de webcam van een laptop, een smartphonecamera, de Intel RealSense dieptecamera en de Leap Motion Controller.



Figuur 3.2: Opstelling voor opnames (globaal)

Na het schrijven van de code voor de opnames moest deze code uiteraard getest worden. Voor zowel de webcam als de dieptecamera verliepen de testen van de opnames probleemloos. De video's werden correct opgenomen met behulp van diverse functies uit de OpenCV-bibliotheek en keurig opgeslagen met een duidelijke naamgeving. Echter, bij het testen van de code voor de Leap Motion Controller in de opstelling van Figuur 3.2 ontstonden er problemen. Tijdens het schrijven van de code lag de Leap Motion Controller plat op tafel met de sensoren naar boven gericht (zie Figuur 3.3a), wat toen perfect functioneerde. Bij de opnames zou deze sensor

echter niet op deze manier geplaatst kunnen worden. De Leap Motion Controller zou verticaal geplaatst moeten worden met de sensoren naar de uitvoerder gericht in plaats van horizontaal (zie Figuur 3.3b). Bij verticale plaatsing bleek de Leap Motion Controller niet effectief in het tracken van handen. Hoewel de sensor af en toe iets detecteerde, was dit sporadisch en onvoldoende betrouwbaar voor consistente handtracking.

(a) Leap horizontaal geplaatst (b) Leap verticaal geplaatst

Figuur 3.3: Posities van Leap Motion Controller

Vanwege deze problemen met de Leap Motion Controller 1, is besloten deze sensor niet in de opstelling op te nemen. Ondanks het aanvankelijke vertrouwen in de Leap Motion Controller, omdat deze sensor speciaal voor handtracking is ontworpen, bleek deze sensor niet geschikt voor dit onderzoek.

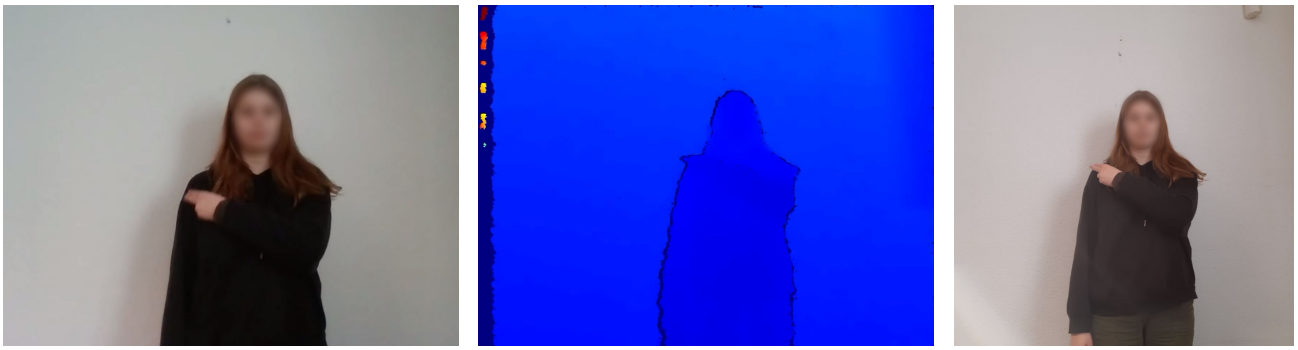
De uiteindelijke opstelling bestaat dus uit drie sensoren: de smartphonecamera van de Samsung Galaxy A50, de webcam van de HP laptop 15s-eq2047nb en de Intel RealSense dieptecamera D435f. De laptop werd op een verhoogde tafel geplaatst zodat de webcam zich ongeveer op schouderhoogte van de uitvoerder bevond. De smartphone werd tegen het scherm van de laptop geplaatst met het scherm naar de uitvoerder gericht, zodat de camera aan de voorkant kon worden gebruikt. De dieptecamera werd met plakband ter hoogte van het touchpad aan de laptop bevestigd. Hierdoor kunnen de drie sensoren gelijktijdig beelden vastleggen, waarbij de uitvoerder telkens op dezelfde plaats in beeld staat. Een detailweergave van de gebruikte opstelling is te zien in Figuur 3.4.

3.2.2 Opname dataset

Na het voltooien van de voorbereidingen en de opstelling, kon het daadwerkelijke opnemen van de dataset beginnen. In totaal hebben 13 individuen met een leeftijd tussen 18 en 24 jaar de gebaren uitgevoerd, waarvan zeven mannen en zes vrouwen. Voordat de deelnemers de gebaren uitbeeldden, kregen zij een video te zien waarin het gebaar werd gedemonstreerd door een professional. Hierdoor ontvingen alle deelnemers dezelfde instructies over hoe het gebaar moest worden uitgevoerd. Zodra duidelijk was hoe het gebaar correct moest worden uitgevoerd, werd elk gebaar tien keer opgenomen. De opnames met de smartphone werden handmatig bediend, waardoor er

Figuur 3.4: Opstelling voor opnames (detail)

per gebaar tien video's per persoon werden gemaakt. Een voorbeeld van deze opnames is te zien in Figuur 3.5.



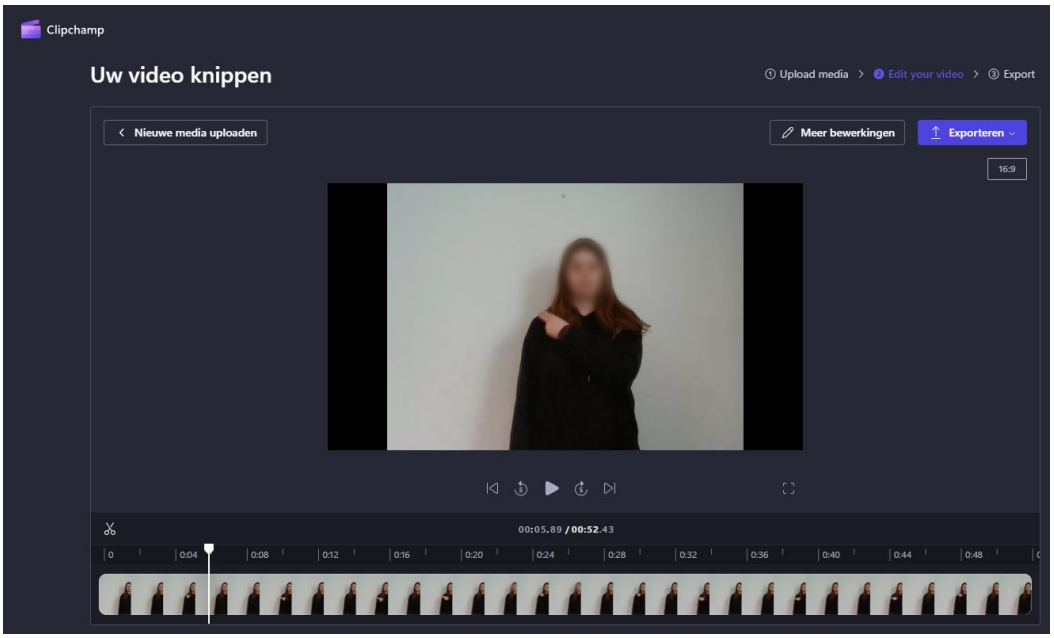
(a) Webcam (b) Dieptecamera (c) Smartphonecamera

Figuur 3.5: Opnames

Bij de eerste deelnemer werd opgemerkt dat het starten en stoppen van de opname niet altijd correct functioneerde, waardoor het gebaar soms werd uitgevoerd wanneer er geen opname plaatsvond, en omgekeerd. Om dit probleem te verhelpen en te voorkomen dat beeldmateriaal verloren ging, werd besloten om vanaf dan één doorlopende opname te maken voor zowel de webcam als de dieptecamera, waarin het gebaar tien keer werd uitgevoerd. Deze aanpak werd gekozen om de efficiëntie van het opnameproces te verbeteren.

Deze aanpak had echter het nadeel dat er na de opnames aanzienlijk meer tijd nodig was om de lange video's met tien gebaren te splitsen in afzonderlijke video's met telkens één gebaar. Het splitsen van de video's gebeurde met behulp van het programma Microsoft Clipchamp, zoals geïllustreerd in Figuur 3.6. De naamgeving van de video's moest eveneens handmatig worden gedaan. De video's werden als volgt gelabeld: een cijfer dat het gebaar in alfabetische volgorde aanduidt, gevolgd door een underscore, een tweecijferige volgnummer van de uitvoerder, nogmaals gevolgd door een underscore en een tweecijferig getal om aan te geven de hoeveelste uitvoering

dat is. Deze systematiek werd ook toegepast op de smartphone-opnames, die eveneens handmatig werden gelabeld.



Figuur 3.6: Video's knippen in Microsoft Clipchamp

Het is belangrijk op te merken dat de dataset enkele beperkingen kent. Er is steeds gebruik gemaakt van een witte achtergrond en enkel het bovenlichaam van de uitvoerders is in beeld gebracht. Alle deelnemers hebben een lichte huidskleur en zijn rond de 20 jaar, wat betekent dat het getrainde model mogelijk niet bij alle populaties goed zal werken. Deze beperkingen zijn echter bewust gekozen gezien de omvang van deze masterproef.

3.3 Model trainen

De laatste fase van dit onderzoek richt zich op het trainen van een *machine learning*-model. Nu de data volledig verzameld is, kan een model worden getraind om gebaren zo accuraat mogelijk te classificeren. Hierbij moet echter rekening worden gehouden met enkele belangrijke factoren. Ten eerste moet het model in staat zijn te leren van een beperkte hoeveelheid data, aangezien het niet haalbaar is om voor alle SMOG-gebaren tienduizenden of honderdduizenden opnames te verzamelen, wat bij traditionele *machine learning* nodig is. Daarnaast moet het model later geïmplementeerd kunnen worden in een VR-omgeving.

Uit de literatuur blijkt dat *transfer learning* een geschikte methode is voor het trainen van een model met een beperkte hoeveelheid data. Dit werkt enkel wanneer het domein van taak X, van het voorgetrainde model, dicht genoeg bij het domein van taak Y, het uiteindelijk doel, ligt. *Transfer learning* maakt gebruik van een voorgetraind model, zoals beschreven in Sectie 2.3.2, waardoor het eenvoudig schaalbaar is om nieuwe gebaren toe te voegen. Op basis hiervan is online gezocht naar beschikbare code voor *transfer learning* met voorgetrainde modellen. Uiteindelijk is gekozen voor een methode die gebruikmaakt van *Mobile Video Networks* (MoViNets) [39], wat in de volgende sectie nader wordt toegelicht.

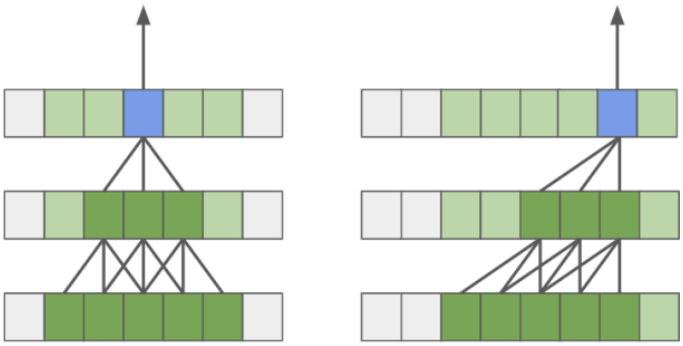
3.3.1 Mobile Video Networks (MoViNets)

MoViNets zijn efficiënte modellen voor videoclassificatie, ontwikkeld in 2021 door Kondratyuk *et al.* [39]. Deze netwerken zijn specifiek ontworpen om hoge nauwkeurigheid en efficiëntie te bereiken op mobiele apparaten, die vaak beperkt zijn in rekenkracht en geheugen. Dit maakt ze bijzonder geschikt voor implementatie in VR-omgevingen. MoViNets demonstreren een state-of-the-art nauwkeurigheid en efficiëntie op verschillende grootschalige datasets voor video-actieherkenning [39].

MoViNets zijn getraind met behulp van de Kinetics datasets, dit zijn veelgebruikte benchmarks voor het herkennen van acties zoals boogschieten of basketballen. Deze datasets bestaan telkens uit honderdduizenden video's, waarbij elke video ongeveer 10 seconden duurt. Kinetics 400, Kinetics 600 en Kinetics 700 bestaan respectievelijk uit 400, 600 en 700 verschillende klasses, Dit resulteert in een voorgetraind model dat verder kan worden verfijnd met eigen data.

Een classificatiemodel gebaseerd op 2D-frames kan eenvoudig worden toegepast op volledige video's. Echter, omdat dergelijke modellen geen rekening houden met het temporele aspect, hebben ze beperkte nauwkeurigheid en kunnen ze inconsistente resultaten tussen frames opleveren. Standaard 3D-convolutionele neurale netwerken maken gebruik van bidirectionele temporele context (kijken naar zowel het verleden als de toekomst), wat de nauwkeurigheid en temporele consistentie kan vergroten. Deze netwerken vereisen echter meer rekenkracht en geheugen, en omdat ze toekomstige frames in overweging nemen, zijn ze niet geschikt voor realtime toepassingen.

De MoViNets-architectuur maakt daarom gebruik van causale 3D-convoluties langs de tijdsas, waarbij alleen naar het verleden wordt gekeken voor het herkennen van gebaren [39]. Dit maakt realtime verwerking mogelijk, wat essentieel is voor deze masterproef. Het verschil tussen standaard convolutie en causale convolutie is geïllustreerd in Figuur 3.7. Hier is te zien dat standaard convoluties frames in zowel het verleden als de toekomst in overweging nemen ten opzichte van het huidige frame (blauw). Bij causale convoluties wordt daarentegen enkel naar frames in het verleden gekeken.



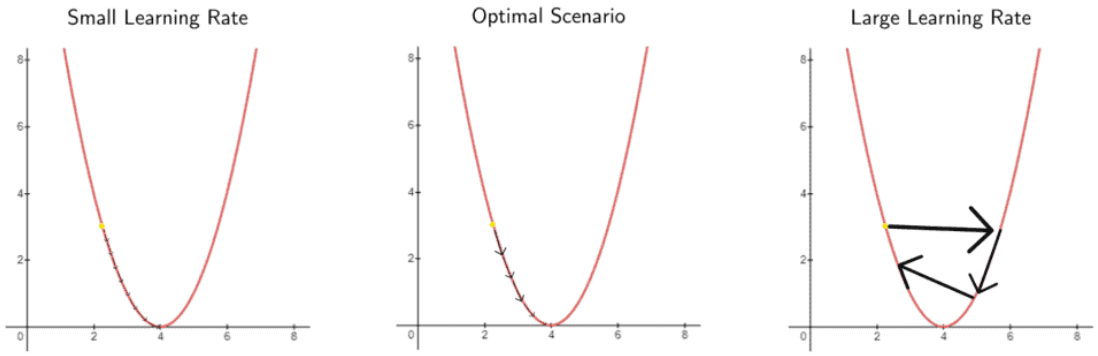
Figuur 3.7: Standaard convolutie (links) en causale convolutie (rechts) [39]

MoViNets maken uitsluitend gebruik van RGB-beelden, wat betekent dat de gegevens van de dieptecamera niet worden gebruikt. Dit impliceert dat het model alleen wordt getraind met data afkomstig van de smartphonecamera en de webcam. Het gebruik van enkel RGB-beelden kan een beperking vormen, aangezien dieptecamera's aanvullende informatie bieden over afstanden en dieptes van objecten.

3.3.2 Voorgetraind model

Het voorgetrainde MoViNets-model werd gebruikt voor verdere training met eigen data. Hierbij werden de convolutionele basis en bijbehorende parameters bevroren, terwijl enkel de laatste laag, de classificatielaag, opnieuw werd getraind om het trainingsproces te versnellen. De training van het model werd uitgevoerd met een dataset die losstaat van de testdata, waarbij telkens verschillende video's werden gebruikt. De dataset is gesplitst in een trainingsset en een testset op basis van personen, waarbij video's van eenzelfde persoon uitsluitend in de trainingsset of in de testset werden geplaatst. De oorspronkelijke parameters van de gekozen methode werden behouden voor de initiële modeltraining om te evalueren of het model voldoende nauwkeurigheid kon bereiken op de testdata. Dit bleek succesvol, met een nauwkeurigheid van al 82,9% voor het correct classificeren van de gebaren met de webcam, waarna verschillende parameters werden gevarieerd om te onderzoeken hoe de nauwkeurigheid verder kon worden verbeterd.

De gevarieerde hyperparameters omvatten het aantal epochs, de learning rate en de *optimizer*. Een andere parameter is de hoeveelheid gebruikte data voor zowel training als testen. Het standaard aantal epochs was 2 dus daarom werd gekozen om deze parameter zowel te halveren naar 1 als te verdubbelen naar 4 om het effect op de nauwkeurigheid te evalueren. Daarnaast speelt de hoeveelheid gebruikte data een cruciale rol. In eerste instantie werd gekozen om de maximale datahoeveelheid van 20 trainvideo's en 6 testvideo's per klasse te gebruiken, om zo de verhouding van 75% trainen en 25% testen te behouden. Deze aantallen werden gekozen omdat er van elke uitvoerder, in totaal 13, twee video's werden geselecteerd, resulterend in 26 video's per klasse. Van deze 26 video's werden de video's van 10 personen gebruikt voor training en de video's van de overgebleven 3 personen voor het testen. Er werd onderzocht of de nauwkeurigheid hoog bleef bij minder data.



Figuur 3.8: Invloed van learning rate [40]

De learning rate kan ook een significante invloed hebben op de nauwkeurigheid. Een te kleine learning rate kan het trainingsproces aanzienlijk vertragen (zie Figuur 3.8 links), terwijl een te grote learning rate het model kan verhinderen te convergeren (zie Figuur 3.8 rechts). Daarom werd ervoor gekozen deze waarde zowel 10 keer groter als 10 keer kleiner te maken. Ten slotte kan de keuze van *optimizer* het trainen beïnvloeden, aangezien elke *optimizer* zijn eigen methodiek heeft. Adam, een veelgebruikte *optimizer*, werd toegepast in de oorspronkelijke code. Daarnaast werden twee andere *optimizers* getest: *Root Mean Squared Propagation* (RMSprop) en *Stochastic Gradient Descent* (SGD).

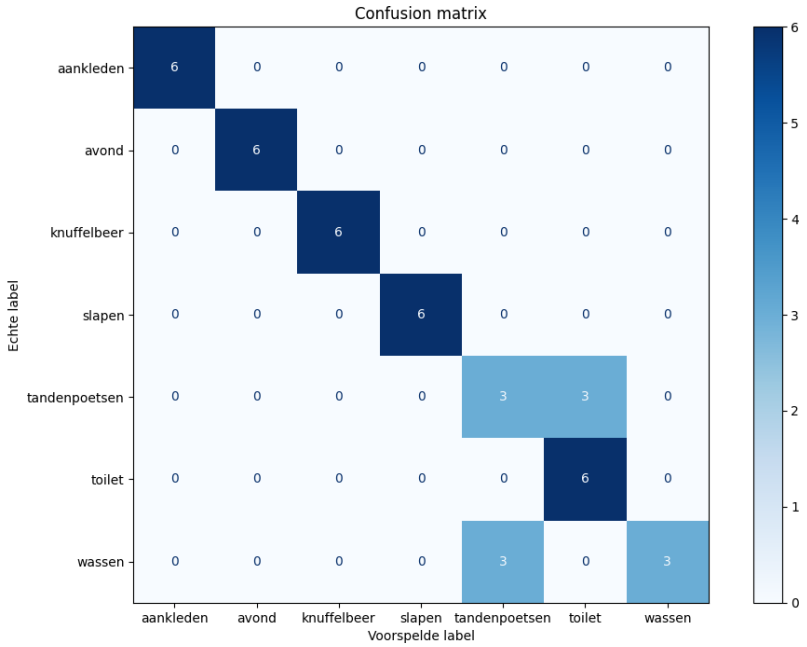
Hoofdstuk 4

Resultaten

In dit hoofdstuk worden de resultaten gepresenteerd, verdeeld in drie secties: één over het referentiemodel van de webcam, één over het referentiemodel van de smartphonecamera en daarna een sectie waar de resultaten worden weergegeven per parameter die de nauwkeurigheid beïnvloedt.

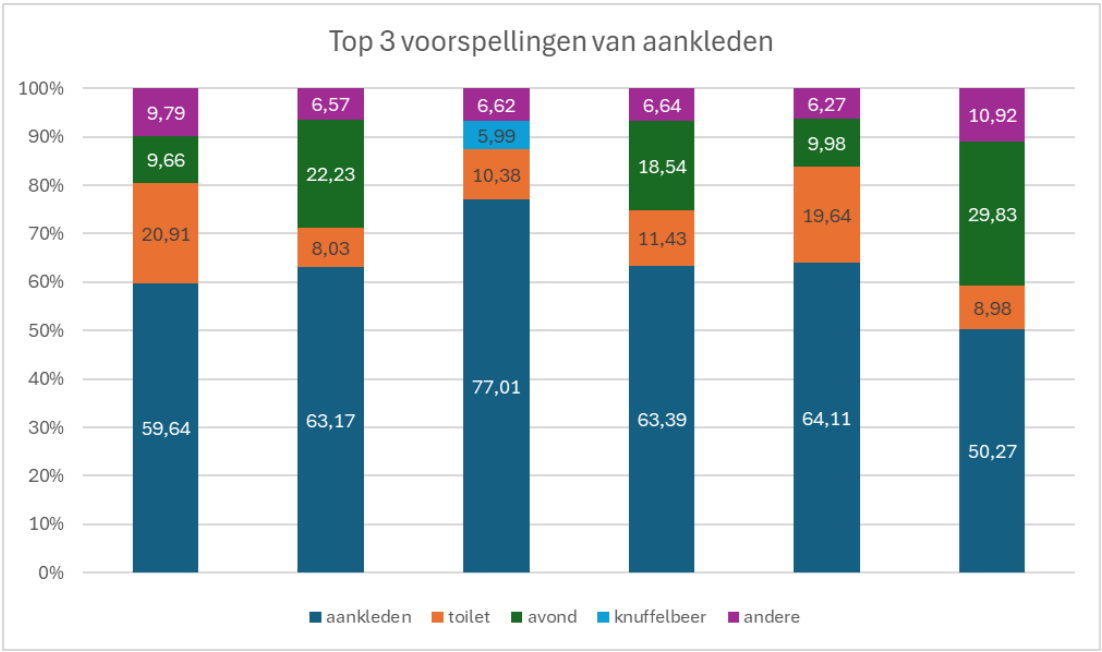
4.1 Webcam

Het initiële model werd getraind met 2 epochs, gebruikmakend van de maximale beschikbare dataset bestaande uit 20 video's per klasse voor training en 6 video's per klasse voor testen. De *learning rate* was ingesteld op 0,001 en de Adam-*optimizer* werd toegepast. Dit leidde tot een nauwkeurigheid van 82,9% met een *loss* van 0,593. Deze resultaten dienen ter referentie voor de evaluatie van alle volgende modellen die gebruik maken van webcamdata. De visualisatie van deze resultaten wordt gepresenteerd in de vorm van een *confusion matrix* (zie Figuur 4.1).

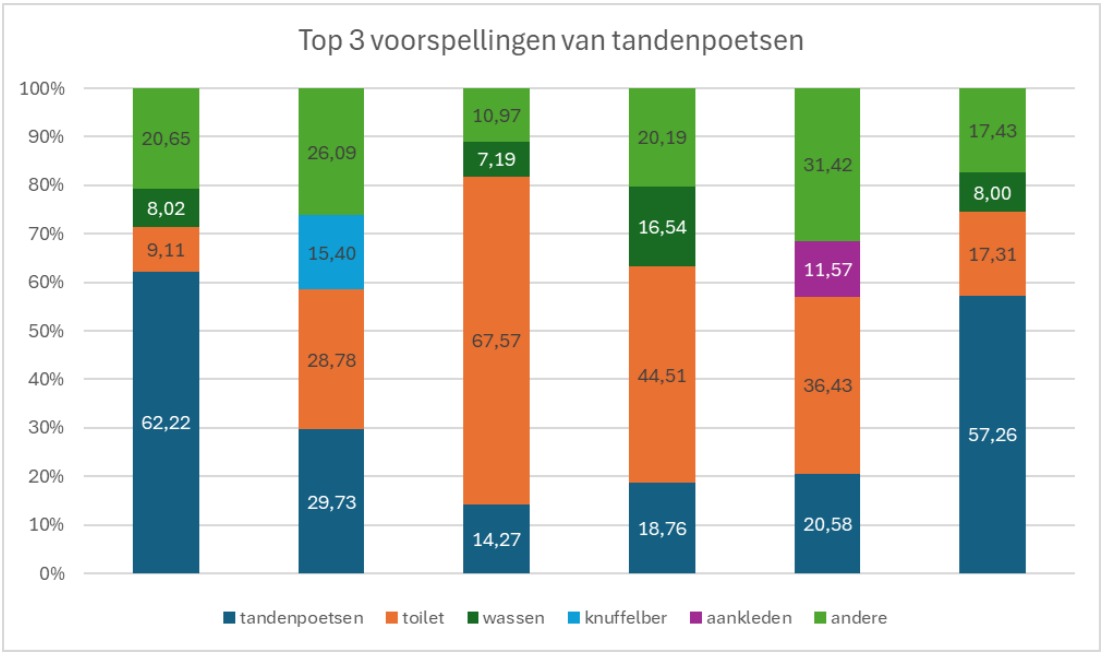


Figuur 4.1: *Confusion matrix* van referentiemodel (webcam)

Figuur 4.2 toont hoe het referentiemodel van de webcam de zes testvideo's van de klasse 'aankleden' classificeert binnen de top drie voorspellingen, inclusief de bijbehorende zekerheidspercentages. Onder 'andere' vallen de overige gebaren die door het model als vierde tot en met zevende zijn geïdentificeerd. In Figuur 4.3 wordt daarentegen de classificatie voor de zes testvideo's van de klasse 'tandenpoetsen' weergegeven. De categorie 'aankleden' is geselecteerd omdat, volgens de *confusion matrix* in Figuur 4.1, alle voorspellingen correct zijn geclassificeerd. Daarentegen is 'tandenpoetsen' als een ander voorbeeld gekozen, vanwege de aanwezigheid van fouten in de classificatie bij dit gebaar.



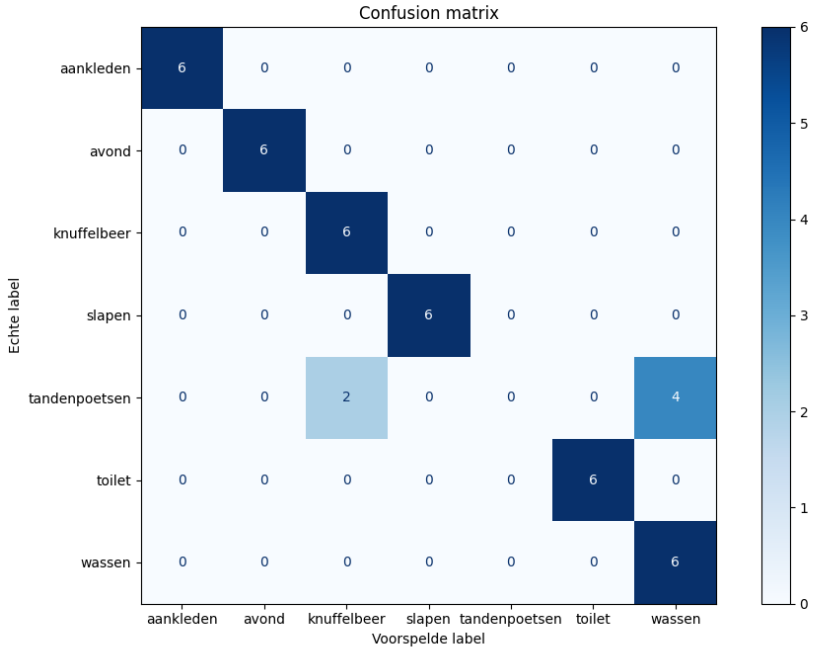
Figuur 4.2: Voorspelling van 'aankleden' bij het referentiemodel (webcam)



Figuur 4.3: Voorspelling van 'tandenpoetsen' bij het referentiemodel (webcam)

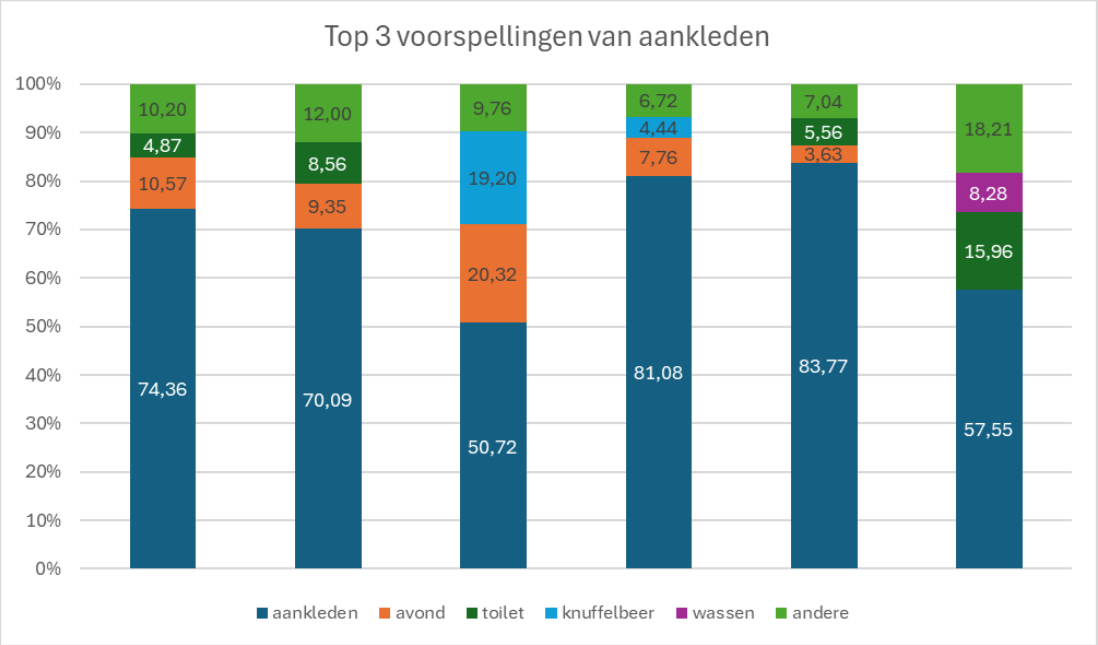
4.2 Smartphonecamera

Het initiële model werd wederom getraind met 2 epochs, gebruikmakend van de maximale beschikbare dataset bestaande uit 20 video's per klasse voor training en 6 video's per klasse voor testen. De *learning rate* was ingesteld op 0,001 en de Adam-*optimizer* werd toegepast. Dit resulteerde in een nauwkeurigheid van 84,3% met een bijbehorende *loss* van 0,591. Deze resultaten dienen ter referentie voor de evaluatie van alle volgende modellen die gebruik maken van de smartphonecamera. Een visuele representatie van deze bevindingen wordt gepresenteerd in de vorm van een *confusion matrix* (zie Figuur 4.4).

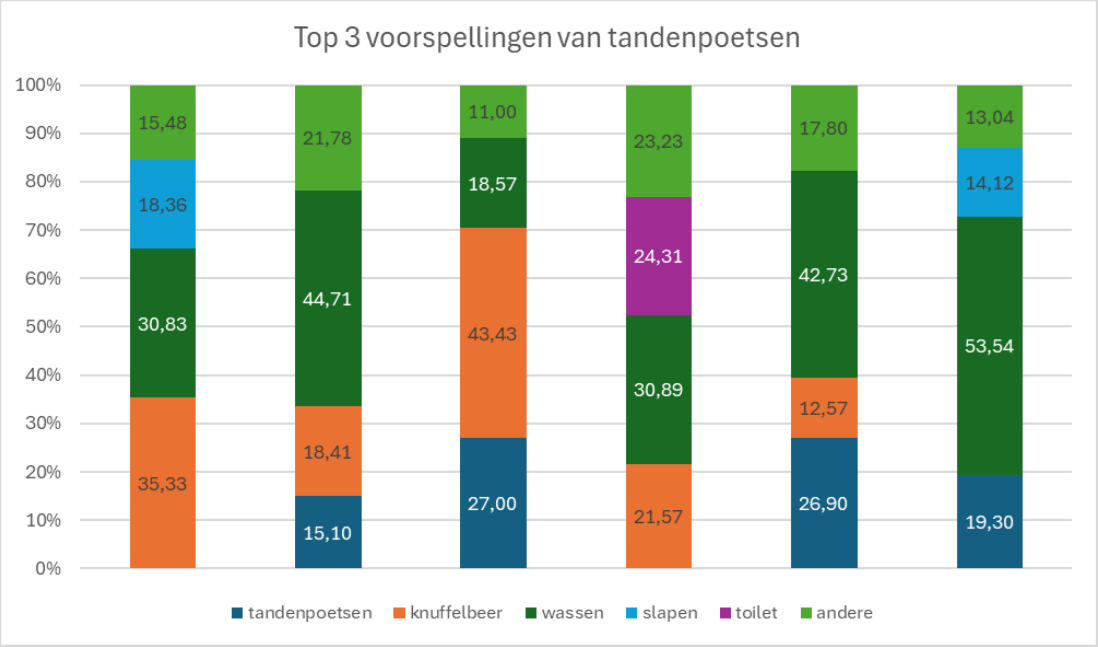


Figuur 4.4: *Confusion matrix* van referentiemodel (smartphonecamera)

Figuur 4.5 toont hoe het referentiemodel van de smartphonecamera de zes testvideo's van de klasse 'aankleden' classificeert binnen de top drie voorspellingen, inclusief de bijbehorende zekerheidspercentages. De categorie 'andere' bevat gebaren die door het model als vierde tot en met zevende zijn geïdentificeerd. Figuur 4.6 daarentegen, presenteert de classificatie voor de zes testvideo's van de klasse 'tandenpoetsen'. De keuze voor de categorie 'aankleden' is gebaseerd op de *confusion matrix* in Figuur 4.4, waaruit blijkt dat alle voorspellingen correct zijn geclassificeerd. 'Tandenpoetsen' is als voorbeeld gekozen vanwege de aanwezige fouten in de classificatie bij deze categorie.



Figuur 4.5: Voorspelling van 'aankleden' bij het referentiemodel (smartphonecamera)



Figuur 4.6: Voorspelling van 'tandenpoetsen' bij het referentiemodel (smartphonecamera)

4.3 Invloed van parameters

4.3.1 Invloed van aantal epochs

In Tabel 4.1 worden de resultaten gepresenteerd van het trainen van het model met verschillende aantallen epochs voor de webcam en in Tabel 4.2 voor de smartphonecamera. Het referentiemodel is getraind met 2 epochs. Met loss wordt altijd de testloss bedoeld. De nauwkeurigheid van de webcam bij 1 epoch bedraagt 47,9% met een *loss* van 1,158, terwijl deze waarden respectievelijk 95,0% en 0,270 zijn bij 4 epochs. De nauwkeurigheid van de smartphonecamera bij 1 epoch bedraagt 53,6% met een *loss* van 0,973, terwijl deze waarden respectievelijk 99,3% en 0,239 zijn

bij 4 epochs.

	1 epoch	**2 epochs**	4 epochs
nauwkeurigheid (%)	47,9	82,9	95,0
loss	1,158	0,593	0,270

Tabel 4.1: Invloed van het aantal epochs (webcam)

	1 epoch	**2 epochs**	4 epochs
nauwkeurigheid (%)	53,6	84,3	99,3
loss	0,973	0,591	0,239

Tabel 4.2: Invloed van het aantal epochs (smartphonecamera)

4.3.2 Invloed van hoeveelheid data

In Tabel 4.3 worden de resultaten weergegeven van het trainen van het model met diverse hoeveelheden data voor de webcam en in Tabel 4.4 van de smartphonecamera. Het referentiemodel is getraind met 20 trainvideo's en 6 testvideo's per klasse. Bij het verminderen van het aantal trainvideo's naar 10 en het aantal testvideo's naar 3 per klasse, werd een nauwkeurigheid van 72,9% bereikt, met een *loss* van 1,274 bij de webcam. Daarentegen leiden 15 trainvideo's en 5 testvideo's per klasse bij de webcam tot een respectievelijke nauwkeurigheid van 77,1% en *loss* van 0,844. Voor de smartphonecamera bedraagt de nauwkeurigheid 78,6% met een *loss* van 1,669 bij 10 trainvideo's en 3 testvideo's per klasse. Bij 15 trainvideo's en 5 testvideo's per klasse zijn deze waarden respectievelijk 85,7% en 0,805.

	10 train en 3 test	15 train en 5 test	**20 train en 6 test**
nauwkeurigheid (%)	72,9	77,1	82,9
loss	1,274	0,844	0,593

Tabel 4.3: Invloed van de hoeveelheid data per klasse (webcam)

	10 train en 3 test	15 train en 5 test	**20 train en 6 test**
nauwkeurigheid (%)	78,6	85,7	84,3
loss	1,669	0,805	0,591

Tabel 4.4: Invloed van de hoeveelheid data per klasse (smartphonecamera)

4.3.3 Invloed van grootte learning rate

In Tabel 4.5 worden de resultaten gepresenteerd van het trainen van het model met verschillende groottes van de *learning rate* voor de webcam en in Tabel 4.6 van de smartphonecamera. Het referentiemodel is getraind met een *learning rate* van 0,001. Het verlagen van de *learning rate* bij de webcam met een factor van 10, dus naar 0,0001, resulteerde in een nauwkeurigheid van

81,4% en een loss van 0,694. Daarentegen leidde een verhoging van de *learning rate* met een factor van 10, naar 0,01, tot een nauwkeurigheid van 87,4% en een loss van 0,648. Bij de smartphonecamera resulteerde het verlagen van de *learning rate* met een factor van 10, dus naar 0,0001, in een nauwkeurigheid van 91,4% en een loss van 0,493. Een verhoging van de *learning rate* met een factor van 10, naar 0,01, leidde tot een nauwkeurigheid van 89,3% en een loss van 0,500.

	learning rate = 0,0001	***learning rate* = 0,001**	*learning rate* = 0,01
nauwkeurigheid (%)	81,4	82,9	87,4
loss	0,694	0,593	0,648

Tabel 4.5: Invloed van de grootte van de *learning rate* (webcam)

	learning rate = 0,0001	***learning rate* = 0,001**	*learning rate* = 0,01
nauwkeurigheid (%)	91,4	84,3	89,3
loss	0,493	0,591	0,500

Tabel 4.6: Invloed van de grootte van de *learning rate* (smartphonecamera)

4.3.4 Invloed van optimizer

In Tabel 4.7 worden de resultaten weergegeven van het trainen van het model met diverse *optimizers* bij de webcam en voor de smartphonecamera in Tabel 4.8. Het referentiemodel is getraind met Adam als *optimizer*. Bij het gebruik van RMSprop als *optimizer* voor de webcam bereikt het model een nauwkeurigheid van 82,9% en een loss van 0,661, terwijl deze waarden respectievelijk 47,9% en 1,914 bedragen bij het gebruik van SGD. Bij het gebruik van RMSprop als *optimizer* voor de smartphonecamera bereikt het model een nauwkeurigheid van 92,1% en een loss van 0,681, terwijl deze waarden respectievelijk 47,9% en 1,866 bedragen bij het gebruik van SGD.

	Adam	RMSprop	SGD
nauwkeurigheid (%)	82,9	82,9	47,9
loss	0,593	0,661	1,914

Tabel 4.7: Invloed van de *optimizer* (webcam)

	Adam	RMSprop	SGD
nauwkeurigheid (%)	84,3	92,1	47,9
loss	0,591	0,681	1,866

Tabel 4.8: Invloed van de *optimizer* (smartphonecamera)

Hoofdstuk 5

Discussie

5.1 Invloed van parameters

5.1.1 Invloed van aantal epochs

Op basis van de waarnemingen bij de webcam blijkt dat het verhogen van het aantal epochs leidt tot een stijging in nauwkeurigheid en een daling in *loss*. Specifiek, bij 4 epochs bij de webcam bereikt het model een nauwkeurigheid van 95,0%, aanzienlijk hoger dan de 47,9% bij slechts 1 epoch. De gebruikte *loss*-functie berekent het logaritmisch verschil tussen de labels en de voorspellingen, en toont een daling in *loss* van 1,158 bij 1 epoch naar 0,270 bij 4 epochs.

Eenzelfde patroon is zichtbaar bij de smartphonecamera. Hier stijgt de nauwkeurigheid ook met het aantal epochs, waarbij het model bij 4 epochs een nauwkeurigheid van 99,3% bereikt met een *loss* van slechts 0,239, tegenover een nauwkeurigheid van 53,6% en een *loss* van 0,973 bij 1 epoch.

Uit deze resultaten blijkt dat het aantal epochs, gebruikt tijdens de training, een kritische hyperparameter is die de prestaties van het model significant beïnvloedt. Bij 1 epoch is het duidelijk dat het model onvoldoende tijd heeft om de onderliggende patronen in de data te leren, wat resulteert in underfitting. Dit verklaart de lage nauwkeurigheid en hoge loss. Naarmate het aantal epochs toeneemt, verbetert het model aanzienlijk. Echter, bij een aantal epochs hoger dan 4 bestaat de mogelijkheid dat het model gaat overfitten. Overfitting treedt op wanneer een model te sterk gefocust raakt op de trainingsdata, waardoor het niet goed generaliseert naar nieuwe, ongeziene data en daardoor slechter presteert op de testset.

Het gebruik van een voorgetraind model impliceert dat het model reeds is getraind op een uitgebreide dataset, wat suggereert dat het model algemene patronen heeft leren kenen. Als gevolg hiervan wordt de kans op overfitting gereduceerd. Bovendien wordt het risico op overmatig aanpassen aan de trainingsdata verder beperkt door het beperkte aantal epochs, waardoor het model slechts een beperkte tijd heeft om zich aan te passen aan de specifieke kenmerken van de trainingsdata.

Deze bevindingen benadrukken het belang van een zorgvuldige afstemming van het aantal epochs om een balans te vinden tussen voldoende training en het vermijden van overfitting.

5.1.2 Invloed van hoeveelheid data

Uit de resultaten blijkt dat de webcam bij 10 trainingsvideo's en 3 testvideo's per klasse een nauwkeurigheid van 72,9% behaalt. Deze nauwkeurigheid stijgt naar 77,1% bij 15 trainingsvideo's en 5 testvideo's per klasse, en vervolgens verder naar 82,9% bij 20 trainingsvideo's en 6 testvideo's per klasse. De *loss* daarentegen daalt van 1,274 bij 10 trainingsvideo's en 3 testvideo's per klasse naar 0,593 bij 20 trainingsvideo's en 6 testvideo's per klasse, wat wijst op een verbetering van de prestaties naarmate er meer data wordt gebruikt voor het trainen en testen.

Bij de smartphonecamera wordt een nauwkeurigheid van 78,6% behaald met 10 trainingsvideo's en 3 testvideo's per klasse, die stijgt naar 84,3% bij 20 trainingsvideo's en 3 testvideo's per klasse. Bij 15 trainingsvideo's en 5 testvideo's per klasse bereikt het model de hoogste nauwkeurigheid van 85,7%, wat wijst op de afwezigheid van een consistent patroon voor de nauwkeurigheid. De *loss* neemt af van 1,669 bij 10 trainingsvideo's en 3 testvideo's per klasse naar 0,591 bij 20 trainingsvideo's en 6 testvideo's per klasse. Dit wijst op een verbetering van de modelprestaties met toenemende hoeveelheid data.

Deze bevindingen impliceren dat een grotere dataset niet noodzakelijk leidt tot een hogere nauwkeurigheid, hoewel dit wel verwacht zou worden. Een grotere dataset verkleint echter de kans op overfitting. Voor de webcam is een grotere dataset gelinkt aan een hogere nauwkeurigheid, terwijl dit voor de smartphonecamera niet consistent is. Wel neemt de loss af naarmate de hoeveelheid data toeneemt, zowel voor de webcam als de smartphonecamera, wat duidt op een betere generalisatie van de modellen bij gebruik van grotere datasets.

5.1.3 Invloed van grootte learning rate

Bij de webcam bedraagt de nauwkeurigheid 81,4% bij een *learning rate* van 0,0001. Deze nauwkeurigheid neemt toe bij een grotere *learning rate*, van 82,9% bij een *learning rate* van 0,001 naar 87,4% bij een *learning rate* van 0,01. De *loss* bereikt zijn laagste waarde van 0,593 bij een *learning rate* van 0,001. Zowel bij een lagere als bij een hogere *learning rate* is de *loss* hoger, respectievelijk 0,694 bij een *learning rate* van 0,0001 en 0,648 bij een *learning rate* van 0,01. Dit toont aan dat hier de hoogste nauwkeurigheid niet samenvalt met de laagste *loss*.

Bij de smartphonecamera is er geen duidelijk verband tussen de *learning rate* en de nauwkeurigheid, noch tussen de *learning rate* en de *loss* van het model. De nauwkeurigheid daalt van 91,4% bij een *learning rate* van 0,0001 naar 84,3% bij een *learning rate* van 0,001 en stijgt vervolgens weer naar 89,3% bij een *learning rate* van 0,01. De *loss* vertoont een omgekeerd patroon ten opzichte van de nauwkeurigheid: de *loss* stijgt van 0,493 bij een *learning rate* van 0,0001 naar 0,591 bij een *learning rate* van 0,001, om vervolgens te dalen naar de waarde van 0,500 bij een *learning rate* van 0,01. Net als bij de webcam impliceert de hoogste nauwkeurigheid hier niet de laagste *loss*. Er bestaat echter een omgekeerd evenredig verband tussen de *loss* en de nauwkeurigheid, zoals verwacht mag worden.

De *learning rate* is een hyperparameter die de aanpassingen aan de interne parameters van het model bepaalt en een cruciale rol speelt bij het optimaliseren van de prestaties van een model. Een hogere *learning rate* betekent dat er grotere veranderingen in de parameters optreden, terwijl een lagere *learning rate* kleinere veranderingen impliceert. Uit deze resultaten blijkt dat er geen eenduidig verband is tussen de *learning rate* en de nauwkeurigheid van het model. Dit geldt ook voor de *loss*. Een te hoge *learning rate* kan resulteren in problemen bij het convergeren naar een

optimale oplossing. Aan de andere kant, als de *learning rate* te laag is, maakt het model slechts kleine aanpassingen, wat leidt tot een trage convergentie en een langere trainingstijd.

5.1.4 Invloed van optimizer

Uit de resultaten van de experimenten met de webcam blijkt dat zowel de Adam- als de RMSprop-*optimizer* een nauwkeurigheid van 82,9% behalen. Hierbij tekent Adam een *loss* van 0,593 op, terwijl RMSprop een iets hogere *loss* van 0,661 laat zien. De SGD-*optimizer* presteert aanzienlijk slechter, met een nauwkeurigheid van slechts 47,9% en een *loss* van 1,914. Hieruit kan duidelijk worden afgeleid dat SGD ongeschikt is als *optimizer* voor deze webcamdata. Adam toont zich iets effectiever dan RMSprop door de lagere *loss*.

Bij de smartphonecamera geeft de RMSprop-*optimizer* de beste nauwkeurigheid, met een waarde van 92,1% en een *loss* van 0,681. De Adam-*optimizer* behaalt een nauwkeurigheid van 84,3% en een *loss* van 0,591, terwijl SGD wederom teleurstelt met een nauwkeurigheid van 47,9% en een *loss* van 1,866. Dit bevestigt opnieuw dat SGD geen geschikte *optimizer* is voor deze modellen, terwijl zowel RMSprop als Adam wel goede prestaties leveren. In het geval van de smartphonecamera overtreft RMSprop zelfs de prestaties van Adam op het vlak van de nauwkeurigheid, terwijl Adam daarentegen een betere prestatie levert op het gebied van de *loss*.

Het voornaamste doel van een *optimizer* is om de *loss* te minimaliseren door de modelparameters iteratief aan te passen. De keuze van de *optimizer* kan de prestaties van het model aanzienlijk beïnvloeden en is daarom een cruciale beslissing. Elke *optimizer* heeft zijn eigen voordelen en beperkingen. Voor dit model blijkt dat SGD geen geschikte *optimizer* is. Zowel Adam als RMSprop presteren hier goed, waarbij RMSprop zelfs een lichte voorsprong heeft op Adam, voornamelijk bij de smartphonecamera.

5.2 Samengevat

In deze sectie worden de belangrijkste bevindingen van de voorgaande secties samengevat.

Wat betreft het aantal epochs, is het evident dat een toename in epochs doorgaans resulteert in een betere prestatie van het model. Echter, in dit specifieke scenario wordt vastgesteld dat het optimale aantal epochs beperkt is tot vier om overfitting te voorkomen.

Met betrekking tot de hoeveelheid beschikbare data blijkt uit de analyse dat voor de beste resultaten bij het gebruik van de webcam, 20 trainvideo's en 6 testvideo's per klasse optimaal zijn. Voor gebruik van de smartphonecamera varieert dit tussen 15 en 20 trainvideo's, en 5 of 6 testvideo's per klasse. Dit wordt gerechtvaardigd door het feit dat een hoger aantal train- en testvideo's essentieel is voor het behoorlijk trainen en evalueren van het model.

De keuze van de *learning rate* vertoont een duidelijk patroon bij het gebruik van de webcam, waarbij hogere waarden over het algemeen een gunstiger effect hebben op de prestaties van het model. Echter, bij waarden groter dan 0,01 neemt de *loss* toe in vergelijking met een *learning rate* van 0,001 dus dit is waarschijnlijk de maximale gunstige waarde voor de *learning rate*. Bij het gebruik van de smartphonecamera is het vinden van een optimale *learning rate* complexer, waarbij geen eenduidig patroon kan worden vastgesteld.

Wat betreft de *optimizer* wordt geconcludeerd dat voor deze toepassing de keuze niet moet vallen

op SGD, maar eerder op Adam of RMSprop. RMSprop vertoont zelfs op de nauwkeurigheid nog betere resultaten dan Adam bij het gebruik van de smartphonecamera.

Het optimale resultaat in termen van nauwkeurigheid en *loss* wordt bereikt bij vier epochs voor zowel de smartphonecamera als de webcam. Voor de smartphonecamera bedraagt de nauwkeurigheid dan 99,3% met een *loss* van 0,239, terwijl voor de webcam de nauwkeurigheid 95,0% bedraagt met een *loss* van 0,270.

Hoofdstuk 6

Conclusie

Deze masterproef heeft een grondige analyse uitgevoerd naar de meest geschikte sensoren voor het detecteren van gebaren, met als resultaat de laptopwebcam en de smartphonecamera. Vervolgens is er een uitgebreide dataset samengesteld, bestaande uit zeven SMOG-gebaren, vastgelegd door 13 verschillende uitvoerders. Door gebruik te maken van *transfer learning* in combinatie met MoViNets, is een *machine learning*-model ontwikkeld dat in staat is deze gebaren nauwkeurig te onderscheiden.

De resultaten van het ontwikkelde *machine learning*-model tonen aan dat het mogelijk is om gebruikers in een VR-omgeving te filmen, terwijl zij gebaren uitvoeren en deze gebaren realtime met een hoge nauwkeurigheid correct te classificeren. Deze masterproef geeft hiermee de aanzet tot het uitbouwen van een virtuele leeromgeving in combinatie met *immersive storytelling*. Met deze aanpak kunnen alle SMOG-gebaren geïntegreerd worden, zodat jongeren de gebaren spelenderwijs kunnen leren en er actief mee aan de slag kunnen gaan.

Een belangrijke aanbeveling voor toekomstig onderzoek is het uitbreiden van de dataset. Het is cruciaal om een meer diverse groep mensen op te nemen, inclusief verschillende leeftijdscategorieën en uiteenlopende achtergronden, tijdens het verzamelen van videomateriaal. Dit zal niet alleen de robuustheid en generaliseerbaarheid van het machine learning-model verbeteren, maar ook de inclusiviteit van de leeromgeving vergroten.

Referentielijst

[1] S. Vlaanderen, "Veelgestelde vragen." Beschikbaar: `https://smog.vlaanderen/faq`, 2024. Geopend op 23 februari 2024.

[2] F. Loncke, M. Nijs, and L. Smet, *SMOG - Spreken Met Ondersteuning van Gebaren*. Leuven: Garant, 2000.

[3] M. E. O. Technologie, "Communiceren met gebaren." Beschikbaar: `https://ikkannietpraten.be/hulpmiddelen/communiceren-met-gebaren`, 2024. Geopend op 11 april 2024.

[4] A. U. of Applied Sciences and A. Antwerp, "Game based learning and development of skills (glados)." Beschikbaar: `https://www.ap.be/en/research/game-based-learning-and-development-skills-glados`, 2024. Geopend op 16 mei 2024.

[5] J. Galván-Ruiz, C. M. Travieso-González, A. Tejera-Fettmilch, A. Pinan-Roescher, L. Esteban-Hernández, and L. Domínguez-Quintana, "Perspective and evolution of gesture recognition for sign language: A review," *Sensors*, vol. 20, no. 12, 2020. Art. no. 3571.

[6] R. Rastgoo, K. Kiani, and S. Escalera, "Sign language recognition: A deep survey," *Expert Systems with Applications*, vol. 164, 2021. Art. no. 113794.

[7] G. Buckingham, "Hand tracking for immersive virtual reality: Opportunities and challenges," *Frontiers in Virtual Reality*, vol. 2, 2021.

[8] C. Khundam, V. Vorachart, P. Preeyawongsakul, W. Hosap, and F. Noël, "A comparative study of interaction time and usability of using controllers and hand tracking in virtual reality training," *Informatics*, vol. 8, no. 3, 2021. Art. no. 60.

[9] S. Mitra and T. Acharya, "Gesture recognition: A survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 3, pp. 311–324, 2007.

[10] A. Wadhawan and P. Kumar, "Sign language recognition systems: A decade systematic literature review," *Archives of Computational Methods in Engineering*, vol. 28, no. 3, pp. 785–813, 2021.

[11] J. Wang, F. Mueller, F. Bernard, S. Sorli, O. Sotnychenko, N. Qian, M. A. Otaduy, D. Casas, and C. Theobalt, "Rgb2hands: real-time tracking of 3d hand interactions from monocular rgb video," *ACM Trans. Graph.*, vol. 39, no. 6, pp. 1–16, 2020. Art. no. 218.

[12] A. Ahmad, C. Migniot, and A. Dipanda, "Hand pose estimation and tracking in real and virtual interaction: A review," *Image and Vision Computing*, vol. 89, pp. 35–49, 2019.

[13] H. Cooper, B. Holt, and R. Bowden, *Sign Language Recognition*, pp. 539–562. London: Springer London, 2011.

[14] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," *Computer Vision and Image Understanding*, vol. 108, no. 1, pp. 52–73, 2007.

[15] B. A. Myers, "A brief history of human-computer interaction technology," *Interactions*, vol. 5, no. 2, p. 44–54, 1998.

[16] OpenCV, "Aruco marker detection (aruco module)." Beschikbaar: `https://docs.opencv.org/4.x/d9/d6d/tutorial_table_of_content_aruco.html`, 2024. Geopend op 10 april 2024.

[17] S. Ghosh, J. Zheng, W. Chen, J. Zhang, and Y. Cai, "Real-time 3d markerless multiple hand detection and tracking for human computer interaction applications," in *Proceedings of the 9th ACM SIGGRAPH Conference on Virtual-Reality Continuum and Its Applications in Industry*, VRCAI '10, (New York, NY, USA), p. 323–330, Association for Computing Machinery, 2010.

[18] R. M. Gurav and P. K. Kadbe, "Real time finger tracking and contour detection for gesture recognition using opencv," in *2015 International Conference on Industrial Instrumentation and Control (ICIC)*, pp. 974–977, 2015.

[19] M. Oudah, A. Al-Naji, and J. Chahl, "Hand gesture recognition based on computer vision: A review of techniques," *Journal of Imaging*, vol. 6, no. 8, 2020. Art. no. 73.

[20] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," *ACM Transactions on Graphics*, vol. 28, no. 3, pp. 1–8, 2009. Art. no. 63.

[21] M. Caeiro-Rodríguez, I. Otero-González, F. A. Mikic-Fonte, and M. Llamas-Nistal, "A systematic review of commercial smart gloves: Current status and applications," *Sensors*, vol. 21, no. 8, 2021. Art. no. 2667.

[22] SenseGlove, "Find out about our nova haptic glove." Beschikbaar: `https://www.senseglove.com/product/nova/`, 2024. Geopend op 13 april 2024.

[23] A. Choudhury, A. Talukdar, and K. Sarma, "A review on vision-based hand gesture recognition and applications," in *Intelligent Applications for Heterogeneous System Modeling and Design*, pp. 261–286, IGI Global, 2015.

[24] B. Liao, J. Li, Z. Ju, and G. Ouyang, "Hand gesture recognition with generalized hough transform and dc-cnn using realsense," in *2018 Eighth International Conference on Information Science and Technology (ICIST)*, 2018.

[25] GooXbox360.nl, "Kinect sensor xbox 360 - microsoft." Beschikbaar: `https://www.gooxbox360.nl/xbox-360-accessoires-kopen/microsoft-kinect/`, 2024. Geopend op 13 april 2024.

[26] J. Suarez and R. R. Murphy, "Hand gesture recognition with depth images: A review," in *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*, pp. 411–417, 2012.

[27] C. Xi, J. Chen, C. Zhao, Q. Pei, and L. Liu, "Real-time hand tracking using kinect," in *Proceedings of the 2nd International Conference on Digital Signal Processing*, p. 37–42, 2018.

[28] C. Dong, M. C. Leu, and Z. Yin, "American sign language alphabet recognition using microsoft kinect," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2015.

[29] H. Cheng, L. Yang, and Z. Liu, "Survey on 3d hand gesture recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1659–1673, 2016.

[30] J. Schioppo, Z. Meyer, D. Fabiano, and S. Canavan, "Sign language recognition: Learning american sign language in a virtual environment," in *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, p. 1–6, 2019.

[31] Ultraleap, "Leap motion controller 2." Beschikbaar: `https://leap2.ultraleap.com/leap-motion-controller-2/`, 2024. Geopend op 13 april 2024.

[32] I. Corporation, "Introducing the intel realsense depth camera d435f." Beschikbaar: `https://www.intelrealsense.com/depth-camera-d435f/`, 2024. Geopend op 13 april 2024.

[33] Oracle, "Wat is machine learning?." Beschikbaar: `https://www.oracle.com/nl/artificial-intelligence/machine-learning/what-is-machine-learning/`, 2024. Geopend op 7 juni 2024.

[34] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, pp. 1–34, 2020.

[35] D. Sarkar, "A comprehensive hands-on guide to transfer learning with real-world applications in deep learning." Beschikbaar: `https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in` 2018. Geopend op 10 mei 2024.

[36] A. Hosna, E. Merry, J. Gyalmo, Z. Alom, Z. Aung, and M. A. Azim, "Transfer learning: a friendly introduction," *J. Big Data*, vol. 9, 2022. Art. no. 102.

[37] Meta, "Je meta quest vergelijken." Beschikbaar: `https://www.meta.com/be/quest/compare/`, 2024. Geopend op 18 mei 2024.

[38] Microsoft, "Kinect for windows." Beschikbaar: `https://learn.microsoft.com/en-us/windows/apps/design/devices/kinect-for-windows`, 2022. Geopend op 20 mei 2024.

[39] D. Kondratyuk, L. Yuan, Y. Li, L. Zhang, M. Tan, M. Brown, and B. Gong, "Movinets: Mobile video networks for efficient video recognition," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16015–16025, 2021.

[40] S. Barreto, "Choosing a learning rate." Beschikbaar: `https://www.baeldung.com/cs/ml-learning-rate`, 2024. Geopend op 30 mei 2024.