UHASSELT
KNOWLEDGE IN ACTION

Maastricht University

Faculteit Wetenschappen
School voor Informatietechnologie

master in de informatica

Masterthesis

Real-time baking assistant

Gijs Konings

Scriptie ingediend tot het behalen van de graad van master in de informatica

PROMOTOR :

Prof. dr. Raf RAMAKERS

BEGELEIDER :

dr. Danny LEEN

2024
2025

De transnationale Universiteit Limburg is een uniek samenwerkingsverband van twee universiteiten in twee landen: de Universiteit Hasselt en Maastricht University.

UHASSELT
KNOWLEDGE IN ACTION

www.uhasselt.be
Universiteit Hasselt
Campus Hasselt:
Martelarenlaan 42 | 3500 Hasselt
Campus Diepenbeek:
Agoralaan Gebouw D | 3590 Diepenbeek

Maastricht University

Faculteit Wetenschappen
School voor Informatietechnologie

master in de informatica

Masterthesis

Real-time baking assistant

Gijs Konings

Scriptie ingediend tot het behalen van de graad van master in de informatica

PROMOTOR :

Prof. dr. Raf RAMAKERS

BEGELEIDER :

dr. Danny LEEN

Acknowledgement

First and foremost, I would like to express my heartfelt gratitude to my promoter Prof. Dr. Raf Ramakers for his assistance during my research. His insightful feedback and guidance throughout the process were irreplaceable. He was always kind and positive towards me and my research and his input made it much easier to achieve a higher quality of work.

I am also grateful to Dr. Danny Leen for sharing his expertise and providing valuable insights that enriched my research. His contribution of knowledge and enthusiasm for my thesis was very inspiring. His invaluable feedback was always given with a smile. And he played a crucial role in helping me overcome the challenges I encountered during my thesis.

Furthermore, I would like to thank the rest of the research group, including Dr. Tom Veuskens, Dr. Mannu Lambrichts, Dries Cardinael, Maties Claesen and Stig Konings. Together with Prof. Dr. Raf Ramakers and Dr. Danny Leen, they organized regular meetings that were extremely valuable to me. These meetings provided ideas and guidance that helped shape the direction of my research. The expertise and support of each member of the team greatly contributed to the quality and progress of my thesis.

I would also like to thank the Expertise Centre for Digital Media (EDM) at UHasselt for providing the necessary infrastructure, resources and equipment to conduct my research. The EDM created an environment where I could carry out my work efficiently and effectively.

Beyond academic support, I am also deeply grateful to my family and friends for their unwavering encouragement throughout my educational journey. Their belief in me, constant motivation and understanding during challenging times were crucial to completing my master's degree. Without their support, achieving this milestone would have been far more difficult.

Finally, I would like to thank all my fellow students who supported me during the making of this thesis. They were always willing to discuss ideas and provide feedback, showing genuine interest in the progress of my work. Their dedication and hard work on their own theses was a great source of motivation for me to carry on with my own work.

Dutch summary

Introductie

Deze thesis onderzoekt hoe large language models (LLM's) en computer vision kunnen worden benut om een interactief augmented reality (AR)-systeem te bouwen dat realtime leren ondersteunt door foutdetectie en adaptieve feedback. In tegenstelling tot traditionele stapsgewijze instructietools, interpreteert het voorgestelde systeem gebruikersacties, identificeert potentiële fouten en biedt contextbewuste, gepersonaliseerde feedback die leerlingen helpt niet alleen te begrijpen wat ze moeten doen, maar ook waarom.

Om dit te onderzoeken, werd een AR-gebaseerde prototype ontwikkeld met behulp van de Magic Leap 2-headset. Het systeem legt continu beelden vast vanuit het perspectief van de gebruiker, verwerkt deze vooraf en zendt ze door naar een LLM voor analyse. Het LLM interpreteert vervolgens de voortgang van de gebruiker en genereert feedback die is afgestemd op de huidige context. Bakken werd gekozen als toepassingsdomein omdat het een complexe vaardigheid is die nauwkeurige technieken en inzicht in de interactie van ingrediënten vereist, waardoor het een geschikte kandidaat is voor deze toepassing.

Het ontwikkelproces omvatte aanvankelijke testen met statische beelden, gevolgd door het ontwerp van een adaptief algoritme voor dynamische beeldcaptatie. Dit werd vervolgens geïntegreerd in een Unity-gebaseerde AR-toepassing die in staat is realtime, gepersonaliseerde feedback te geven. Het prototype toont de haalbaarheid aan van het combineren van AR, LLM's en computer vision voor het overdragen van menselijke vaardigheden, hoewel huidige beperkingen in nauwkeurigheid, latentie en prestaties blijven bestaan.

De belangrijkste bijdrage van dit onderzoek ligt in het aantonen hoe intelligente AR-systemen verder kunnen gaan dan statische stapsgewijze instructies en richting meer interactieve, gepersonaliseerde begeleiding. Door de kloof tussen doen en weten te overbruggen, beoogt deze thesis te laten zien hoe realtime, gepersonaliseerde feedback effectiever en boeiender vaardigheidsverwerving kan ondersteunen.

Gerelateerd werk

Dit werk put uit en bouwt voort op eerder onderzoek in verschillende overlappende gebieden, waaronder augmented reality (AR) voor vaardigheidsoverdracht en leren, AR-toepassingen in kookomgevingen, realtime feedbacksystemen, computer vision voor AR en het gebruik van large language models (LLM's) voor instructie- en feedbacktoepassingen.

Op het gebied van AR-gebaseerde vaardigheidsoverdracht tonen systemen zoals AdapTutAR [1] en Reflective Make-AR [2] aan hoe contextuele overlays en adaptieve instructies gepersonaliseerd leren kunnen ondersteunen. Evenzo tonen telepresence-systemen zoals Loki [3] de voordelen van menselijk begeleide realtime feedback aan, terwijl studies zoals Perspective Matters [4] het effect van het perspectief op taakprestaties benadrukken. Deze benaderingen zijn echter afhankelijk van vooraf gedefinieerde instructiesets of menselijke betrokkenheid en missen autonome foutdetectie. Deze thesis breidt dit onderzoeksgebied uit door LLM's te gebruiken voor realtime, gepersonaliseerde feedback en foutcorrectie.

Keukenomgevingen bieden een bijzonder uitdagende maar waardevolle omgeving voor AR-toepassingen vanwege hun dynamische, multimodale aard, die ingrediënten, gereedschappen en timing omvat. Toepassingen zoals AREasyCooking [5], Smart Cook [6] en Smart Kitchen [7] hebben aangetoond hoe AR-, sensor- en AI-technologieën kunnen helpen bij het herkennen van ingrediënten, het begeleiden bij recepten en het volgen van de voortgang van gebruikers. Andere inspanningen, zoals de AR-gebaseerde kooktoepassing van Majil et al. [8] en CookAR [9], richten zich op toegankelijkheidsgerichte interfaces en apparaten. Hoewel deze systemen de bruikbaarheid verbeteren, bieden ze voornamelijk stapsgewijze instructies zonder adaptieve foutdetectie aan. Meer recente benaderingen, zoals Step Differences [10], laten zien hoe videogeconditioneerde modellen gebruikersacties kunnen vergelijken met demonstraties om fouten te identificeren, hoewel ze beperkt blijven tot offline videoanalyse. Voortbouwend op deze fundamenten, introduceert het voorgestelde systeem een realtime AR-bakassistent die in staat is fouten te observeren, interpreteren en corrigeren terwijl ze zich voordoen.

Onderzoek naar realtime feedbacksystemen toont hun waarde aan in meerdere domeinen, zoals koken, industrieel trainen, reanimatie (CPR) en online onderwijs. Systemen zoals AdapTutAR [1], Smart Kitchen [7], CPR Tutor [11] en Sara the Lecturer [12] benadrukken de effectiviteit van adaptieve, multimodale en dialooggestuurde feedback bij het verbeteren van vaardigheidsverwerving. Deze oplossingen zijn echter vaak afhankelijk van sensoren of gecontroleerde omgevingen. Daarentegen onderzoekt deze thesis een meer flexibele, sensorloze omgeving waarin computer vision en LLM's worden gebruikt om realtime, gepersonaliseerde foutdetectie in baktaken mogelijk te maken.

Computer vision is centraal in AR-toepassingen en maakt objectherkenning, actietracking en contextbewustzijn mogelijk. Systemen zoals AREasyCooking [5], Smart Cook [6], Smart Kitchen [7] en de AR-gids van Majil et al. [8] laten zien hoe beeldherkenning en deep learning-modellen zoals YOLO [13] kunnen worden gebruikt in keukenomgevingen. Datasets zoals EPIC-KITCHENS [14], CMU-MMAC [15], Ego4D [16] en Ego-Exo4D [17] bieden bovendien grootschalige benchmarks voor egocentrisch en multimodaal visieonderzoek. Hoewel deze ontwikkelingen aantonen hoe computer vision AR kan verbeteren, stoppen ze vaak voordat de adaptieve feedback daadwerkelijk gebruikersfouten kan aangeven. Deze thesis maakt gebruik van computer vision niet alleen voor herkenning, maar ook voor het interpreteren van gebruikersacties, waardoor intelligentere feedbackgeneratie mogelijk wordt.

Ten slotte tonen recente ontwikkelingen in LLM's veel potentieel voor het bieden van adaptieve, mensachtige feedback. Systemen zoals Step Differences [10], AQuA [18], belichaamde AI-tutoren [19] en multimodale AR-agenten [20] benadrukken het vermogen van LLM's om te redeneren over visuele en tekstuele input, gebruikers door complexe taken te begeleiden en realtime gepersonaliseerde antwoorden te geven. Toepassingen in de voedingswetenschap [21], waaronder RecipeGPT en FoodGPT, suggereren bovendien hun potentieel bij het genereren en aanpassen van kookinstructies. Voortbouwend op deze lijn van onderzoek, integreert de

in deze thesis ontwikkelde AR-bakassistent LLM's met computer vision om contextbewuste, gepersonaliseerde en corrigerende feedback direct in echte baktaken te leveren.

Samengevat, terwijl eerder onderzoek aanzienlijke vooruitgang toont in AR voor leren, keukenapplicaties, realtime feedback, computer vision en LLM-gebaseerde instructie, blijven bestaande systemen beperkt in hun vermogen om gebruikersfouten autonoom en realtime te detecteren en corrigeren. Deze thesis adresseert deze kloof door een systeem te introduceren dat AR, computer vision en LLM's combineert om te fungeren als een mensachtige assistent, die dynamische, contextbewuste, gepersonaliseerde feedback biedt en vaardigheidsoverdracht een interactiever en boeiender proces maakt.

Concept

Het verwerven van nieuwe vaardigheden is zelden zo eenvoudig als het volgen van stapsgewijze instructies. Of je nu een vak leert, iets in elkaar zet of voedsel bereidt, vooruitgang hangt af van het herkennen van de dingen die je verkeerd hebt gedaan. Het begrijpen van de oorzaak van je fouten en leren hoe je ze kunt corrigeren is een cruciaal onderdeel van vaardigheidsontwikkeling. Bestaande instructiesystemen leggen de nadruk op stapsgewijze begeleiding, maar bieden zelden adaptieve, realtime feedback. De voorgestelde augmented reality (AR) bakassistent maakt gebruik van technologieën zoals kunstmatige intelligentie en AR om deze kloof te verkleinen. Het systeem begeleidt beginnende bakkers door een meeslepende en handsfree leerervaring. Het levert bakinstructies en gepersonaliseerde, contextbewuste, realtime feedback direct in het gezichtsveld van de gebruiker. In tegenstelling tot traditionele AR-keukentools grijpt het bovendien in wanneer fouten optreden, legt de oorzaken uit en geeft aan hoe ze te corrigeren. Het volgt daarmee de trend in onderzoek naar vaardigheidsverwerving, dat laat zien dat directe en contextbewuste feedback de leerervaring verbetert. Bakken werd gekozen als toepassingsdomein vanwege de behoefte aan precisie en de duidelijke zichtbaarheid van fouten, waardoor het een ideaal domein is voor foutdetectie en directe feedback. De assistent draait op een op het hoofd gedragen AR-apparaat, de Magic Leap 2, dat zowel het perspectief van de gebruiker vastlegt als contextuele begeleiding direct op het AR-scherm weergeeft.

Een demonstratie met een eenvoudig recept voor een mokcake illustreert de functionaliteit van het systeem. Spraakopdrachten maken handsfree interactie mogelijk, terwijl overlays zoals dynamische weegvisualisaties, timers en tekstuele feedback continue ondersteuning bieden. Wanneer fouten worden gemaakt, detecteert het systeem de fout, legt het probleem uit en stelt oplossingen voor. De demonstratie laat zien hoe de assistent een recept transformeert tot een meeslepende leerervaring. Het combineert precisiemetingen, foutdetectie, corrigerende feedback en handsfree interactie in één naadloos proces dat gebruikers niet alleen helpt een recept te voltooien, maar ook hun vaardigheden in de loop van de tijd verbetert. Door gebruik te maken van AR en LLM-gestuurde feedback belichaamt het systeem een nieuwe benadering van digitale vaardigheidsverwerving, die realtime, gepersonaliseerde begeleiding biedt zonder de noodzaak van een menselijke instructeur.

Implementatie

De augmented reality bakassistent is ontwikkeld in meerdere fasen, te beginnen met offline experimenten en uiteindelijk uitgegroeid tot een volledig functioneel proof-of-concept systeem. Het proces combineerde vroege verkenningen op het gebied van zowel computer vision als large language models (LLM's) met zorgvuldige overweging van de beperkingen van onze hardware en de prestaties daarvan. De eerste fase richtte zich op het evalueren van de haalbaarheid van het gebruik van large language models om fouten te detecteren vanuit statische afbeeldingen. Veel tests met bakvoorbeelden toonden aan dat GPT-4 betrouwbaar de stapprogressie en gemaakte fouten tijdens het bakproces kon herkennen. Dit benadrukte het potentieel van LLM-gebaseerde foutdetectie. Tegelijkertijd rees de vraag: welke frames zijn belangrijk genoeg om vast te leggen en naar het LLM te sturen voor analyse? Om dit aan te pakken, hanteerden we een strategie waarbij frames op vaste intervallen werden vastgelegd, samengevoegd tot een gelabeld raster en ingediend als een gecombineerde afbeelding. Deze aanpak bood voldoende contextuele informatie en bleef tegelijkertijd computationeel haalbaar.

De volgende fase integreerde deze bevindingen in een realtime AR-toepassing. Het Magic Leap 2-headset werd gekozen als het primaire apparaat vanwege de balans tussen draagcomfort, beeldkwaliteit en cameraintegratie. Unity diende als de ideale ontwikkelomgeving vanwege de sterke ondersteuning voor het Magic Leap SDK. Helaas deden zich enkele prestatieknelpunten voor in de rastercompositiefuncties van Unity, wat leidde tot de beslissing om de preprocessing van de afbeeldingen in Python uit te voeren. Er werd een TCP-socket geïmplementeerd om Unity's AR-interface te verbinden met de geoptimaliseerde backend van Python, waardoor efficiënte preprocessing, gestructureerde LLM-verzoeken en JSON-gebaseerde feedbacklevering mogelijk werden. Deze architectuur zorgde ervoor dat Unity zich kon richten op rendering en interactie, terwijl Python de computationeel intensieve taken afhandelde. Om de reactietijd verder te verbeteren en overhead te minimaliseren, paste het systeem bovendien dynamisch de frame-capturerate aan op basis van de door het LLM gerapporteerde `completion_time`.

Tot slot werd een grafische gebruikersinterface ontworpen om alle functionaliteiten te integreren in een toegankelijke gebruikerservaring. Belangrijke componenten waren onder andere een receptcatalogus, informatieve overlays voor spraakopdrachten en een instructieweergave die tips, oplossingen, extra informatie en foutdetectie direct in het gezichtsveld van de gebruiker presenteerde. Extra functies zoals timers en weegoverlays werden eveneens onderzocht om de bruikbaarheid van het systeem te vergroten, hoewel technische beperkingen in Optical Character Recognition-bibliotheken de robuustheid beperkten. Ondanks deze beperkingen bood de interface een meeslepende en functionele ervaring die gebruikers in staat stelde recepten te volgen met realtime, contextbewuste, gepersonaliseerde feedback.

De implementatie toonde aan hoe large language models en augmented reality kunnen worden gecombineerd tot een samenhangend systeem dat complexe, praktische activiteiten ondersteunt. Door ontwerpafwegingen tussen prestaties, bruikbaarheid en technische haalbaarheid te balanceren, valideerde het project met succes het concept van een AR-gebaseerde bakassistent, wat de weg vrijmaakt voor verdere verfijning en toekomstige verkenning.

Evaluatie

De evaluatie van de augmented reality bakassistent richtte zich op het testen of het systeem kon voldoen aan zijn kerndoel: het bieden van accurate, contextbewuste, gepersonaliseerde feedback die gebruikers kan helpen bij het detecteren en corrigeren van fouten tijdens het bakproces. Zowel bij fictieve als bij recepten uit de praktijk toonde de assistent duidelijke veelbelovendheid, maar tegelijkertijd kwamen enkele beperkingen naar voren op het gebied van foutdetectie, stapvoortgang en prestaties.

Bij de tests met fictieve recepten slaagde de assistent er consequent in om fouten op basis van hoeveelheden te identificeren wanneer het verschil tussen de vereiste en de daadwerkelijke hoeveelheid visueel duidelijk was op de weegschaal. Zo detecteerde hij bijvoorbeeld betrouwbaar zowel te weinig als te veel suiker. De evaluatie bracht echter ook twee terugkerende zwaktes aan het licht. Ten eerste ging het systeem soms door naar de volgende receptstap, zelfs wanneer een fout was vastgesteld. Ten tweede had de assistent moeite met het onderscheiden van visueel vergelijkbare ingrediënten, zoals suiker en zout, of een blender en een keukenmachine.

De evaluaties met recepten uit de praktijk zonder opzettelijke fouten toonden aan dat de assistent in de praktijk kon functioneren zoals bedoeld. Bij het volgen van recepten zoals chocolademousse en chocolate chip cookies doorliep de assistent de stappen op een logische manier en gaf hij consequent nuttige tips en aanvullende informatie. Valse positieve foutmeldingen kwamen zelden voor. Bij meer uitdagende recepten, zoals macarons, kwamen echter inconsistenties naar voren. Hier gaf de assistent af en toe misleidende of irrelevante feedback, zoals waarschuwingen over ontbrekende ingrediënten die al waren toegevoegd of het verwarren van keukengerief (blender versus food processor). Hoewel dergelijke fouten niet extreem storend waren, laten ze zien dat het systeem minder ervaren bakkers zou kunnen misleiden met zogenaamde positieve negatieve fouten.

Ten slotte gaven de recepten uit de praktijk met opzettelijke fouten de meeste inzichten in de foutafhandelingsmogelijkheden van het systeem. In scenario's waarbij het gegeven recept niet overeenkwam met de daadwerkelijke handelingen van de gebruiker, genereerde de assistent talrijke foutmeldingen en identificeerde correct het verschil tussen de verwachtingen van het recept en de waargenomen stappen. Echter, opnieuw ging de assistent vaak te vroeg door naar de volgende stap, in plaats van ervoor te zorgen dat de gebruiker de fout eerst corrigeerde. Bij complexere tests, zoals het macaronrecept met opzettelijke fouten, identificeerde hij twee van de drie opzettelijke fouten en gaf soms praktische oplossingsgerichte feedback. Toch raakte de instructiestroom rommelig en uit volgorde wanneer meerdere fouten tegelijk optraden tijdens het bakproces, wat suggereert dat het LLM overweldigd raakte door te veel valse input.

Al met al bevestigde de evaluatie dat de assistent in staat is om zinvolle, realtime begeleiding te bieden in veel bakscenario's, vooral bij eenvoudige recepten en in situaties waarin fouten duidelijk en meetbaar zijn. Het systeem verbetert de gebruikerservaring door contextbewuste, gepersonaliseerde feedback, nauwkeurige timers en nuttige inzichten, waarmee het potentieel als effectief leermiddel wordt gevalideerd. Niettemin wijzen de terugkerende uitdagingen van te vroege stapvoortgang, verkeerde classificatie van ingrediënten/gereedschap en sommige inconsistente foutdetecties in complexe scenario's op gebieden die verdere verfijning vereisen. Deze bevindingen benadrukken de noodzaak voor een robuustere voortgangscontrole, verbeterde visuele discriminatie tussen ingrediënten en gereedschappen en een sterker mechanisme om echte fouten te onderscheiden van opzettelijke variaties.

Discussie

Deze thesis presenteerde een interactieve augmented reality bakassistent die computer vision en large language models combineert. Als proof of concept toonde het systeem aan dat realtime begeleiding bij taken haalbaar is en bijdraagt aan een effectievere leerervaring. De evaluatie bracht echter ook verschillende beperkingen aan het licht, waaronder misinterpretatie van prompts, moeilijkheden bij het onderscheiden van visueel vergelijkbare objecten en inconsistente foutdetectie en feedback. Bepaalde ontwerpkeuzes, zoals de visualisatie van weegresultaten en dynamische capture-intervallen, bleken weinig toegevoegde waarde te hebben of zelfs de betrouwbaarheid te verminderen.

Ondanks deze uitdagingen legt dit werk een solide basis voor toekomstig onderzoek. Door gebruik te maken van krachtigere LLM's, geavanceerde software voor actieherkenning te integreren, prompt engineering te verfijnen en flexibele receptinvoer mogelijk te maken, zou het systeem betrouwbaarder, aanpasbaarder en beter schaalbaar kunnen worden. Over het geheel genomen toont het systeem duidelijk potentieel als realtime bakassistent. In zijn huidige vorm moet het worden beschouwd als een proof of concept dat de basis legt voor verdere ontwikkeling. Het aanpakken van de geïdentificeerde beperkingen in foutdetectie, consistentie van feedback en schaalbaarheid van recepten zal essentieel zijn voor het bouwen van een betrouwbaardere en breed inzetbare toepassing.

Conclusie

Deze thesis onderzocht het ontwerp en de ontwikkeling van een realtime bakassistent die AR-gebaseerde interfaces combineert met large language models om realtime, contextbewuste, gepersonaliseerde feedback te geven tijdens het bakproces. Bakken werd gekozen als domein omdat het precisie vereist en weinig ruimte laat voor fouten, wat het de perfecte omgeving maakt om een dergelijk systeem te testen. De resultaten toonden aan dat de assistent gebruikers door recepten kon begeleiden terwijl hij veelvoorkomende fouten zoals verkeerde hoeveelheden, overgeslagen stappen en onjuiste technieken detecteerde.

In vergelijking met statische tutorials creëerde deze benadering een meer interactieve en adaptieve leerervaring, die zowel het wat, als het waarom van elke stap uitlegde. Tegelijkertijd bracht het werk verschillende beperkingen aan het licht: systeem-bottlenecks, misinterpretaties door het LLM, visuele verwarring en af en toe voortijdige of rommelige reacties. Deze problemen benadrukken de uitdagingen van het balanceren van nauwkeurigheid, bruikbaarheid en prestaties in één systeem. Het ontwikkelen van dit prototype ging net zozeer over het navigeren van afwegingen als over het bouwen van technologie. Meer geavanceerde modellen en methoden hadden de resultaten kunnen verbeteren, maar zouden ook middelen hebben vereist die buiten de scope van dit project lagen. Uiteindelijk benadrukte dit proces dat innovatie vaak ligt in het maken van functionele compromissen in plaats van het streven naar perfectie. Het benadrukte ook dat menselijke vaardigheidsoverdracht niet alleen afhangt van technologie, maar van hoe mensen daadwerkelijk leren. De assistent zou minder moeten optreden als een strikte instructeur en meer als een ondersteunende partner, die gebruikers begeleidt, corrigeert en aanmoedigt, terwijl hij nog steeds ruimte laat voor fouten en groei.

Vooruitkijkend zouden vooruitgangen in LLM's, computer vision en systeembouw veel van de huidige beperkingen kunnen oplossen. Het incorporeren van actieverkenningsmodellen, videoanalyse en community-gedreven receptendatabases zou de mogelijkheden en schaalbaarheid van de assistent vergroten. Concluderend presenteert dit onderzoek een vroege maar betekenisvolle stap richting AR-gebaseerde menselijke vaardigheidsoverdracht. Hoewel het prototype nog onvolmaakt is, toont het aan dat realtime AI-feedback kan ondersteunen bij het leren van complexe taken zoals bakken op een meer toegankelijke en intuïtieve manier.

Summary

Introduction

This thesis explores how large language models (LLMs) and computer vision can be leveraged to build an interactive augmented reality (AR) system to support real-time learning through error detection and adaptive feedback. Unlike traditional step-by-step instructional tools, the proposed system interprets user actions, identifies potential errors and provides context-aware, personalized feedback that help learners understand not only what to do but also why.

To investigate this, an AR-based prototype was developed using the Magic Leap 2 headset. The system continuously captures images from the user's perspective, preprocessing them and transmitting them to a LLM for analysis. The LLM then interprets the user's progress and generates feedback tailored to the current context. Baking was chosen as the application's domain because it is a complex skill that involves precise techniques and understanding of ingredient interactions, making it a good candidate for this application.

The development process involved initial testing with static images, followed by the design of an adaptive algorithm for dynamic image capture. This was then integrated into a Unity-based AR application capable of giving real-time, personalized feedback. The prototype demonstrates the feasibility of combining AR, LLMs and computer vision for human skill transfer, though current limitations in accuracy, latency and performance remain.

The key contribution of this research lies in showing how intelligent AR systems can move beyond static step-by-step instructions toward more interactive, personalized guidance. By bridging the gap between doing and knowing, this thesis aims to show how real-time, personalized feedback can support more effective and engaging skill acquisition.

Related work

This work draws from and builds upon prior research in several intersecting areas including augmented reality (AR) for skill transfer and learning, AR applications in cooking environments, real-time feedback systems, computer vision for AR and the use of large language models (LLMs) models for instruction and feedback applications.

In the domain of AR-based skill transfer, systems such as AdapTutAR [1] and Reflective Make-AR [2] demonstrate how contextual overlays and adaptive instructions can support personalized learning. Similarly, telepresence systems like Loki [3] show the benefits of human-guided real-time feedback, while studies such as Perspective Matters [4] highlight the influence of viewpoint on task performance. However, these approaches rely on predefined instruction sets or human involvement and lack autonomous error detection. This thesis extends this body of work by using LLMs to provide real-time, personalized feedback and mistake correction.

Kitchen environments provide a particularly challenging yet valuable place for AR applications due to their dynamic, multimodal nature involving ingredients, tools and timing. Applications such as AREasyCooking [5], Smart Cook [6] and Smart Kitchen [7] have demonstrated how AR, sensor and AI technologies can support ingredient recognition, recipe guidance and user progress tracking. While other efforts, including Majil et al.'s AR-based cooking application [8] and CookAR [9], focus on accessibility-oriented interfaces and devices. While these systems improve usability, they primarily provide step-by-step instructions without adaptive error detection. More recent approaches, such as Step Differences [10], highlight how video-conditioned models can compare user actions against demonstrations to identify mistakes, though they remain limited to offline video analysis. Building on these foundations, the proposed system introduces a real-time AR baking assistant capable of observing, interpreting, and correcting errors as they occur.

Research on real-time feedback systems demonstrate their value across multiple domains such as cooking, industrial training, CPR and online education. Systems like AdapTutAR [1], Smart Kitchen [7], CPR Tutor [11] and Sara the Lecturer [12] highlight the effectiveness of adaptive, multimodal and dialogue-driven feedback in enhancing skill acquisition. Yet, these solutions often depend on sensors or controlled settings. By contrast, this thesis explores a more flexible, sensor-free environment where computer vision and LLMs will be used to enable real-time, personalized mistake detection in baking tasks.

Computer vision is central to AR applications, enabling object recognition, action tracking and contextual awareness. Systems such as AREasyCooking [5], Smart Cook [6], Smart Kitchen [7] and Majil et al.'s AR guide [8] illustrate how image recognition and deep learning models like YOLO [13] can be used in kitchen environments. While datasets such as EPIC-KITCHENS [14], CMU-MMAC [15], Ego4D [16] and Ego-Exo4D [17] further provide large-scale benchmarks for egocentric and multimodal vision research. While these advances demonstrate how computer vision can enhance AR, they often stop short of delivering adaptive feedback tied directly to user errors. This thesis leverages computer vision not only for recognition but also for interpreting user actions, enabling more intelligent feedback generation.

Lastly, recent advances in LLMs show a lot of potential in providing adaptive, human-like feedback. Systems such as Step Differences [10], AQuA [18], embodied AI tutors [19] and multimodal AR agents [20] highlight the ability of LLMs to reason over visual and textual inputs, guide users through complex tasks and personalize responses in real time. Applications in food science [21], including RecipeGPT and FoodGPT, further suggest their potential in generating and adapting cooking instructions. Building on this line of work, the AR baking assistant developed in this thesis integrates LLMs with computer vision to deliver context-aware, personalized and corrective feedback directly in real-world baking tasks.

In summary, while prior research demonstrates substantial progress in AR for learning, kitchen applications, real-time feedback, computer vision,and LLM-based instruction, existing systems remain limited in their ability to autonomously detect and correct user mistakes in real time. This thesis addresses this gap by introducing a system that combines AR, computer vision and LLMs to act as a human-like assistant, offering dynamic, context-aware, personalized feedback that transforms human skill transfer into a more interactive and engaging process.

Concept

Acquiring new skills is rarely as simple as following step-by-step instructions. Whether you are learning a craft, assembling something or preparing food, progress depends on recognizing the things you did wrong. Understanding the cause of your mistakes and learning how to correct them is a crucial part in skill development. Existing instructional systems tend to emphasize step-by-step guidance but rarely provide adaptive, real-time feedback. The proposed augmented reality (AR) baking assistant leverages technologies like artificial intelligence and AR to try and close this gap. The system guides novice bakers through an immersive and hands-free learning experience. It delivers baking instructions and personalized, context-aware, real-time feedback directly onto the user's field of view. And unlike traditional AR kitchen tools, it intervenes when errors occur, explaining their causes and how to fix them. It therefore follows the trend of the research on skill acquisition, which shows that immediate and context-aware feedback enhances the learning experience. Baking was chosen as the domain due to its demand for precision and the clear visibility of mistakes, making it an ideal domain for error detection and immediate feedback. The assistant runs on a head-mounted AR device, the Magic Leap 2, which both captures the user's perspective and delivers contextual guidance directly onto the AR screen.

A walkthrough using a simple mug cake recipe demonstrates the system's functionality. Voice commands enable hands-free interaction, while overlays such as dynamic weighing visualization, timers and textual feedback provide continuous support. When mistakes are made, the system will detect the error, explain the issue and suggest solutions. The walkthrough illustrates how the assistant transforms a recipe into an immersive learning experience. It combines precision measurements, error detection, corrective feedback and hands-free interaction into a seamless process that not only helps users complete a recipe but also improves their skills over time. By leveraging AR and LLM-driven feedback, the system embodies a new approach to digital skill acquisition, offering real-time, personalized guidance without the need for a human instructor.

Implementation

The augmented reality baking assistant was developed in multiple stages, beginning with offline experiments and eventually turning into a fully functional proof-of-concept system. The process combined early explorations in both computer vision and large language models (LLMs) with careful consideration of the constraints of our hardware and its performance. The first stage focused on evaluating the feasibility of using large language models to detect mistakes from static images. Many tests with baking examples demonstrated that GPT-4 could reliably recognize the step progression and mistakes made during the baking process. This highlighted the potential of LLM-based mistake detection. But it also raised a question, which frames are important enough to capture and send to the LLM for analysis? To address this, we adopted a strategy of capturing frames at fixed intervals, assembling them into a labeled grid and submitting them as a combined image. This approach provided sufficient contextual information while remaining computationally feasible.

The next stage integrated these findings into a real-time AR application. The Magic Leap 2 headset was selected as the primary device due to its balance of comfort, display quality and camera integration. While Unity served as the perfect development environment due to its strong support for Magic Leap's SDK. Unfortunately, there were a few performance bottlenecks in Unity's grid composition functions, which therefore led to the decision to do the preprocessing of the images in Python. A TCP socket was implemented to connect Unity's AR interface with Python's optimized backend, enabling efficient preprocessing, structured LLM requests and JSON-based feedback delivery. This architecture ensured that Unity could focus on rendering and interaction, while Python handled the computationally intensive tasks. To further improve responsiveness and minimize overhead, the system also dynamically adjusted the frame capture rate based on the LLM's reported `completion_time`.

Finally, a graphical user interface was designed to unify all functionalities into an accessible user experience. Key components included a recipe catalogue, informational overlays for voice commands and an instruction display that presented tips, solutions, extra information and mistake detection directly in the user's field of view. Additional features such as timers and weighing overlays were also explored to extend the system's usefulness, though technical limitations in Optical Character Recognition libraries restricted their robustness. Despite these constraints, the interface provided an immersive and functional experience that enabled users to follow recipes with real-time, context-aware, personalized feedback.

The implementation demonstrated how large language models and augmented reality can be combined into a cohesive system capable of supporting complex, hands-on activities. By balancing design trade-offs between performance, usability and technical feasibility, the project successfully validated the concept of an AR-based baking assistant, paving the way for further refinement and future exploration.

Evaluation

The evaluation of the augmented reality baking assistant focused on testing whether the system could deliver on its core objective: providing accurate, context-aware, personalized feedback that could help users in detecting and correcting mistakes during the baking process. Across fictional and real-world recipes, the assistant showed clear promise, while also revealing some limitations in error detection, step progression and performance.

In the fictional recipe tests, the assistant consistently succeeded in identifying quantity-based mistakes when the difference between the required and actual amounts was visually obvious on the scale. For example, it reliably detected both under- and over-measurements of sugar. However, the evaluation also revealed two recurring weaknesses. First, the system sometimes advanced to the next recipe step even when a mistake had been identified. Second, the assistant struggled to differentiate between visually similar ingredients, such as sugar and salt or a blender and a food processor.

While real-world recipe evaluations without intentional mistakes demonstrated that the assistant could operate as intended in practice. When following recipes such as chocolate mousse and chocolate chip cookies, the assistant progressed through the steps in a logical manner, consistently provided useful tips and additional info. It rarely produced false positive mistakes. However, more challenging recipes like macarons exposed inconsistencies. Here, the assistant occasionally produced misleading or irrelevant feedback, such as warning about missing ingredients that had already been added or flagging tool substitutions (blender vs. food processor) as mistakes. Although such errors were not overly disruptive, they demonstrate that the system could mislead less experienced bakers with positive negative mistakes.

And finally, The real-world recipes with intentional mistakes provided the most insight into the system's error-handling capabilities. In scenarios where the provided recipe did not match the user's actual actions, the assistant generated numerous mistake detections, correctly identifying mismatches between recipe expectations and observed steps. However, once again, the assistant often progressed through steps prematurely rather than ensuring that the user corrected the mistake first. In more complex tests, such as the macaron recipe with deliberate errors, it identified two of the three intentional mistakes and sometimes provided practical solution-oriented feedback. Still, the instruction flow became cluttered and out of sequence when multiple mistakes occurred during the baking process, suggesting that the LLM got overwhelmed with too much false input.

Overall, the evaluation confirmed that the assistant is capable of delivering meaningful, real-time guidance in many baking scenarios, especially for straightforward recipes and in situations where mistakes are clear and measurable. The system enhances the user experience through context-aware, personalized feedback, accurate timers and helpful insights, validating its potential as an effective learning tool. Nonetheless, the recurring challenges of premature step advancement, ingredient/tool misclassifications and some inconsistent error detection in complex scenarios point to areas requiring further refinement. These findings highlight the need for more robust progression control, improved visual discrimination between ingredients and tools and a stronger mechanism for distinguishing between true errors and intentional variations.

Discussion

This thesis presented an interactive augmented reality baking assistant that combines computer vision and large language models. As a proof of concept, the system demonstrated that real-time task guidance is feasible and contributed to a more effective learning experience. However, the evaluation also revealed several limitations, including prompt misinterpretation, difficulties distinguishing visually similar items and inconsistent error detection and feedback. Certain design decisions, such as weighing visualization and dynamic capture intervals, were found to add little value or even reduce reliability. Despite these challenges, the work establishes a solid foundation for future research. By adopting more capable LLMs, integrating advanced action recognition software, refining prompt engineering and enabling flexible recipe input, the system could become a more reliable, adaptable and scalable. Overall, the system demonstrates clear potential as a real-time baking assistant. In its current form, it should be regarded as a proof of concept that establishes a foundation for future development. Addressing the identified limitations in error detection, feedback consistency and recipe scalability will be essential for building a more reliable and widely usable application.

Conclusion

This thesis explored the design and development of a real-time baking assistant that combines AR-based interfaces with large language models to provide real-time, context-aware, personalized feedback during the baking process. Baking was chosen as the domain because it demands precision and leaves little room for error, which makes it the perfect environment to test such a system.The results showed that the assistant was able to walk users through recipes while catching common errors such as incorrect measurements, skipped steps and improper techniques. Compared to static tutorials, this approach created a more interactive and adaptive learning experience, one that explained both the what and the why behind each step. At the same time, the work revealed several limitations: system bottlenecks, LLM misinterpretations, visual confusion and occasional premature or cluttered responses. These issues highlight the challenges of balancing accuracy, usability and performance in one system. Developing this prototype was as much about navigating trade-offs as it was about building technology. More advanced models and methods could have improved results but also demanded resources beyond the scope of this project. Ultimately, this process reinforced that innovation often lies in making functional compromises rather than chasing perfection. It also emphasized that human skill transfer depends not only on technology but on how people actually learn. The assistant should act less like a strict instructor and more like a supportive partner, guiding, correcting and encouraging users while still leaving space for errors and growth.

Looking ahead, advances in LLMs, computer vision and system design could resolve many of the current limitations. Incorporating action recognition models, video analysis and community-driven recipe databases would expand the assistant's capability and scalability. In conclusion, this research presents an early but meaningful step toward AR-based human skill transfer. While the prototype remains imperfect, it demonstrates that real-time AI feedback can support learning complex tasks like baking in a more approachable and intuitive way.

Abstract

This research investigates how large language models (LLMs) and computer vision can be combined with an augmented reality (AR) based application to provide real-time feedback during human skill acquisition. Existing AR learning systems primarily focus on guiding users with step-by-step instructions throughout a process. But often lack in determining mistakes and giving context-aware, personalized feedback towards the user. Therefore limiting the effectiveness of human skill transfer. To address this gap, a prototype AR baking assistant was developed for the Magic Leap 2 headset. The system integrates computer vision with an LLM to continuously analyze user actions, detect potential errors and provide corrective feedback directly in the user's field of view through visual overlays.

The implementation process began with experiments on static images to assess the LLM's ability to recognize baking actions and detect mistakes. Insights from these tests informed the design of a real-time pipeline for image preprocessing and dynamic frame capturing, which was eventually integrated into a Unity-based AR application. The assistant was then evaluated using both fictional and real recipes, under scenarios with and without intentional mistakes.

Evaluation results show that the system was capable of detecting errors and offering context-aware, personalized feedback. Therefore making the baking process more interactive and supportive than conventional instructional methods. However, the system still has some limitations when it comes to accuracy and performance. Suggesting the need for improvements such as advanced action recognition models, better prompt engineering or more advanced LLMs.

This research demonstrates the feasibility of combining AR with LLMs and computer vision to deliver interactive, real-time guidance in practical tasks. While the prototype is constrained within our scope, it offers a foundation for future systems to move beyond static instructions towards a more adaptive, human-like learning assistance.

Inhoudsopgave

1 Introduction **17**

2 Related work **19**
2.1 Augmented Reality in Skill Transfer and Learning . . . 19
2.2 AR in kitchen environments . . . 21
2.3 Real-time feedback systems for learning . . . 23
2.4 Image analysis and computer vision in augmented reality applications . . . 25
2.5 LLMs within instruction and feedback applications . . . 28

3 Concept **31**
3.1 Determining the domain of my thesis . . . 32
3.2 Walkthrough . . . 33
3.2.1 Step 1: Launching the application . . . 33
3.2.2 Step 2: Starting the mug cake recipe and weighing the flour . . . 34
3.2.3 Step 3: Weighing sugar, mistake detection . . . 36
3.2.4 Step 4: Continuing with measuring the other ingredients . . . 37
3.2.5 Step 5: Mixing and adding liquids . . . 38
3.2.6 Step 6: Pouring the mixture and timing the microwave . . . 39
3.2.7 Step 7: Finishing the recipe . . . 40
3.3 Hardware and interaction design . . . 41

4 Implementation **43**
4.1 Testing the LLM with static images . . . 43
4.2 Hardware integration . . . 46
4.2.1 Choosing the AR headset . . . 46
4.2.2 Development environment . . . 47
4.3 Graphical user interface . . . 48

5 Evaluation **51**
5.1 Low level fictional Recipe Error Detection Tests . . . 52
5.2 Real-world recipe tests without intentional mistakes . . . 54
5.3 Real-World Recipe Tests With Intentional Mistakes . . . 55

6 Discussion **57**
6.1 Limitations . . . 57
6.2 Future work . . . 58

7 Conclusion **59**

Hoofdstuk 1

Introduction

Instructional technologies have evolved rapidly over the years, starting from cookbooks and manuals to online video tutorials and eventually interactive augmented reality (AR) [8] or virtual reality (VR) [22] applications. While these tools make it easier to complete tasks, they often follow a step-by-step instructional model that does little to support actual learning. When systems only tell users what to do and not why, mistakes tend to go unnoticed until it is too late to correct them. As a result, learners often just follow the instructions rather than actually learning the underlying skills. This suggests a need for a system that not only tracks user progress but also interprets errors and offers meaningful guidance.

This limitation motivated the central research question of this thesis:
How can large language models (LLMs) and computer vision be leveraged to build an interactive assistant that helps users detect mistakes and improve human skill transfer in real time?

Addressing this question involves overcoming several key challenges. First, user actions must be observed accurately in real time, which in this work is achieved through continuous image capturing from a head-mounted AR device. Second, the system must interpret those actions flawlessly. It has to accurately distinguish the correct progress from the potential mistakes. Finally, the system's feedback must be immediate, context-aware and personalized. It has to point out mistakes and explain how to solve them but also be able to explain why the user has to perform certain actions.

To tackle these challenges, we designed an AR-based learning application that uses computer vision and large language models to give context-aware, personalized feedback. The system continuously captures frames from the user's perspective through the Magic Leap 2 headset. These frames are then preprocessed and transmitted for analysis. The LLM then reasons about the user's current state and generates appropriate feedback. Unlike conventional AR systems that present fixed guidance, this assistant adapts dynamically to user actions, stimulating a more interactive, personalized learning experience.

For this research, we chose baking as the application's domain. Because it is a complex skill that involves precise techniques and understanding of ingredient interactions, making it a good candidate for this application. Baking is a skill-intensive activity where mistakes can easily lead to failed outcomes. Mistakes made during the baking process often go unnoticed until it is too late to correct them, leading to frustration and a steep learning curve. Learning how to bake is an iterative process, where mistakes play a crucial role in skill development. The complexity of baking mirrors many real-world skills where success depends on precise techniques, timing and an understanding of material interactions. For these reasons, baking serves as a valuable domain to test the potential of a system with real-time, adaptive feedback. At the same time, the insights gained from this work will therefore not only be limited to the kitchen. The same

approach could be applied to other domains such as manufacturing, education or sports.

While some existing systems already incorporate computer vision and AI for object detection [5, 6, 8] or tracking cooking progress [7] they often lack detailed explanations or adaptive feedback. This thesis takes a different approach by leveraging an LLM to analyze visual input and generate human-like explanations and suggestions. This therefore makes the learning experience more interactive and personalized.

To develop this application, we began with initial testing using static images. We manually captured images of key steps in the baking process of a Swiss roll cake and analyzed OpenAI's ability to interpret them. This was followed by a more dynamic approach where we recorded a video of making chocolate mousse, extracted frames at one-second intervals and analyzed OpenAI's feedback on each frame. Based on these findings we designed an adaptive algorithm that adjusts image capture frequency according to the complexity and duration of each task within the recipe. Finally, the image processing system was integrated into a Unity-based AR application allowing for real-time testing and refinement.

This research resulted in a working proof-of-concept application that demonstrates the feasibility of combining AR, computer vision and LLMs for real-time mistake detection and feedback. While the prototype has clear limitations in terms of accuracy, latency and performance, it provides valuable insights into how such systems can be designed and evaluated.

The contribution of this research lies in its potential to demonstrate how LLMs can be combined with AR and computer vision to create an interactive system that not only instructs the user but can also give context-aware personalized feedback. It also highlights the potential of transforming learning experiences from following step-by-step instructions toward developing a deeper understanding of the reasoning behind each action. By bridging the gap between doing and knowing, this thesis aims to show how real-time, personalized feedback can support more effective and engaging skill acquisition.

Hoofdstuk 2

Related work

This work draws from and builds upon prior research in several intersecting areas including augmented reality (AR) for skill transfer and learning, AR applications in cooking environments, real-time feedback systems, computer vision for AR and the use of large language models (LLMs) models for instruction and feedback applications.

2.1 Augmented Reality in Skill Transfer and Learning

Augmented Reality (AR) has become a powerful tool for enhancing skill transfer and learning in numerous domains like manufacturing, healthcare, exercise, cooking and education. This is because traditional learning methods like video tutorials often fail in providing immersive, contextual and personalized instructions. This has lead to the development of AR-based systems that aim to bridge these gaps by displaying information onto the physical environment.

AdapTutAR [1], for instance, provides a tutorial-based system designed to train workers in complex machine operations with the aid of Augmented Reality (AR). The system adapts instructions based on user interaction and performance. This allows for a more personalized learning experience and instruction generation. It continuously monitors the learner's progress and adjusts content in real-time. This is achieved by recognizing both machine state and user activity through deep learning in object recognition, as well as user activity recognition. This approach highlights the importance of adapting instructional content based on the user's state, a critical factor for effective skill transfer.

Reflective Make-AR [2] extends this idea by encouraging more engagement with the user by asking them to reflect on their actions and decisions while learning Maker skills. The system supports self-paced learning by monitoring learner performance in real-time and prompting reflections both during and after activities. It utilizes an AR head-mounted device to monitor, prompt and record reflections while the maker activity is in progress [2]. It then leverages AR features to highlight real-world objects, providing contextual information related to the maker's activity and offers multi modal feedback to learners for self-reflection. This design demonstrates the value of real-time feedback and reflective prompts to help users understand their actions and improve their techniques.

However, both AdapTutAR and Reflective Make-AR largely rely on predefined instructions and reactive feedback based on limited user interaction data. While AdapTutAR dynamically adjusts instructions, it does not detect or respond to specific errors during task execution. Similarly, Reflective Make-AR encourages user reflection but lacks the capability to identify and address real-time mistakes. This thesis proposes an approach that utilizes the capabilities of large language models (LLMs) to overcome these limitations by providing real-time feedback based on user interactions. Unlike predefined instruction sets, LLMs can generate adaptive

feedback and highlight specific errors as they occur. This helps learners not only to reflect on what went wrong but also get personalized feedback on how to correct their mistakes. Thereby enhancing effective skill transfer and learning.

Beyond automated adaptation, researchers have also explored the advantages of involving a human instructor who can provide real-time feedback during skill acquisition. Particularly remote instructions of physical tasks using Mixed Reality (MR) telepresence. Loki [3] is a system that facilitates bidirectional mixed reality telepresence and builds on the idea of Fitts and Posner's three-stage model of acquisition of motor skills [23]. Which describes how learners progress from understanding a task, to refining their movements and ultimately performing the skill automatically. The system enables an instructor to remotely guide a learner through physical tasks. It leverages video, audio and spatial data capturing alongside mixed-reality techniques. To allow users to interact with and annotate virtual environments while also providing the ability to record performances for later analysis and review. In this system, the remote instructor and the learner share a virtual workspace that supports direct manipulation of objects and viewpoint awareness through avatar representation. This therefore allows the instructor to guide the learner in real time with continuous communication, making the learning experience more closely aligned with in-person instruction. Despite it's strengths, Loki is primarily designed for synchronous teachings and depends heavily on instructor availability. The system lacks autonomous feedback and is constrained by technical demands like synchronized viewpoints. This therefore does not scale well to casual or asynchronous learning environments.

While systems like Loki focus on involving human guidance within their system. Another critical factor in human skill transfer through AR is how information is visually presented to the user. Perspective Matters [4] investigates how a user's spatial viewpoint influences their understanding and execution of motion-guided tasks in Mixed Reality. The study emphasizes that the effectiveness of motion-based instructions varies significantly depending on whether the guidance is presented from a first-person (egocentric), third-person (allocentric) or mirrored perspective. Key findings indicate that the viewpoint significantly influences user accuracy and time, therefore reducing cognitive load. In general the egocentric viewpoint outperformed in both accuracy and completion time. However the researchers found that incorporating multiple perspectives particularly the allocentric view can improve timing performance. This is especially the case in more complex tasks involving multiple body parts, where the allocentric view proved to be especially useful. While the study highlights the importance of visual perspective in guiding user performance. It doesn't address the need for adaptive feedback based on the user's real-time actions.

Although prior work has advanced AR-based skill transfer through spatial viewpoints, remote human assistance and adaptive instructions. There is still a critical gap in the ability to autonomously detect user errors and provide real-time feedback. Many current approaches deliver step-by-step guidance or adaptive instructions, but fall short in identifying specific mistakes or offering suggestions during task execution. This thesis addresses this by introducing an AR-based baking application that not only guides users through the baking process but also uses large language models (LLMs) and computer vision techniques to detect errors and deliver personalized feedback. This approach will therefore transforms the system into an autonomous human-like learning assistant. It enables users to detect errors as they occur, understand the reasons behind them and give appropriate solutions in real time. This will lead to a more engaging learning process and promotes skill mastery without the need of a human instructor.

2.2 AR in kitchen environments

Several augmented reality (AR) systems have been developed for use in kitchen settings. It is a challenging environment involving various tools, ingredients and skills. This makes it an ideal but still complex place for AR applications. These systems typically guide users through recipes using step-by-step instructions with the aid of visual overlays. This not only simplifies the cooking process but also enhances safety when handling complex recipes.

Early efforts in the AR kitchen applications often focused on providing step-by-step recipes based on ingredient identification. AREasyCooking [5] is a mobile application that uses augmented reality and barcode reading to help users identify ingredients and suggest recipes. The application allows users to input available ingredients using either AR-based image recognition via the Clarifai API[1] or a barcode scanner. These input methods improve usability and ease of use by eliminating the need to manually write down the ingredients. Once all ingredients are registered, the system matches them with recipes from the traditional Eastern European food database eCULTFOOD Atlas [24]. After the user has chosen a recipe, the system will provide a step-by-step tutorial video to guide them through the preparation process. Notably, AREasyCooking makes use of voice and eye control to navigate the tutorial. This allows the user to follow along even with their hands occupied. This hands-free interface increases both safety and convenience in the kitchen. However, it also highlights certain usability limitations with AR applications in hands-on environments like a kitchen. For instance, hand gesture controls can often be unreliable, as the system may misinterpret unintended movements while the user is actively using their hands.

Smart Cook [6] follows the same idea as AREasyCooking in suggesting recipes based on ingredients but takes a different approach. Rather than using barcode scanners or relying on third-party APIs, Smart Cook implements a custom deep learning algorithm to recognize ingredients directly from images. This not only simplifies the ingredient input method but also increases flexibility and robustness by eliminating the need for external services that may be prone to errors or limitations. It leverages convolutional neural networks (CNNs) to analyze the images and retrieves suitable recipes from the Elasticsearch database[2]. While Smart Cook shares the goal of AREasyCooking. It distinguishes itself through its use of machine learning to interpret ingredients for recipe recommendation. However, just like AREasyCooking, it lacks mechanisms for offering real-time, personalized feedback during the cooking process itself.

These systems demonstrate a growing potential of AI-powered interfaces in kitchen environments. However, their functionalities mainly focus on the early stages of the cooking process, like ingredient identification, recipe suggestion and basic instructional guidance. While these features enhance usability and lower the barrier to entry for novice cooks. They fall short in offering real-time, personalized feedback during the actual execution of tasks. This thesis builds upon the foundational work of systems like AREasyCooking and Smart Cook by introducing an AR-based baking assistant that leverages large language models (LLMs) and computer vision to detect and respond to user errors as they occur. The system moves beyond basic step-by-step tutorials by integrating adaptive instruction delivery with real-time, personalized feedback. It aims to give a more human-like and intelligent form of feedback that responds to the users actions as they happen.

To enable real-time error detection in kitchen settings, a solution has been proposed by Nagarajan and Torresan [10] to make a system that would be capable of recognizing differences between user actions and correct demonstrations. Their paper, step differences in instructional video, trains a video-conditioned language model (VCML) to identify differences between videos like variation in tools, ingredients or techniques. This therefore makes it possible for the system to track progress, detect mistakes and provide feedback bases on user actions. This is achieved by using video segments from large-scale datasets like HowTo100M [25], pairing them and leveraging existing annotations and narration to train the VCML. This approach offers a valuable

[1] https://www.clarifai.com/developer
[2] https://www.elastic.co/

mechanism for detecting subtle errors like identifying that a user is flipping the dough with a fork rather than a tong. Unlike earlier systems that offered fixed instructions, Step Differences introduces a system capable of comparison and mistake detection. This could significantly enhance the quality of real-time guidance by dynamically adapting feedback based on how a user's actions diverges from the norm. While this provides a strong foundation for automatic mistake detection, the application has so far only been applied in offline video analysis. Integrating it into a real-time AR kitchen environment remains a major challenge.

To address this challenge of delivering real-time feedback in a live kitchen environment. Systems have been explored to track progression more practically with the use of sensor-based technologies. For instance, Smart kitchen [7] proposes a user-centric approach designed to recognize cooking actions and ingredients in real time. It uses a combination of optical and thermal sensors embedded in the kitchen environment to identify ingredients, track them and identify cooking actions. The system is assumed to know the recipe in advance and gives contextual instructions based on the user's progression. Importantly the design acknowledges the flexible and often non-linear nature of real-life cooking. To support this, the recipe is represented as a tree structure that enables a more natural and adaptable cooking process. This gives users freedom to move through the steps in a way that suits their own workflow. Still, the system's dependency on sensors to determine when a cooking step is complete makes it prone to errors, especially when dealing with varied food appearances or complex actions.

Complementing this approach, Majil et al. [8] introduces an AR cooking assistant application that leverages the Magic Leap One headset. By using this wearable device, users can interact with the system without having to rely on external sensors or devices inside their kitchen. This system allows users to identify ingredients simply by looking at them, suggests suitable recipes and presents step-by-step instruction via video tutorials. It uses the built-in camera of the Magic Leap One headset to detect and classify food ingredients in real time using YOLOv5 (You only look once) [13], a deep learning model for object detection. The system then overlays instructional video clips directly onto the user's field of view. Interaction is fully controllable through natural hand gestures, making it easy to operate without the need of physically touching any user interfaces. However, this interaction method can still be prone to errors, especially when users are using their hands in kitchen activities. While the system offers notable advances by combining ingredient recognition, recipe suggestion and hands-free interactions into one single wearable device. The system doesn't support real-time monitoring of user actions or give feedback when mistakes are made. Although the video tutorials can be helpful, they are pre-recorded and cannot adapt to user mistakes.

These two systems demonstrate that real-time feedback in AR kitchen assistants could be an achievable goal. While Smart kitchen highlights the importance of having context-aware adaptable instructions in such systems, Majil et al. shows the potential of wearable devices and deep learning models for object detection. This thesis builds on the strengths of both systems by proposing an AR baking assistant that combines visual overlays with real-time, context-aware instructions powered by a large language model (LLM). Unlike systems with pre-recorded video instructions, this assistant responds to user actions as they happen. It gives real-time feedback and guides the user thought the baking process. The goal is not only to instruct but also to observe, interpret, and intervene when mistakes are made. This bridges the gap between basic tutorials and a truly immersive and interactive assistant.

Another line of work in AR kitchen environments addresses the challenge of accessibility within a kitchen setting. CookAR [9] proposes a wearable AR application designed to help people with low vision use kitchen tools more safely and efficiently. It uses real-time object affordance augmentation to distinguish between grabbable and hazardous areas on tools like knives, ladles and pans. The system combines a stereo depth camera with an Oculus Quest 2 headset to perform real-time object segmentation and display the resulting visual augmentations directly in the user's field of view. This is made possible through the development of a custom egocentric dataset of kitchen tool affordances, constructed by selecting and labeling images from the EPIC-KITCHENS dataset [14]. A user study demonstrated that CookAR significantly enhanced

participants' ability to locate, identify, and safely interact with kitchen tools. While CookAR improves safety by highlighting affordances, it does not guide users through recipe steps or detect mistakes during cooking.

Naturally, Virtual Reality (VR) has also been explored as a method for culinary education. Gorman et al. [22] demonstrated how VR can support culinary training by simulating kitchen environments and procedures. While VR offers high immersion and control over the learning environment, it fundamentally differs from AR. VR replaces the real-world with a simulated one and therefore limits its applicability to real-life situations. This is especially true in a kitchen where interaction with tools and ingredients is essential for developing skills. VR can be useful for safe training and understanding basic cooking techniques but it does not address the physical challenges of real cooking.

In summary, while existing AR and smart kitchen systems have made significant strides in areas such as ingredient recognition, hands-free recipe guidance and accessibility. A common limitation across these approaches is the lack of real-time mistake detection and personalized, context-aware feedback. The dynamic nature of cooking proposes the need for a system that can understand the user's state, but also recognize mistake and correct them as they happen. This thesis aims to bridge this gap by introducing a baking assistant that leverages a large language model to offer real-time, personalized feedback and error detection.

2.3 Real-time feedback systems for learning

Real-time feedback has always been a critical component in learning environments, especially those involving complex tasks. It enables users to correct mistakes as they happen, reinforce correct behavior and refine techniques. In contrast to delayed or retrospective feedback, it can enhance learning by maintaining momentum and reducing the risk of reinforcing incorrect behaviors. Advancements in sensor technology, computer vision and artificial intelligence have enabled more adaptable and personalized feedback systems. These could interpret user progression, detect errors and offer personalized feedback.

We have already mentioned Smart Kitchen [7] and AdapTutAR [1] as systems that recognized real-time progress through the use of deep learning algorithms. AdapTutAR changes its feedback dynamically by monitoring both machine state and user activity in real time. It leverages computer vision and a Convolutional Neural Network (CNN) to monitor what the user is doing, where they are looking at and if they're touching the right machine part. It then personalizes the instruction based on how detailed they should be. Smart Kitchen takes a more user-centric approach by embedding optical and thermal sensors in a kitchen environment to recognize and track food items while also classifying cooking actions. It assigns IDs to each food item, allowing for continuous tracking even as they are chopped, cooked, or mixed. Recipes are represented as a tree structure to reflect the non-linear nature of real cooking and enables users to follow steps in an order that suits their own workflow. While both approaches highlight the value of adaptive, real-time feedback, they do not address the need for personalized feedback or mistake detection. This thesis builds upon these ideas and leverages large language models (LLMs) and computer vision to deliver personalized, real-time feedback in a dynamic kitchen environment.

We have also looked at Step Differences [10] as a proposed solution for real-time error detection. This approach leverages a video-conditioned language model (VCLM) to compare a user's activity with a reference instructional video and identify differences in variation of tools, ingredients, and techniques. It can highlight subtle mistakes, such as using the wrong utensil or incorrect techniques. While the system has so far been applied mainly in offline video analysis, it demonstrates the possibility of real-time, context-aware, personalized feedback through the use of artificial intelligence. Building on this, this thesis applies a similar reasoning to create an AR baking assistant to provide immediate and personalized feedback during the baking process.

Another valuable contribution to the area of real-time feedback is the CPR Tutor presented in Keep Me in the Loop by Di Mitri et al. [11]. This system delivers real-time multimodal feedback to learners practicing cardiopulmonary resuscitation (CPR). The CPR Tutor identifies mistakes in chest compression techniques by analyzing a combination of kinematic data from a Kinect and electromyographic (EMG) signals collected via EMG sensors in a Myo armband. It uses a long short-term memory (LSTM) neural network to detect five key performance indicators and delivers real-time audio feedback to correct mistakes and enhance performance. In addition to the standard metrics like compression rate, depth, and release, the system also detects two new indicators not commonly tracked by commercial CPR tools: correct arm locking and proper use of body weight. The models were trained on datasets from 10 CPR experts to ensure accurate and reliable detection. The system classifies each chest compression in approximately 70 milliseconds, making it suitable for CPR training. A user study showed that the CPR Tutor contributed to short-term improvements, particularly in three out of the five monitored performance areas. This study aims to demonstrate how multimodal data can be used to support psychomotor skill development through real-time feedback. It shows the power of combining sensor-based data with machine learning algorithms to deliver personalized feedback in time-sensitive, skill-based training environments. However, the system remains domain specific and depends on a controlled environments with dedicated hardware. In contrast, this thesis tries to explore the possibility of bringing a similar real-time feedback system into a more casual and flexible setting like a kitchen. Instead of relying on sensors, it leverages large language models (LLMs) and computer vision to interpret user actions and provide personalized, real-time feedback.

Additionally, in the domain of online education, Sara, the Lecturer [12] introduces a conversational agent that provides scaffolding-based voice and text feedback during video lectures. Sara engages with the student by asking comprehension questions at key moments during a lecture. When a student responds incorrectly, the system gives a follow-up dialogue with prompts designed to guide them toward the correct answer. Sara uses natural language processing (NLP) to interpret student inputs and adapts its responses accordingly. By communicating through both voice and text, Sara leverages the Cognitive Theory of Multimedia Learning [26], which suggests that learning improves when information is delivered through multiple channels. The system provides guidance tailored to the students current level of understanding, encouraging real-time reflection and promoting the application of what they've learned to new problems. The study highlights how scaffolding-based voice and text feedback enhances both information retention and transfer ability within the students. Despite being implemented in a passive learning environment, the essential principles of adaptive, dialogue-driven feedback offer valuable insights. This thesis applies a similar approach by integrating an interactive dialogue with a LLM during physical tasks like baking. The proposed system uses adaptive LLM-driven prompts to detect errors and provide personalized, real-time feedback.

The reviewed systems demonstrate the potential of real-time feedback in enhancing skill acquisition across various domains. They show the value in shifting from static step-by-step instruction to a more dynamic, user-centered approach. They highlight the benefits of combining sensor data, computer vision, and AI-driven models to monitor user actions, detect errors and deliver real-time, personalized feedback. However, many of these approaches are reliant on specialized hardware or are limited to passive or highly controlled environments. Building on these foundations, this thesis tries to explore how real-time, adaptive feedback can be implemented in a more flexible and active settings, specifically a sensor-free kitchen environment. By leveraging large language models and computer vision, the proposed system aims to accurately interpret user actions, identify errors, and generate personalized, real-time feedback. In doing so this thesis shows how advanced AI techniques such as large language models and computer vision can be used in everyday learning activities. It shows their potential outside of controlled lab settings and helps make real-time personalized feedback more accessible in practical contexts like cooking.

2.4 Image analysis and computer vision in augmented reality applications

Image analysis and computer vision are the foundation of many augmented reality (AR) systems. They enable the program to interact with objects within the physical world. These technologies allow AR applications to detect objects, track user actions and understand the environment in real time. This plays a crucial role in things like recognizing tools, ingredients and user behavior. Computer vision makes it possible to bridge the gap between digital and physical worlds, enabling AR systems to deliver fast, context-aware feedback based on real-world interactions.

Several system previously mentioned incorporate some kind of computer vision within their AR application to interpret user input, recognize objects and support real time interaction. Their image analysis techniques highlight the growing role of computer vision within AR, allowing for more intelligent, adaptive and responsive user experience. AREasyCooking [5], for instance, uses computer vision to support ingredient recognition. The system allows users to scan available ingredients either through a barcode scanner or via image-based recognition powered by the Clarifai API[3]. This eliminates the need for manual input and significantly simplifies the cooking preparation phase. AREasyCooking demonstrates how vision APIs can be integrated into AR applications to support basic object recognition. However, the system is limited to identifying ingredients. It doesn't recognize user actions or detect errors, therefore restricting its usefulness in providing real-time feedback. Smart Cook [6] enhances this approach by introducing a more robust and flexible ingredient recognition system. The system implements its own custom deep learning model instead of using third party API's. It uses convolutional neural networks (CNNs) to classify food items from captured images. This increases reliability and gives more control over the dataset and training. However, systems like Smart Cook and AREasyCookings only focus on the preparation phase of cooking and do not incorporate computer vision techniques during the cooking process itself.

In contrast, Smart Kitchen [7] integrates computer vision more holistically into the cooking process. The system uses optical and thermal sensors to identify ingredients, label them with IDs and continuously tracks them as they are cut, mixed or cooked. This approach enables the system to autonomously monitor cooking progress in real time, without having to rely on user interaction. The system also classifies cooking actions. These action are grouped into five categories: cut/peel, stir-fry, deep-fry, boil, and mix. All categories except mix are divided based on user location, either at the tabletop area or the stove. The system then uses thermal sensor data to detect which action is being performed, with an action being recognized once the user remains in a given area for more than five seconds. These classifications enables the system to adapt its instructions in real time based on the non-linear nature of the recipe tree. The system is notable for its real-time ingredient recognition and tracking capabilities, allowing users to cook at their own pace while still providing feedback when needed. However, the system remains prone to errors due to its reliance on sensor data and can only be used in controlled environment with the appropriate setup. This limits its portability and makes it less suitable for use in everyday kitchens. The Augmented Reality Based Interactive Cooking Guide by Majil et al. [8] demonstrates how wearable AR devices can serve as a portable alternative for using computer vision to monitor cooking progress. The system utilizes the YOLOv5 (You only look once) [13] object detection algorithm to identify food items in real time through the built-in camera of a Magic Leap One headset. The architecture of the YOLOv5 model is highly complicated and can be divided into three stages: the backbone for feature extraction, the neck for feature fusion and the head for object prediction. The model is then trained on the Q-100 food ingredient dataset[4]. Once the ingredients are correctly recognized, instructional videos are overlaid onto the user's field of view, providing video clips of step-by-step cooking instruction. This approach showcases how advanced object detection models like YOLOv5 can

[3]https://www.clarifai.com/developer
[4]https://github.com/Q-100/ingredients-classification

be effectively integrated with wearable AR headsets in kitchen environments without having to rely on external sensors. However, while the system combines wearable AR technology and deep learning, it is limited to ingredient detection and does not track user activity, nor does it adjust instructions dynamically in response to user errors during the cooking process.

Together, these systems highlight the growing role of computer vision in augmented reality applications within kitchen environments, covering everything from ingredient recognition to action tracking and environmental understanding. Although each approach introduces techniques to improve usability, none provide real-time error detection or deliver personalized, context-aware feedback. This thesis builds on these foundations by integrating computer vision not only for recognition but also for interpreting user actions, allowing for more adaptive and personalized feedback through the use of large language models.

Another notable application of computer vision in AR kitchen environments is CookAR [9], which provides real-time tool affordance augmentations for people with low vision. CookAR distinguishes itself from other AR systems by focusing on accessibility rather than the cooking process itself. The system combines a stereo depth camera with an Oculus Quest 2 headset to perform real-time object segmentation and display them on the users field of view. It augments specific parts of kitchen tools and labels them as "grabbableäreas in green and "hazardousäreas in red. This affordance labeling ensures more user awareness and confidence, allowing safer and more efficient interactions with kitchen equipment. To support this, CookAR developed a custom egocentric affordance dataset by extracting and labeling images from the EPIC-KITCHENS dataset [14]. The authors used the YOLOv8 model[5], trained on the MS COCO dataset [27], to automatically identify relevant frames and then manually label parts of the tools with the Roboflow platform[6]. The dataset consists of 10,152 labeled images across 18 kitchen tool affordance classes, such as pan handle, knife blade, etc. This study showcases the power of using large-scale datasets for developing and training real-time AR applications.

The EPIC-KITCHENS dataset [14] itself has emerged as a foundational benchmark in egocentric vision for kitchen activities and is increasingly used in augmented reality research. It contains 55 hours of first-person video recordings captured in native kitchen environments by 32 participants across four cities and ten nationalities. It includes 11.5 million frames, Over 39,000 action segments and more than 454,000 object bounding boxes. The videos within the dataset depict non-scripted daily activities, resulting in more authentic behavior. The participants were asked to just start recording every time they entered the kitchen environment. The videos were then annotated using participants own narration recordings after filming. Therefore reflecting true authenticity and intent. These narrations support a wide range of tasks, including object interaction recognition, action segmentation and future action prediction. These are critical components for AR systems that aim to understand and respond to user behavior in real time. In addition to its foundational role in egocentric vision, the EPIC-KITCHENS dataset has continued to grow. The most notable update is the release of EPIC-KITCHENS 100 [28], which expands the dataset to 100 hours of annotated video. This version nearly doubles the original collection and introduces a much higher level of detail in the annotations through the new "pause-and-talk"feature. This allows participants to pause the video to take breaks or provide more precise narrations. The improved annotation pipeline enables a wider range of research tasks beyond basic action recognition. These include action anticipation, cross-modal retrieval and unsupervised domain adaptation, making the dataset valuable for a broader range of computer vision challenges. Similarly, the CMU-MMAC (Multimodal Activity Database) [15] offers a database with multimodal perspectives of kitchen activities. CMU-MMAC provides highly detailed data captured through a combination of video, audio, motion capture, inertial sensors and wearable devices. Although the dataset is smaller in scale and recorded in a controlled environment, therefore making it less naturalistic than the EPIC-KITCHENS dataset. It does provide accurate synchronization between different data types and enables analysis of human movements and sensor data during cooking tasks. These datasets complement each

[5]https://github.com/ultralytics/ultralytics
[6]https://roboflow.com/

other well. EPIC-KITCHENS provides large-scale, naturalistic egocentric video data ideal for training computer vision models in real-world settings. While CMU-MMAC offers a multimodal database that enables more precise analysis of human activities through synchronized sensor data and human movement data.

In summary, systems like CookAR and datasets such as EPIC-KITCHENS and CMU-MMAC highlight the growing potential of egocentric vision data in building accessible and intelligent AR applications for kitchen environments. These tools lay the groundwork for AR systems that are not only functional and context-aware but also inclusive and supportive of user safety.

While egocentric vision is central to many AR systems, other viewpoints also play an crucial role in how users understand and perform motion-guided tasks like cooking. This is demonstrated in the previously mentioned study, Perspective Matters [4], which investigates how different spatial viewpoints affect user performance in Mixed Reality environments. The study found that egocentric (first-person) perspectives generally resulted in higher accuracy and faster task completion, especially in simpler activities. However, allocentric (third-person) perspectives were found to be useful in complex tasks that required full-body coordination and therefore showed that the combination of different perspective could make instructions easier to follow. This highlights the importance of including multiple viewpoints in datasets to better support complex tasks and improve the effectiveness of instructions. Building upon these insights, Ego4D [16] and Ego-Exo4D [17] created large-scale datasets that combine first-person (ego) and third-person (exo) perspectives to explore how different viewpoints influence skill acquisition and action understanding. Ego4D [16] offers 3,670 hours of egocentric videos footage of daily life activities spanning hundreds of scenarios, captured by 931 unique camera wearers from 74 worldwide locations and 9 different countries. The dataset is annotated for tasks such as hand-object interaction and action anticipation, making it suitable for training models that need to understand how users act and react in natural environments. Meanwhile, Ego-Exo4D [17] extends this by providing synchronized egocentric and exocentric video of skilled human activities like cooking, sports, and music. Alongside video, it includes multichannel audio, eye gaze, 3D point clouds, camera poses and IMU sensor data. The dataset also features detailed language annotations, such as "expert commentary" provided by coaches and instructors. The combination of both egocentric and exocentric perspectives in the Ego-Exo4D dataset allows models to learn how actions appear from different viewpoints, making it possible to map demonstrations rom one viewpoint to another. Together, the Perspective Matters study and the egocentric and exocentric datasets demonstrate that viewpoint is not just a design decision but a critical factor that shapes how users understand, learn and perform actions in mixed reality environments. Combining both egocentric and exocentric perspectives offers significant improvements for developing more effective and adaptable systems for skill transfer and real-world guidance. However, this thesis will only focus on the egocentric perspective, as incorporating an exocentric view would be challenging task within augmented reality environments.

Another important step in using computer vision in real-time AR systems is recognizing user actions across longer periods of time. Many existing action recognition models are designed to work on short clips or individual frames, which makes it difficult to capture the full context of longer activities like those found in cooking scenarios. To solve this, the Temporal Segment Network (TSN) framework [29] introduces a new approach for action recognition that is better for tasks that unfold over time. Instead of analyzing frames continuously, TSN divides a video into a fixed number of segments and randomly samples one snippet from each segment. These snippets are then analyzed and the results are combined to recognize the overall action across the entire video. This therefore allows the system to understand longer activities without having to process every frame. This could be especially useful in AR cooking environments, as it recognizes multi-step tasks like mixing ingredients as they happen over a longer duration of time. The system can then provide more accurate feedback based on the full sequence of actions rather than individual ones.

2.5 LLMs within instruction and feedback applications

While computer vision enables AR systems to perceive and interpret the physical world, providing meaningful real-time, instructional feedback often requires a deeper understanding of user intent. Recent advances in large language models (LLMs) have created new possibilities for improving instructional systems through natural language understanding and generation. LLMs can be used in applications to provide dynamic feedback and guide users through complex tasks in a more personalized way.

One of these systems, already mentioned earlier, is Step Differences [10]. This approach uses a video-conditioned language model to compare user activity with reference instructional videos. The system is trained to identify differences in tools, techniques and outcomes between pairs of videos that show the same procedural step. For example, it can tell wether a user has stirred enough or whether they used the correct utensil. It can then provide feedback such as keep stirring till they become golden brown like in the video clip. The system compares videos of the same procedural step from the HowTo100M dataset [25] and uses large language models to generate question-answer pairs. This allows it to answer personalized questions like "what did I do wrong" or "am I done yet." This approach offers a valuable way to detect errors and give adaptive feedback based on the task context. However, the system has only been used in offline video settings and applying it within a real-time AR application remains a challenge.

Another example of using language models in instructional settings is AQuA (Automated Question Answering with Visual Anchors) [18]. This system focuses on software tutorial videos and allows users to ask specific questions about parts of the video by creating visual anchors. These anchors, together with the user's question, are combined into a prompt that is sent to a large language model. The model then provides context-aware feedback based on the visual and textual input of set prompt. AQuA demonstrates the importance of visual context for understanding user questions, especially in complex software interfaces. The study found that nearly half of all questions could not be fully interpreted without visual references, highlighting how closely visual and language-based reasoning must work together in effective instructional systems. This thesis implements a similar system where the prompt includes a visual representation in the form of multiple frames captured during the baking process and with a request to provide the next instruction, identify any mistakes in the current step and offer overall feedback. This allows the system to deliver more personalized and context-aware feedback during real-time baking tasks.

But recent developments have also begun to explore the idea of how large language models can deliver real-time, personalized instructions inside immersive environments. One example is the use of embodied AI tutors in virtual reality classrooms [19]. These tutors change their teaching approach based on students emotional and cognitive states. The system clusters students into three groups: stressed, disengaged or motivated. It then adjusts the difficulty and tone of instruction to match their needs. The tutor not only responds to inputs related to learning content but also to how students react to their instructions. As a result, students receive more personalized feedback that both improves engagement and learning outcomes. The system demonstrates notable increases in user satisfaction and knowledge retention, especially for disengaged students who showed the largest improvement in post-assessment scores. This shows how LLMs can effectively be used in real-time instructional systems by accurately prompting the user's state.

Expanding on this idea, Singh et al. [20] present another use of LLMs through a multimodal immersive agent built for an AR environment. This system tries to assist users in interactive real-world tasks like shopping. It uses GPT-4 Vision to process images of the user's surroundings and respond to both visual and text-based prompts. For instance, the agent can recommend a sofa that matches the users living room by analyzing the layout and aesthetic of the room in real time. To accomplish this, the agent first goes through an exploration phase where it learns the structure of the mobile shopping application. It uses a combination of autonomous exploration and human demonstration to understand the apps interface and register its interactive elements.

In the deployment phase, the agent applies this knowledge to complete real-time tasks in the AR application. It recognizes visual patterns, executes interface actions such as placing items in the environment and evaluates user preferences to generate recommendations. The agent shows how LLMs can support users in making decision and task completion through interactive visual feedback. This therefore highlights the potential of LLMs for guiding users through complex, real-world activities.

Beyond educational and retail purposes, large language models are also starting to show significant promise within the domain of food science. A recent review by Ma et al. [21] highlights how LLMs are being applied to tasks such as recipe generation, nutritional analysis, food safety prediction and even regulatory compliance. Systems like RecipeGPT, FoodGPT and Cook-Gen demonstrate how LLMs can convert a list of ingredients, dietary restrictions or even food images into full cooking instructions. This demonstrates the potential for LLMs to generate real-time AR cooking instructions that could adapt to available ingredients or user mistakes. The paper also explores how LLMs can support food safety through predictive analytics and inspection tasks. They analyze past contamination events and review regulatory documents and quality reports to identify potential safety risks and could suggest corrective actions. Together, these examples show that LLMs are not only powerful tools for understanding and generating instructions but also hold untapped potential in kitchen-based AR applications. Their ability to reason, personalize and adapt based on user actions makes them perfectly suited to enhance cooking instruction systems, especially when integrated with computer vision.

In summary, recent advances in large language models show potential for enhancing instructional systems through real-time, personalized feedback. LLMs have proven to be capable of guiding users through complex tasks with personalized feedback. Building on these developments, this thesis introduces an augmented reality baking assistant that combines computer vision and LLMs to deliver real-time feedback during the baking process. By interpreting user actions and generating dynamic instructions, the system aims to create a more intuitive and supportive learning experience inside the kitchen.

Hoofdstuk 3

Concept

Acquiring new skills is rarely as simple as following step-by-step instructions. Whether you are learning a craft, assembling something or preparing food, progress depends on recognizing the things you did wrong. Understanding the cause of your mistakes and learning how to correct them is a crucial part in skill development. Many existing systems can display instructions but rarely detect mistakes as they happen or explain how to correct them. They primarily focus on delivering step-by-step guidance but often offer little adaptive, personalized, real-time feedback. This means that users might just finish a task by simply following the steps without gaining the deeper understanding required to improve their technique and skills.

The proposed augmented reality (AR) baking assistant leverages technologies like artificial intelligence and AR to try and close this gap. Baking has been chosen as the domain for this thesis as it combines precise techniques, clear instructions and visible outcomes, therefore making it an ideal environment for testing a real-time interactive feedback system. The system guides novice bakers through an immersive and hands-free learning process. Unlike traditional kitchen AR application that rely on static step-by-step instructions or pre-recorded videos. This system actively monitors the user's actions and progress in real time with the aid of computer vision and the use of a large language model (LLM). It intervenes when errors occurs, explaining their causes and how to fix them. It delivers personalized, context-aware, real-time feedback directly onto the user's field of view via a head-mounted AR display. This ensures that all guidance and interaction remain within the application, without the need for additional tools or screens. The proposed approach follows the trend of the research on skill acquisition, which shows that immediate and context-aware feedback enhances the learning experience [1].

To illustrate the application's capabilities, this section presents a simple demonstration of a walkthrough of the application. The scenario uses a fictional recipe, chosen to showcase the graphical user interface (GUI) and how to interact with it. The section will focus on the user experience by showing the functionalities of the system and explaining the rationale behind key design choices. The purpose is to give a clear and high-level understanding of the system's behavior, as well as the scope of its intended use. Additionally, a brief overview of the hardware and software components will be provided without going into too much technical details.

3.1 Determining the domain of my thesis

The initial vision for this thesis began as an ambitious idea of creating a general-purpose cooking assistant capable of identifying any dish being made, recognizing user mistakes, giving solutions to those mistakes and providing contextual feedback. The goal was to create a tool that adapts to users' skill level, therefore supporting both novice and more experienced cooks in improving their kitchen skills. The idea behind this was that most cooking applications and tutorials only provide instructions and don't monitor user progress or intervene when mistakes happen. Usually, when users misread an instruction, measure ingredients incorrectly or apply a wrong technique, the mistake will only become evident at the end of the cooking process. This therefore hinders effective skill development. While such an application is appealing, it also introduces a lot of complex challenges that could interfere with our research goal. Identifying a dish as it's being prepared introduces a lot of complex questions. When does the application know which dish is being made? How can it spot early mistakes if it doesn't know the recipe yet? Addressing these issues would require a complex computer vision pipeline with large-scale AI model training and an extensive dataset spanning diverse cuisines. A task like identifying a dish would thereby shift the focus from our central research question: How effectively can large language models (LLMs) detect and respond to user mistakes in real time?

To address these challenges, the scope was narrowed and baking emerged as the perfect domain to explore an application capable of effectively answering our main research question. Baking is a form of cooking that requires precise measurements, timing and technique. It is a skill where even minor errors can significantly impact the appearance and taste of the final product, making it an ideal domain for an application focused on error detection. A cake batter whipped too long, bread dough not kneaded enough or butter melted instead of softened, these mistakes are often immediately obvious to an experienced baker and thus provide clear opportunities for real-time error detection. By also narrowing the scope to include baking recipes inside the system, the assistant can follow the instructions directly rather than trying to identify the intended dish first. This reduces system complexity and maximizes the ability for the system to detect user errors. The result is a concept for an augmented reality baking assistant that guides the user step-by-step through a recipe while simultaneously monitoring their actions through a head-mounted AR device. This assistant is designed to detect errors and intervene instantly, providing personalized, real-time feedback. The innovation for this system lies in its ability to provide a real-time feedback system for skill acquisition without the need of a human teacher.

3.2 Walkthrough

To showcase the envisioned user experience of the AR baking assistant, we demonstrate the process of preparing a simple mug cake. This recipe was chosen because it is short, accessible and involves a variety of weighing, mixing and timing steps, making it suitable for showcasing the strengths of the system. This therefore makes it a realistic demonstration of how the system assists users during their baking tasks.

The recipe consists of the following steps:

1. In a bowl, put 15 g of flour.
2. Add 18 g of sugar.
3. Add 5 g of cocoa powder.
4. Add 1/8 tsp of baking powder.
5. Add a small pinch of salt.
6. Stir dry ingredients until evenly combined.
7. Add 30 ml of milk.
8. Add 15 ml of vegetable oil.
9. Mix well until smooth and lump-free.
10. Pour mixture into a mug and microwave for 60 seconds.
11. Let cool slightly before tasting.

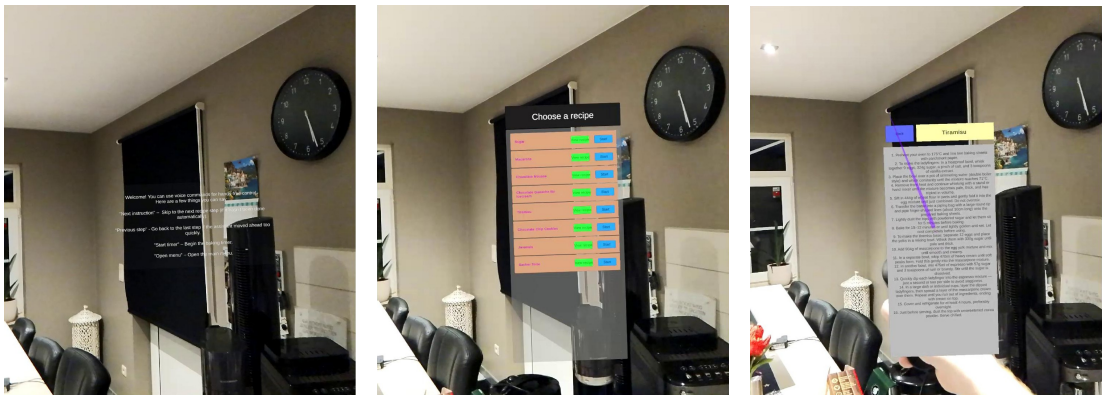
This lighthearted example proved surprisingly effective in illustrating the key features of the user interface. It showcases the application's core interactions, including ingredient measurement, timing and progressing through each step.

3.2.1 Step 1: Launching the application

When a user launches the application, they are greeted by an informational overlay explaining the available voice commands: *"open menu"*, *"start timer"*, *"next instruction"* and *"previous step"*. This feature is designed to enable hands-free interaction during baking, where hands are often covered in flour, dough or batter. Voice commands significantly improve accessibility and eliminates the need for controllers during the baking process. It therefore allows the user to focus on the task at hand, thereby improving learning capabilities.

- **Open menu:** Displays the recipe catalogue. Each recipe entry shows a the name, a *View Recipe* button and a *Start* button.
- **Next instruction/Previous step:** Enables manual control over progression in case the AI advances too fast or fails to detect step completion.
- **Start timer:** Activates a countdown timer when a recipe step requires timing (e.g. baking, resting dough).

The "open menu"command presents a scrollable catalogue of available recipes within the AR application. Each entry contains the title of a recipe, a *View Recipe* button and a *Start* button. Viewing a recipe reveals a new interface where users can read all steps and ingredients required for that specific recipe, allowing the user to prepare before actually staring the baking process. Once the user presses the *Start* button, the system briefly shows the voice command informational overlay again for five seconds before displaying the first instruction. This serves as a quick reminder, recognizing that users may not recall every single voice input option from when they launched the application.



(a) Informational overlay explaining voice commands

(b) Menu with all the system's recipes

(c) A recipe description of tiramisu

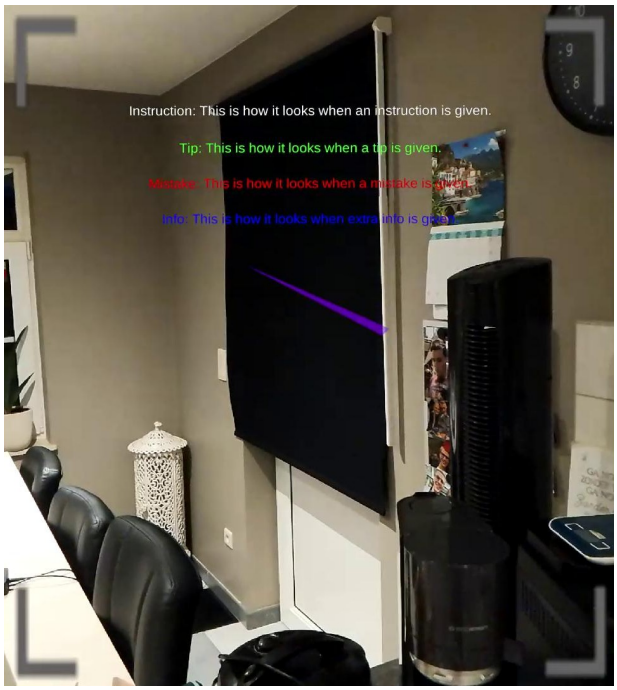
Figuur 3.1: Different views of the application interface

3.2.2 Step 2: Starting the mug cake recipe and weighing the flour

When the application starts a recipe, it displays the first instruction along with any additional feedback directly in the user's field of view. At this point, no images have been sent to the LLM yet, so the feedback is independent of the user's actual progress.

The feedback may include:

- The instruction for the current step.
- Helpful tips related to the instruction.
- Mistakes detected by the LLM.
- Additional information or suggested solutions for mistakes.



Figuur 3.2: Mockup of every potential feedback line

In our case the first instructions of our application are as followed:

- **Instruction:** 1. In a bowl, put 15 g of flour.
- **Tip:** Use a kitchen scale for accuracy when measuring the flour. Make sure the bowl is dry and clean before adding the flour.

In addition to this feedback, a dynamic weighing overlay appears in the user's field of view. This overlay consists of a horizontal progress bar that gradually fills up as the measured weight approaches the target amount, in this case 15 g. The current weight is also shown numerically beneath the progress bar itself, allowing the user to make fine adjustments. This functionality is powered by an AI-based vision library that continuously analyses the captured images for potential numbers. It then extracts those numbers in real time, ensuring that the progress bar updates automatically as the weighing takes place. As a result, the system offers a clear visual reference that guides users smoothly toward the correct measurement.

Figuur 3.3: AR view during the flour weighing event

3.2.3 Step 3: Weighing sugar, mistake detection

Upon successfully completing of the first instruction, measuring the flour, our application proceeds to the next step.

The displayed instructions are as follows::

- **Instruction:** 2. Add 18 g of sugar.
- **Tip:** Use the same scale to measure the sugar accurately. Make sure to zero the scale before adding the sugar.

At this point in the demonstration, an intentional mistake was made. Instead of adding the required 18 g of sugar, the user measures 20 g of sugar. This means that the quantity of sugar exceeded the required amount. The system therefore recognized this as a mistake and immediately updated its response.

The feedback now warns the user as follows:

- **Instruction:** 2. Add 18 g of sugar.
- **Mistake:** The sugar appears to be over 18 g.
- **Extra info:** Carefully remove some sugar until the scale reads exactly 18 g. Use a spoon to scoop out the excess gradually.

This demonstrates the application's most valuable feature: real-time error prevention. The system does not let the recipe fail due to inaccurate measurements or other mistakes. Instead it identifies errors, explains them and suggests possible solutions. By providing context-aware, personalized feedback, the application will aid users in understanding what went wrong. As a results, they will learn from their mistakes and will eventually improve their baking skills over time.

Figuur 3.4: AR view during the sugar weighing event with mistake

3.2.4 Step 4: Continuing with measuring the other ingredients

After correcting the sugar, the user continues with the remaining dry ingredients.

The application guides the process as follows:

- **Feedback 1:**
  - **Instruction:** 3. Add 5 g of cocoa powder.
  - **Tip:** Use a clean spoon to add the cocoa powder gradually to avoid spilling. Zero the scale before adding the cocoa powder.
- **Feedback 2:**
  - **Instruction:** 4. Add 1/8 tsp of baking powder.
  - **Tip:** Use a measuring spoon to ensure accuracy. Level off the baking powder with a knife for a precise measurement.
- **Feedback 3:**
  - **Instruction:** 5. Add a small pinch of salt.
  - **Tip:** A small pinch is typically between 1/16 and 1/8 of a teaspoon. Use your fingers to pinch the salt and sprinkle it evenly over the mixture.

As the user measures the cocoa powder, the application shows the weighing overlay once more, thereby offering a clear visual reference for them. The application also ensures that even seemingly trivial actions such as adding salt or measuring baking soda are accompanied by practical tips that make the baking process easier and more precise. They provide actionable guidance that improves accuracy, reduces errors and builds the user's confidence in following the recipe. By breaking down each step and offering these insights, the application transforms a basic baking task into a more informed and enjoyable experience.

Figuur 3.5: AR view displayed throughout instructions 3 to 5

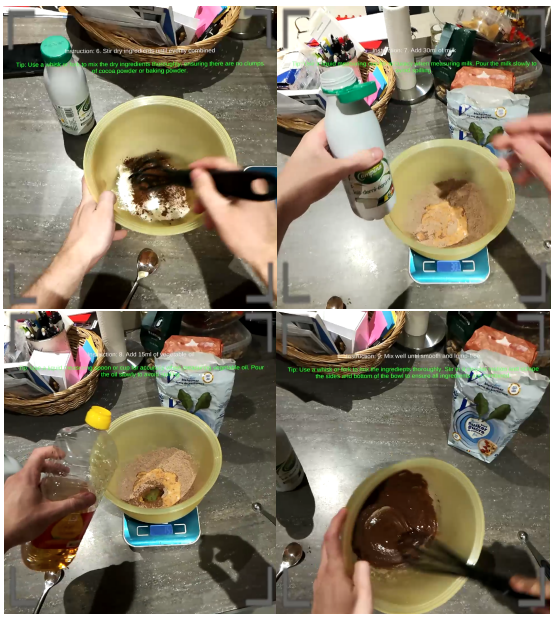
3.2.5 Step 5: Mixing and adding liquids

With the dry ingredients complete, the recipe proceeds to mixing and measuring the liquids. These steps highlight how the application can also provide qualitative feedback beyond numerical weighing,

The feedback went as follows:

- **Feedback 4:**
  - **Instruction:** 6. Stir dry ingredients until evenly combined.
  - **Tip:** Use a whisk or fork to mix thoroughly, ensuring no clumps of cocoa or baking powder remain.
- **Feedback 5:**
  - **Instruction:** 7. Add 30 ml of milk.
  - **Tip:** Use a liquid measuring cup for accuracy when measuring milk. Pour slowly to avoid spilling.
- **Feedback 6:**
  - **Instruction:** 8. Add 15 ml of vegetable oil.
  - **Tip:** Use a liquid measuring spoon or cup for accuracy. Pour the oil gradually.
- **Feedback 7:**
  - **Instruction:** 9. Mix well until smooth and lump-free.
  - **Tip:** Use a whisk or fork to stir in a circular motion, scraping the sides and bottom of the bowl to ensure all ingredients are fully combined.

Once again, the application displays the weighing overlay while the user measures the milk and vegetable oil. During the mixing steps, the system also highlights proper technique and consistency, showing how the application actively supports skill development and reinforces correct methods.



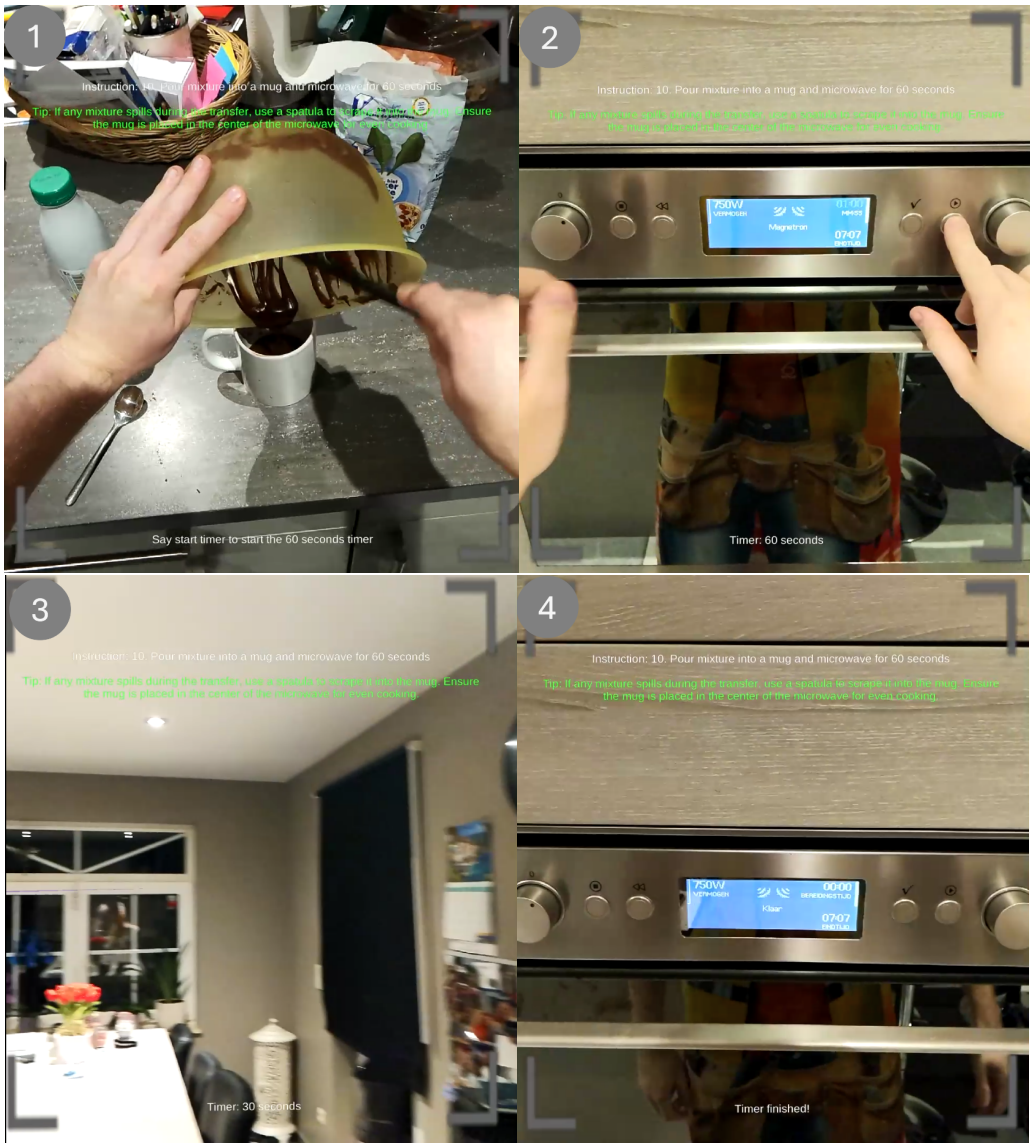
Figuur 3.6: AR view displayed throughout instructions 6 to 9

3.2.6 Step 6: Pouring the mixture and timing the microwave

Once the batter is ready, the system displays the following instructions:

- **Instruction:** 10. Pour mixture into a mug and microwave for 60 seconds
- **Tip:** If any mixture spills during the transfer, use a spatula to scrape it into the mug. Ensure the mug is placed in the center of the microwave for even cooking.

To assist the user during this step, a countdown overlay appears at the bottom of the display. The timer is automatically set to the duration specified in the instruction and can be started using the *"start timer"* voice command. The timer will then count down and remain visible until it reaches zero, at which point the timer will tell the user it has finished. This overlay was designed with the intension to improve the user's quality of life by reducing unnecessary distractions and allowing them to remain fully engaged in the baking process.



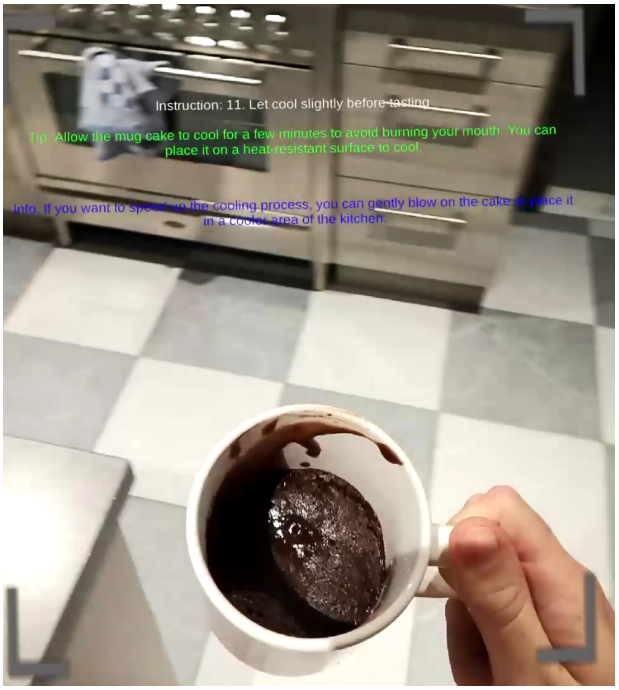
Figuur 3.7: AR view during the timer event

3.2.7 Step 7: Finishing the recipe

After removing the mug from the microwave, the application presents the final instructions:

- **Instruction:** 11. Let the cake cool slightly before tasting
- **Tip:** Allow the mug cake to cool for a few minutes to avoid burning your mouth.
- **Extra info:** If you want to speed up the cooling process, you can gently blow on the cake or place it in a cooler area of the kitchen.

This step illustrates how the application continues to provide meaningful support even after the core baking process is finished.



Figuur 3.8: AR view when the baking process had been completed

Through the mug cake example, the walkthrough illustrates how the application guides the baking process from start to finish. It helps measure ingredients with precision, using a progress bar for an extra visual reference. If a mistake happens, the application detects it in real time and offers corrective feedback. At each step, it provides contextual tips and extra information tailored to the current instruction, ensuring the user feels supported throughout. The experience remains hands-free thanks to voice commands. And built-in timers ensure that every stage of baking stays perfectly on track.

3.3 Hardware and interaction design

The hardware used for this system is the magic leap 2 headset, which serves both as the display and the main input capture device. This head-mounted AR headset was selected to align with the system's goal of providing a fully immersive, hands-free baking assistant. The Magic Leap 2 offers an egocentric viewpoint and therefore captures exactly what the user sees and does. This perspective allows the LLM to analyze actions without the need for additional cameras or kitchen based-sensors. The system always has the most relevant visual perspective for detecting potential user mistakes because the headset's camera always aligns with the user's gaze. The AR display also helps the user by keeping all instructions and feedback directly within the user's line of sight, avoiding the need to use separate equipment like material instructions or timers. This approach prevents interruption and supports the learning progress by keeping the user's focus on the task rather than on separate screens or printed instructions. Alternative configurations were considered during the design phase, such as combining a head-mounted GoPro with a tablet or external monitor. While technically feasible, these options introduced several drawbacks, higher hardware requirements, more complex installation and reduced immersion due to the need to glance away from the workspace. By selecting a self-contained AR headset, the system achieves a balance between portability, simplicity, and functionality. The Magic Leap 2 delivers the hardware capabilities necessary for real-time vision processing while also supporting a context-aware user interface. This combination makes it well-suited for the demands of an interactive, error detecting, real-time baking assistant.

Hoofdstuk 4

Implementation

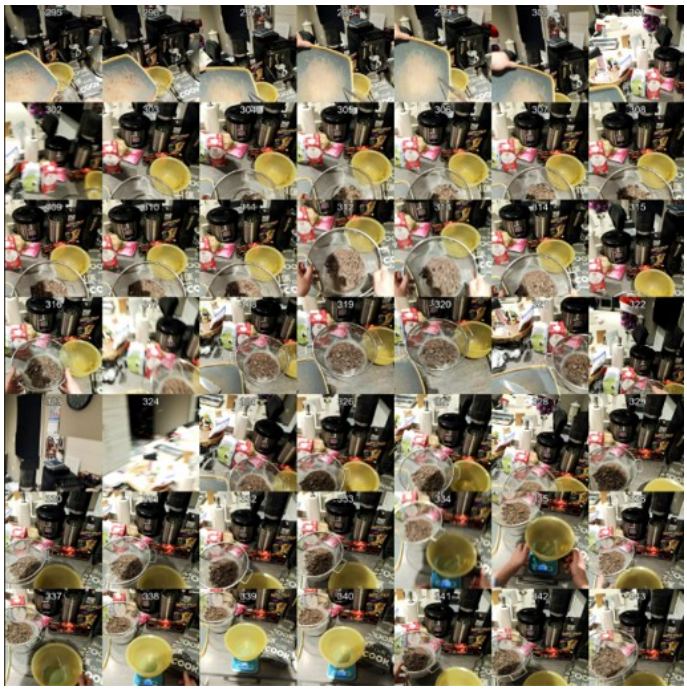
The augmented reality baking assistant was developed in multiple stages, beginning with offline experiments and eventually turning into a fully functional proof-of-concept system. The process combined early explorations in both computer vision and large language models (LLMs) with careful consideration of the constraints of our hardware and its performance. This chapter outlines the development in chronological order, detailing the purpose, design choices, challenges faced and solutions applied at each stage. The technical architecture and design rationale are also discussed in detail to clarify how each component fits within the overall system.

4.1 Testing the LLM with static images

The first step was to determine whether a large language model could reliably identify mistakes based on visual inputs in a baking process. Two Swiss role cakes were prepared to test this. One was made without any mistakes while the other had a skill-based error that novice cooks might easily make. We then documented each important action inside the recipe with images during preparation, such as rolling the cake, mixing the batter and whipping the eggs. Using a web browser, we then submitted these images to OpenAI's GPT-4 model via the Assistant Playground and experimented with a variety of prompts. Here we tested if the model was capable of identifying the dish, detect any mistake and provide relevant feedback. GPT-4 was selected as the large language model for this project after considering alternative OpenAI models, as well as other popular systems like Gemini and also open source solutions. GPT-4 is one of the most advanced commercially available multimodal systems. With an ability to process both text and images simultaneously, which was essential for our application. Compared to other models like GPT-3.5 or Gemini, GPT-4 demonstrated higher accuracy in identifying visual details and providing relevant feedback during testing. While open-source solutions may offer advantages in terms of cost and customization, they generally require significant computational resources to achieve comparable performance. Additionally, GPT-4 provided thorough documentation, which simplified the integration with our application. In the playground we also tested different numbers of images per request, ranging from three to ten, to observe how performance changed as the amount of visual information varied. In some cases, we deliberately excluded certain frames to simulate real-world conditions where the application might not capture every action. In all configurations, the model eventually recognized both the dish and the intentional mistake. Although it sometimes required multiple requests to identify the dish, it still performed decently accurate even when important actions where missing. These results confirm the viability of visual mistake detection. But our initial approached revealed a more critical limitation. The system can't always recognize the dish instantaneously, therefore making it impossible to detect early mistakes in the cooking process.

This insight led us to narrow our scope to baking with recipes, as described in Section 3.1. On the other hand it also raised a key design question: how should the system decide which frames are valuable enough to capture and send to the LLM for analysis?

Several strategies were considered, including more advanced approaches such as action recognition through the use of hand tracking and object detection models like YOLOv5 [13]. However, these methods would have required extensive model training, additional computational resources and a significant amount of time. That's why we opted for a more nuanced approach in which frames would be captured at fixed intervals during the baking process. These frames were then processed and assembled into one singular combined image and given to the LLM. To create this combined image we arranged the captured frames into a singular grid layout. We then labeled each frame with the number of the order they were taken in. This could help the LLM to understand the order of events and improve its analysis. The grid composition was implemented in Python using the Pillow[1] and OpenCV[2] libraries which allowed us to automate the placement and labeling of frames. We evaluated this by recording a chocolate mousse preparation from an egocentric viewpoint. The recording was captured with the Magic Leap 2 camera and its recording function. This allowed us to simulate how our application would eventually look and evaluate how well the LLM could process the frames in a realistic scenario. We tested several grid configurations to find a balance between image clarity and the number of frames displayed. In the end we chose a 7×7 grid because it held the most frames while still keeping enough detail for the model to analyze. This setup gave the model more process information in a single prompt. Which therefore reduces the need for multiple requests and avoids overwhelming the LLM with inputs.



Figuur 4.1: Example of a 7×7 grid layout generated from frames of a chocolate mousse preparation which was used as input for GPT-4

[1] https://pillow.readthedocs.io/en/stable/
[2] https://docs.opencv.org/4.x/

We then developed a Python program using the OpenAI[3] library to interact with the GPT-4 model directly via code instead of having to use a browser. Initially, we created a response model that could receive images and return textual feedback. However, this model did not automatically keep track of previous requests. This meant that the conversation history had to be manually included with each new request, which was inconvenient and inefficient. To improve usability, we converted this into an assistant model. The assistant automatically maintains threads that store the history of all requests and responses. This allowed for a smoother and more continuous evaluation of the baking process. Early trials used with this setup produced promising results. The outputs were initially unstructured text, which was difficult to integrate with an interactive AR interface. To address this, we redesigned the code to ensure that outputs were structured exclusively in a JSON schema format, allowing for predictable parsing and easier integration into the application. The JSON structure was designed as follows:

```
{
  "instruction": "<The instruction the user is currently on from the recipe>",
  "tips": "<Any relevant tips the user might find useful when doing the current step of
     the recipe>",
  "mistake": "<Point out the mistake if the user makes any. If no mistakes are visible,
     write null>",
  "solution_extra_info": "<Additional context or explanation about the step or how to
    fix the mistake>",
  "weight": "<If an ingredient needs to be weighed, specify the target weight in grams
    >",
  "timer": "<If the user needs to wait, specify the time in seconds>",
  "completion_time": "<Estimate how many seconds the user needs for the next step when
    should the next image be provided?>"
}
```

This structure allows the system to give clear and effective guidance. It can detect mistakes, provide instructions, give extra information or solutions and offer timing and measurement cues for the UI elements.

During testing, we also noticed that too many requests in a single assistant thread could overload the model. This could therefore reduce performance or produce incorrect responses. To address this, we implemented a summarization step. The assistant is prompted to generate a summary of the progress and feedback in the current thread every ten request. This summary is then carried over to a new thread, thereby preserving context while also preventing overload, allowing the system to maintain continuity in the recipe evaluation.

[3]platform.openai.com

4.2 Hardware integration

After laying the groundwork for mistake detection with large language models and validating their practicality with some examples. The next stage was to integrate them into a functional augmented reality application. Instead of analyzing pre-recorded videos, the system now had to operate in real time on an AR headset, continuously capturing images, processing them and providing personalized, context aware feedback. Choosing the right combination of hardware and software was therefore a critical step, as it would directly impact the system's speed, accuracy and overall usability.

4.2.1 Choosing the AR headset

The Magic Leap 2 headset was selected as our primary device for the application. As it offers a combination of an AR display with a built-in camera, providing an egocentric viewpoint for our computer vision analysis. It also supports hands-free interaction through the use of voice commands. This allows users to interact with the system without having to use their hands, which could be useful in a baking environment where hands are often occupied. Compared to other headsets such as the HoloLens2, which offers similar AR capabilities, the Magic Leap 2 provides a wider field of view, brighter displays for well-lit rooms like a kitchen and a lighter, more comfortable design for prolonged use. The Magic Leap 2 thus represents the perfect choice. However, adopting this device also introduced some challenges, particularly with the head-mounted camera. One of those challenges was the offset between the user's gaze and the camera's perspective. The Magic Leap's camera sits above the eyes, therefore didn't always show the users exact point of view. During initial tests with the chocolate mousse recipe, this misalignment caused some steps to be filmed incorrectly.

To address this issue, we decided to introduce a rectangular frame overlay onto the AR interface that corresponded exactly with the camera's point of view. This ensures that users can accurately record the baking process as long as they keep their actions within the frame. Although this solution requires conscious efforts from the user, it provided a practical fix that ensures reliable capture for mistake detection.

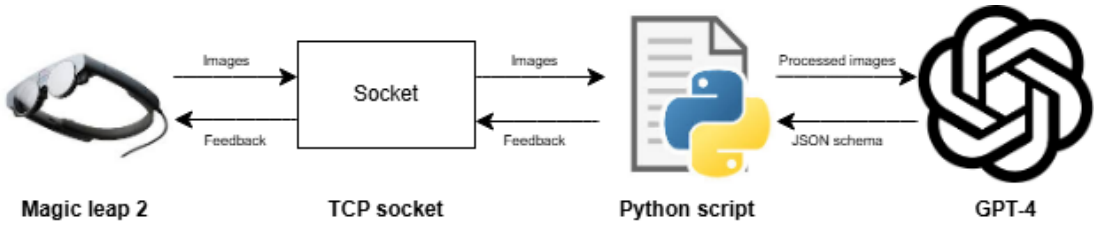
Figuur 4.2: The Magic Leap 2 headset

4.2.2 Development environment

Once the headset choice was finalized, the next step was building the application environment itself. Unity was chosen as the development platform due to its strong support for Magic Leap's SDK, its flexibility and the abundance of community-created examples that could support the system's development. The Unity engine enables rapid prototyping of user interfaces and provides direct access to the Magic Leaps `MLCamera` API to capture images. However, while the `MLCamera` API offered the necessary functionality, it also introduced an important limitation. The Magic Leap 2 only supports a single capture mode at any given time, meaning that it was not possible to record videos and capture images at the same time. As a result, we were unable to record demos in real time and had to rely on manually inserting feedback for demo recording, instead of depending on the application to do this automatically.

With the image capture process defined, the next design challenge was determining how to transform the raw frames into the proposed combined images. An early attempt to implement this in Unity quickly exposed some performance bottlenecks. Unity's `SetPixels()` function, which modifies texture data pixel-by-pixel, introduced a significant delay for high-resolution frames like ours. Because the method modifies textures on a pixel-by-pixel basis, the processing time scaled poorly as the combined image size increased. This therefore resulted in long delays that were incompatible with near real-time feedback. In contrast, our Python script had already demonstrated strong performance using Pillow[4] for grid composition and OpenCV[5] for pixel manipulation. OpenCV, in particular, leverages optimized C and C++ backends for its pixel operations, making it significantly faster than Unity's. This performance gap made it clear that preprocessing needed to remain in Python.

The next step was figuring out how to bridge Unity's AR interface with Python's optimized backend. An initial exploration into Python scripting directly within Unity proved unproductive, as the integration seemed difficult and required considerable efforts for minimal results. After losing significant time on this approach, a different solution was implemented using a TCP socket that served as a bridge between Unity and a Python server. In this setup, Unity periodically captured and encoded frames using the `MLCamera` API. These frames were then transmitted over the socket connection to a Python script running externally on a laptop. To ensure reliable communication, the socket protocol included a fixed-length header specifying the payload size, which guaranteed complete data transmission. Both Unity and Python also handled socket communication on separate threads, thereby preventing the capture loop to be be blocked. The Python script then performed grid composition, labeling and OpenAI API requests. Once the GPT-4 model returned a structured JSON schema response, the script parsed the result and forwarded it back through the socket to Unity. Unity then updated the AR overlays with the feedback received. This architecture ensured that the most computationally expensive tasks were handled by Python's optimized libraries while Unity focused on rendering and user interaction. Although the network-based solution introduced some latency, it was acceptable within the proof-of-concept scope of the project.



Figuur 4.3: Integration of the full application

[4]https://pillow.readthedocs.io/en/stable/
[5]https://docs.opencv.org/4.x/

In the final result, Unity transmitted the active recipe to the Python server at startup. Images were then captured at one-second intervals and stored until forty-nine frames had been collected. These were then assembled into a 7×7 grid, labeled with their order number and sent to GPT-4 model. The returned JSON response was then parsed in Python and forwarded to Unity, which updated the AR overlays accordingly. In addition, the `completion_time` field of the JSON schema was used as a parameter to dynamically change the capture frequency of the frames. In earlier iterations, the system captured frames at one-second intervals regardless of the task at hand. This approach often produced redundant captures during long steps and reduced responsiveness during short actions. To address this, the JSON schema was extended with a `completion_time` variable that estimated the duration of the current instruction. This value, generated by the LLM, was then transmitted to Unity and used to dynamically change the capture interval. This strategy improved the user experience but also reduced computational overhead and API costs. By leveraging Python's optimized backend alongside Unity's AR interface, this architecture allowed the system to provide near real-time, context-aware feedback to users, demonstrating the effectiveness of our proof of concept approach.

4.3 Graphical user interface

Once the basic system pipeline had been implemented and validated, the next step was to design and implement a graphical user interface (GUI) that would transform the prototype into a usable application. The GUI was developed inside Unity and directly integrated with the existing socket-based communication algorithm. The main purpose of the interface was to provide a more intuitive and user-friendly experience by bringing all system feedback directly into the user's field of view. The design aimed to minimize distraction while still delivering as much helpful information as possible. In addition to displaying textual feedback, the interface also incorporated features like a visual timer for timed events and a weighing progress bar for ingredient measurements. These elements ensure that the system not only communicates feedback to the user but also supports them with task-specific visualizations, making the application more immersive and useful in real baking scenarios.

The first stage of development focused on implementing the key UI elements, already mentioned in our concept chapter, for helping the user navigate the application. These include:

- **Menu catalogue:** A scrollable list of recipes with a *View Recipe* button for reading instructions and a *Start* button for launching the baking process of that recipe.
- **Informational overlay:** A start-up overlay that reminded the user of the available voice commands such as *"open menu"*, *"next instruction"*, *"previous step"* and *"start timer"*.
- **Instruction display:** The current step of the recipe and any additional tips, extra information, mistakes or solutions provided by the LLM, always rendered in the user's field of view.

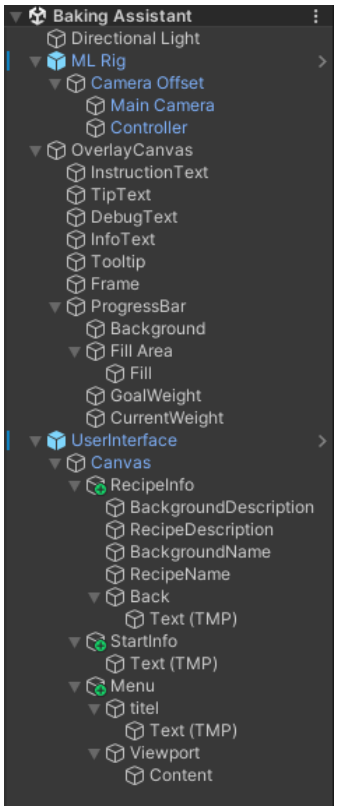
All of these components were implemented using Unity's built-in UI system (Canvas, TextMeshPro, buttons, etc.) and configured for AR by rendering them as world-space or screen space canvases positioned within the Magic Leap 2's field of view. Voice commands were also linked to Unity's input system so that users could effectively use all commands displayed in the informational overlay.

Beyond the core UI elements, additional visual overlays were added to enhance the application's usability. One such overlay was a visual timer, when the LLM would detect an instruction that required timing, it would return a `timer` field within the JSON schema specifying the duration of time in seconds. This information was then forwarded to unity, where a basic countdown timer was displayed in the user's field of view. The timer could then be started by the user through the *"start timer"* voice command. We also explored the possibility of automatically starting the timer by asking the LLM to identify the exact frame where a timing event should begin. This would therefore allow us to calculate the precise moment the timer should start. However, GPT-

4 proved to be unreliable for this task, frequently ignoring the request or returning incorrect frames. The current manual activation via voice command therefore provided the most practical solution.

In addition to the timer, a weighing progress bar was also implemented to guide users during ingredient measurement. When an instruction required weighing, the LLM would include a `weight` field inside the JSON schema specifying the exact target weight in grams. Unity then displayed a progress bar directly onto the user's field of view, showing the target weight numerically at the top and the current weight at the bottom of the bar. To determine the current weight and dynamically change the progress bar, we experimented with integrating a Python based library named `easyocr` for Optical Character Recognition. This library extracts text from images and could therefore be used to read numbers displayed on a kitchen scale in real time. While some results were promising, the performance overall was very inconsistent. In some cases the system could accurately determine the number displayed on the kitchen scale, while in others it misread the digits entirely. Additionally, the Optical Character Recognition also introduced latency, which made it unreliable for effective real-time use. This unreliability prevented the weighing overlay from functioning consistently. As a result, the progress bar was kept for illustrative purposes rather than as a fully robust, functional feature.

Despite some of the technical limitations encountered, the implemented graphical user interface completed the transition from a technical prototype into a fully intractable AR application. Although certain features like automated timers and ingredient measurement detection were limited by the performance of current AI libraries. The GUI successfully unified the key functionalities into a cohesive and functional system. This allowed the application to deliver step-by-step instructions, real-time, context-aware feedback and additional visual features that support the user in a natural and immersive way.



Figuur 4.4: Unity structure for graphical user interface

Hoofdstuk 5

Evaluation

After implementing the augmented reality baking assistant and utilizing computer vision and large language model components. The next step was to evaluate how well the system actually performed. The evaluation focused on determining whether the application meets its central goal: providing accurate, context-aware, personalized, real-time feedback that helps users detect and correct mistakes during the baking process.

A key part of this evaluation was testing how different LLM instruction prompts influenced the accuracy and consistency of the feedback. Multiple prompt formulations were tried, each aiming to balance strict JSON output formatting with the flexibility needed for the model to interpret diverse baking scenarios. While none of the tested prompts produced the promising outcome we had hoped for. There was still one prompt that was produced who consistently had the most reliable and user friendly responses. This final version told the model to act as a baking assistant and analyze a set of timestamped images from the user's baking process. It would then return a structured JSON schema object with the current instruction, relevant tips, detected mistakes, extra info or solutions and any timing or weighing requirements.

This prompt gave the best balance between strict formatting and useful content generation, therefore making it suitable for the examples presented in the remainder of this section. The evaluation process consisted of both controlled fictional recipes and real-world baking recipes. The controlled scenarios enabled a deeper examination of the LLM's ability to detect errors, while the real-world tests assessed how well the system performs in practical baking situations. This dual approach allowed us to thoroughly assess the assistant's accuracy in identifying mistakes as well as its practical usefulness in real baking contexts.

This prompt went as follows:

Baking Assistant Instructions

You are a baking assistant dedicated to helping the user improve their baking skills.
Your task is to analyze multiple images with timestamps of the user's baking process.
The first thing the user will give is the recipe.
After this, you will receive one or more images, and your task is to provide feedback based on those images.
This feedback should be provided in the following JSON format:

```
{
  "instruction": "<The instruction the user is currently on from the recipe>",
  "tips": "<Any relevant tips the user might find useful when doing the current
    step of the recipe>",
  "mistake": "<Point out the mistake if the user makes any. If no mistakes are
    visible, write null>",
  "solution_extra_info": "<Additional context or explanation about the step or
    how to fix the mistake>",
  "weight": "<If an ingredient needs to be weighed, specify the target weight in
     grams>",
  "timer": "<If the user needs to wait, specify the time in seconds>",
  "completion_time": "<Estimate how many seconds the user needs for the next
    step when should the next image be provided?>"
}
```

You must always give feedback in this exact format.
If a response was already given in previous answers or you don't have a response for a specific field, write null or omit it (do not repeat fields).
Only update the instruction when you are certain the user has completed the current step based on the images. If uncertain or ambiguous, do not advance.
Never mention which specific image something relates to the user does not have that information.
The user may also manually tell you to skip to the next instruction, indicating that the current step is complete.
The most important fields are instruction and mistake be very careful, as incorrect feedback here can negatively affect the outcome.
You must reply only with a single valid JSON object and nothing else. Do not repeat or include the recipe or additional commentary.

5.1 Low level fictional Recipe Error Detection Tests

To evaluate the system's capabilities to detect common user mistakes, a series of tests were conducted using variations of a fictional sugar-making recipe.

The recipe goes as followed:

1. Place 100 g of sugar in a bowl.
2. Set a timer for 1 minute to allow the sugar to "oxidize".
3. After the timer ends, return the sugar to its packet.
4. You are done! You have "made" sugar.

This recipe was chosen because it allowed for an easy, repeatable scenario without the need of actually having to bake. Each test was performed using pre-recorded videos processed through the final LLM prompt described earlier in this section. The correct execution of the fictional recipe was first evaluated to confirm that the assistant behaved as intended. It correctly identified each instruction within the video without producing false positive mistakes.

We then tested how well the system could spot quantity-related mistakes by tweaking the first step of the fictional sugar-making recipe. In both tests, the only thing we changed was the amount of sugar written in the recipe. The video of the preparation stayed exactly the same. This way, we could see if the assistant would notice when the measured amount didn't match up with what the recipe had said.

Under-measurement according to recipe: The recipe was modified to require 50 grams of sugar, while the recorded video showed 100 gram being measured into the bowl. The system accurately identified this mismatch and returned:

```
mistake: The sugar in the bowl appears to be more than 50g.
```

This was the intended behavior, as the assistant identified that the quantity exceeded the recipe requirement.

Over-measurement according to recipe: The recipe was modified to require 200 grams of sugar, while the recorded video showed 100 gram being measured into the bowl. The system accurately identified this mismatch and returned::

```
mistake: The sugar is not yet measured to 200g.
```

However, despite correctly identifying the mismatch, the system still advanced to the next step as soon as the user continued with the recipe. This highlighted a recurring issue in several of the tested prompts where the assistant sometimes skips ahead after detecting an error when the input suggests that the user has moved on, rather than prompting them to correct the mistake first.

To explore the limits of the visual detection capabilities of the LLM, a variation was made in which the recipe called for salt instead of sugar. While the recorded video still showed sugar being used. The system did not recognize the mistake, most likely due to the high visual similarity between granulated sugar and table salt. This highlights the limitation the LLM has in telling certain ingredients apart when relying on visual cues, even if their packaging is present.

In the last variation of this recipe a new first step was added before weighing the 50 grams of sugar. It instructed the user to measure 500ml of water into a cup. This time the system correctly detected an error and did not advance to the next step:

```
mistake: The user is using a bowl instead of a cup to measure water.
mistake: The user is still using a bowl instead of a cup to measure water.
```

Unlike the earlier variation, the assistant held its position on the current instruction instead of skipping ahead, showing that it could maintain step focus when a persistent mistake was detected.

The fictional recipe tests confirmed that the assistant could detect certain quantity-based errors reliably when the visual difference was clear. However, they also revealed two notable limitations of this system. A tendency to advance through the recipe even after detecting a mistake if the user continued to subsequent actions. And the difficulty in identifying ingredient substitutions where visual features were too similar to differentiate without additional context.

5.2 Real-world recipe tests without intentional mistakes

To assess how the system performs under ideal conditions, a series of tests were conducted using real baking recipes executed without intentionally introducing errors. These tests aimed to determine whether the assistant could correctly follow the user's progress, provide valuable tips/info and advance through the recipe at an appropriate pace without generating false positive mistakes that might hinder the learning process.

The first recipe tested was a chocolate mousse prepared without deviations from the original recipe. As expected, the assistant progressed smoothly through the steps without advancing too quickly or falling behind. It consistently provided relevant tips and additional information, contributing positively to the cooking experience. For example, during the chocolate melting stage, it advised:

```
tip: Ensure the water in the pot does not touch the bottom of the bowl.
Stir the chocolate gently and continuously to prevent burning.
```

This is in particular a valuable insight for novice cooks, who may not be aware that direct heat or contact with boiling water could easily burn chocolate. The assistant also supplied accurate timer and weight values throughout the process, allowing the user interface to function as intended. However, there were still two unexpected errors that occurred. At one point, the system flagged:

```
mistake: It seems that some yolk might have gotten into the egg whites.
```

This was a false positive mistake in the early stages of baking progress. Because the eggs actually hadn't been used yet. Later in the process, the system also reported this mistake:

```
mistake: It appears that the mousse is being divided into fewer than 8 servings.
```

This second mistake was indeed correct, since I deviated from the exact instructions provided in the recipe and put the chocolate mousse inside fewer than 8 servings. Apart from these two instances, no other mistakes were detected and the system consistently provided accurate step tracking, relevant tips and helpful extra information. This therefore demonstrates the intended behavior for a recipe executed without any mistakes.

The second recipe tested was macarons, a notoriously challenging baking recipe where precise technique is crucial. In the no mistake version, the assistant again provided valuable tips and helpful information. However, its mistake detection was more inconsistent than in the chocolate mousse test. Four mistakes were reported that had not occurred during the cooking process:

```
mistake: The user has not yet added the espresso powder as per the images.
```

When this mistake was shown the powder had already been added to the dish.

```
mistake: The almond flour does not appear to have been weighed accurately before
being added to the bowl.
```

This was generated before weighing took place, making it irrelevant.

```
mistake: You have added the ingredients to the food processor and pulsed,
but the sugar hasn't dissolved completely.
```

In this step, the sugar only needed to be ground to a fine powder and didn't have to dissolve, therefore making the feedback very misleading. While such false positives are unlikely to cause major confusion for experienced bakers, they could have the potential to mislead novice users.

And the final mistake reported during this test was very debatable:

```
mistake: Too many attempts to pulse the mixture, possibly over-processing.
```

In this instance the system was right because the mixture was indeed pulsed a lot. This was due to the use of a blender instead of a food processor recommended by the recipe. However, this wasn't really a mistake but rather an intentional choice because the recipe stated that the ingredient had to be grounded into a fine powder. And because there wasn't a food processor at hand in the kitchen we decide to ground it with a blender. This highlights a recurring limitation of the system where it has a difficult time distinguishing between visually similar tools or ingredients (salt vs sugar and blender vs food processor).

The final no-mistake test was conducted with a chocolate chip cookie recipe. The assistant performed exactly as intended. It offered practical tips and relevant additional information without generating any false positive mistakes or missing instruction steps. This demonstrates that for a more straightforward recipes, the system is capable of delivering accurate real-time feedback.

The assistant generally succeeded in providing accurate instructions, pacing its step progression appropriately and delivering useful tips during the evaluation of recipes with no mistakes. While occasional false positives were observed, their impact on the overall baking process was minimal in most cases. Nevertheless, the macaron test revealed that such errors have the potential to mislead less experienced users, therefore highlighting the need for further refinements of the system.

5.3 Real-World Recipe Tests With Intentional Mistakes

To evaluate the system's effectiveness in detecting and responding to actual user errors during an actual real life baking scenario. A series of tests were conducted with real baking recipes in which deliberate mistakes were introduced. The aim was to examine whether the assistant could correctly identify these mistakes, avoid advancing prematurely through recipe steps and provide helpful correctional guidance.

In the first test, a baking video of cookies was used, but instead of giving the system a corresponding cookie recipe, it was given a recipe for ice cream ganache. While there is an overlap in the first instruction step, chopping chocolate, the overall recipe and techniques used are significantly different. The system therefore generated numerous mistakes, for example:

```
mistake: The cream has not been poured over the chocolate yet.
```

```
mistake: The mixture is not yet smooth and glossy, indicating it
hasn't been stirred enough or the chocolate hasn't fully melted.
```

These detections reflect the intended behavior of identifying differences between the provided recipe and the observed user actions. However, despite only the first step (chopping the chocolate) actually being completed, the system still progressed through numerous instructions. This problem of advancing without resolving errors therefore happened again, showing that the step progression logic doesn't verify that every necessary action is truly completed.

The final test involved a macaron recipe in which three significant mistakes were deliberately introduced:

- Using an incorrect egg amount.
- Burning the sugar syrup.
- Over folding the cookie batter.

The system performed reasonably well. It provided useful tips and extra context with each step. However, the instruction responses were very cluttered. Instructions appeared out of sequence multiple times during the baking progress. The LLM likely became overwhelmed due to the combination of that many mistake and their severity. In terms of mistake detection, two of the three errors were identified, but unfortunately didn't detect the incorrect egg amount:

```
mistake: The images show the batter, but it's unclear if the folding
technique is correct. Ensure the folding is gentle to maintain the
batter's consistency.
```

Here, the system correctly recognized that the batter had been over-folded.

```
mistake: The user should ensure that they stop stirring once the sugar
has dissolved and the mixture is heating to prevent crystallization.
mistake: The image sequence does not show a thermometer being used,
which is important for accurately reaching the desired syrup
temperature of 115 degree C.
```

While these observations do not directly state that the syrup was burned, they strongly suggest it. This mistake also provides the user with a way to fix the issue by using a thermometer. It is important to note that other easily correctable mistakes, such as incorrect quantities, did not trigger similar solution-oriented feedback.

These tests demonstrated that the assistant can detect and comment on some process errors during the baking process and could sometimes even offer solutions. However, challenges in controlling step progression, detecting errors and generating consistent solutions are still evident. Particularly in situations where multiple cooking mistakes occurred.

Hoofdstuk 6

Discussion

This thesis successfully demonstrates the potential of combining computer vision and large language models for an interactive augmented reality baking application. As a proof of concept, the system showed across both fictional and real-world examples that real-time guidance during complex tasks is possible. When the recipe progression matched expectations, the system was able to provide accurate instructions, practical tips and appropriate timing and weighing information. In simple scenarios the application behaved reliable and contributed to a better learning experience.

6.1 Limitations

However, the systems also highlighted some important limitations. One of those limitations lies in the construction of instruction prompts. Despite efforts to fine-tune prompts, it was very clear that the LLM often struggled to interpret prompt instructions correctly. The evaluation also revealed that the model had difficulties in differentiating visually similar items. For example, it confused salt with sugar and mixed up a blender with a food processor. These issues sometimes caused mistakes to be undetected and reduced confidence in the system's feedback. The LLM was also prone to generating false negatives when detecting mistakes. These false alarms could very well mislead novice users. On the other hand, some mistakes, like using the wrong amount of eggs, were completely overlooked. This therefore reveals a clear gaps in the system's error detection capabilities. In addition, the assistant sometimes moved on to the next instruction before the current step was completed. This created a lot of confusion, especially when mistakes were missed or not fully solved. And when faced with too many mistakes, the system got overwhelmed and started cluttering its responses. Another limitation was that while the system sometimes provided helpful solutions, such as using a thermometer to fix sugar syrup issues, it did not consistently offer guidance for other mistakes. These issues collectively reveal clear gaps in the system's error detection and guidance capabilities. Therefore reducing user confidence and potentially misleading novice users.

Beyond these broader limitations, certain features also proved to be more redundant than others. The weighing visualization, for instance, did not contribute significantly to the system's learning experience. While intended to provide additional feedback, its actual use case was limited. Even if the feature had worked flawlessly, its contribution to error detection and user feedback would have remained minimal. In hindsight, this component added unnecessary complexity to the system without actually improving the user experience.

Another design decision that could potentially give mixed results was the implementation of the dynamic capture intervals. While intended to optimize the performance of the application, this mechanism could introduce potential risks. For example, in cases where a single step contains multiple small, error-prone actions, the system could incorrectly assume that the step was completed without any mistakes. As a result, errors could go undetected, undermining the reliability of mistake detection.

One practical limitation is that the system currently depends on custom-written recipes, which limits scalability. Recipes must be added manually, making it harder to expand the recipe catalogue or support user-generated content. This prevents broader adoption and highlights the need for a mechanism that enables more flexible and scalable recipe input.

Overall, the system demonstrates clear potential as a real-time baking assistant. In its current form, it should be regarded as a proof of concept that establishes a foundation for future development. Addressing the identified limitations in error detection, feedback consistency and recipe scalability will be essential for building a more reliable and widely usable application.

6.2 Future work

Future iterations of this system could benefit both from advances in AI technologies, as well as improvements in our application's design. A clear improvement would come from employing a more capable LLM such as GPT-5 or future successors, which are likely to offer better reasoning, contextual awareness and better differentiation between ingredients, tools and actions. These improvements would therefore directly address many of our current issues with instructions, mistake detection and feedback.

Another promising avenue is to integrate an action recognition model with hand tracking and object detection models. By identifying key actions in real time, the system could capture only the most relevant frames instead of capturing them over a fixed interval. This would not only improve mistake detection but also reduce the processing load. Extending on this idea, future AI systems could also become capable of interpreting short video clips rather than only static images and text. This could therefore provide an even richer context evaluation, resulting in more natural and accurate guidance.

In terms of prompt engineering, the current approach relied heavily on trial and error. Developing a more systematic strategy, possibly supported by automated prompt generation or reinforcement learning techniques, could significantly improve the large language model's performance. Similarly, the system's speed could also be enhanced by simplifying the current architecture. For example, removing the socket connection and integrating the logic directly into Unity trough Python scripting or another lightweight backend would reduce latency and therefore improve responsiveness.

Recipe flexibility represents another key improvement area. A future version could allow users to record or input their own recipes, with the AI automatically converting them into a structured format that the system can easily process. This could be enhanced by a public recipe database, where users can share and refine recipes collectively. Such a feature would greatly expand the system's scalability and encourage community-driven growth.

Ultimately, the next stage of development should focus on making the system more reliable, flexible and scalable. Advances in AI models and more robust error detection could transform the prototype into a practical tool for everyday use. With these improvements, the assistant could grow from a proof of concept into a trusted system that supports learning and helps users master complex baking recipes.

Hoofdstuk 7

Conclusion

This thesis has explored the design and development of a real-time baking assistant capable of providing immediate, context-aware feedback during the baking process. What began as a technical investigation into how we can easily transfer human skills through AR-based training and multimodal feedback. Soon turned into an investigation of whether large language models (LLMs), in combination with augmented reality, could detect mistakes as they happen and help users correct them. Baking was chosen as the domain because it demands precision and leaves little room for error, which makes it the perfect environment to test such a system.

The results show that it is indeed possible to guide users through a recipe in real time while also identifying mistakes and giving useful feedback. Tests with both fictional and real recipes demonstrated that the assistant could catch errors like incorrect measurements, skipped steps and faulty techniques. Compared to more traditional baking tutorials, this application creates a much more interactive and flexible experience. Instead of only telling the user what to do, the system also explains why, which makes it more reliable as a learning tool.

That being said, the work also revealed some limitations. Performance bottlenecks in Unity required external preprocessing, which added system complexity. The LLM sometimes struggled with interpreting prompt instructions and distinguishing visually similar items, leading to occasional errors or inconsistent guidance. The assistant also cluttered its responses sometimes or proceeded to go to the next instruction before the current one was completed. Altogether, these issues show how hard it is to balance accuracy, performance and usability in a single system. But also illustrates the compromises that we had to make during the development of this thesis

Working on this thesis was not just about building a prototype, but also about learning how to deal with these trade-offs. Each design decision required me to weigh ambition against practicality. For example, while more advanced computer vision models could have improved detection, they would have also required significantly more resources and training data. Instead, I had to make compromises that kept the system functional within the available constraints. This process taught me that innovation is often less about chasing the “perfect” solution and more about creating something that works, even if that means accepting imperfections.

One thing I learned from this project is that creating technologies for human skill transfer is about more than just building an application. It’s about understanding how people actually learn. In this thesis I explored multiple research papers and technologies like LLMs, image capturing, optical character recognition and AR interface. And realized that the goal was to use these tools in a way that genuinely supports learning. Baking is not just a sequence of steps. It involves precision, difficult techniques and small decisions that no system can fully teach. The system should guide the user, highlight mistakes and offer advice when needed. While still letting the user make errors and experiment on their own. The technology should act as a

supportive partner rather than a strict instructor. It should give feedback at the right moments and allow the learner to build habits and confidence over time. Ultimately this project taught me that effective human skill transfer combines technological capability with an understanding of how people actually learn.

Looking forward, there are several clear directions for future work. The combination of LLMs, computer vision and augmented reality offers a powerful foundation for a system that could provide real-time feedback. More advanced LLMs will likely resolve many of the interpretation errors and inconsistencies observed in our system. While action recognition and short video analysis could significantly improve the system's understanding of user actions. Improvements in prompt engineering and system architecture would also help reduce complexity and latency, making the assistant more usable in practice. Finally, expanding recipe flexibility and maybe integrating a community-driven recipe database would broaden the system's scalability.

In conclusion, this research suggest that the prototype developed here is an early step toward an actually AR-based application for human skill transfer. By building on advances in AI models and refinements to the application's design, future iterations could move beyond a proof-of-concept and become a practical tool for real-world use. In the end, this thesis shows that the possibility of a real-time AR baking assistant is more than just an idea. It demonstrated that AI technologies can actually be used in supporting human skill development by detecting mistakes and responding to them in real time. While the system is still a prototype with limitations, it points toward a future where learning complex skills like baking can be made less intimidating and more intuitive to learn.

Bibliografie

[1] Gaoping Huang, Xun Qian, Tianyi Wang, Fagun Patel, Maitreya Sreeram, Yuanzhi Cao, Karthik Ramani, and Alexander J. Quinn. Adaptutar: An adaptive tutoring system for machine tasks in augmented reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, page 1–15. ACM, May 2021.

[2] Dishita G Turakhia, Peiling Jiang, and Stefanie Mueller. The reflective make-ar in-action: Using augmented reality for reflection-based learning of makerskills. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, page 1–6. ACM, April 2023.

[3] Balasaravanan Thoravi Kumaravel, Fraser Anderson, George Fitzmaurice, Bjoern Hartmann, and Tovi Grossman. Loki: Facilitating remote instruction of physical tasks using bi-directional mixed-reality telepresence. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19, page 161–174. ACM, October 2019.

[4] Xingyao Yu, Katrin Angerbauer, Peter Mohr, Denis Kalkofen, and Michael Sedlmair. Perspective matters: Design implications for motion guidance in mixed reality. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, page 577–587. IEEE, November 2020.

[5] Adrian Iftene, Diana Trandabăț, and Vlad Rădulescu. Eye and voice control for an augmented reality cooking experience. *Procedia Computer Science*, 176:1469–1478, 2020.

[6] Hyoyoung Lim, Xiaolei Huang, Samuel Miller, Joshua Edelmann, Timothy Euken, and Stephen Voida. Smart cook: making cooking easier with multimodal learning. In *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers*, UbiComp '19, page 129–132. ACM, September 2019.

[7] Atsushi Hashimoto, Naoyuki Mori, Takuya Funatomi, Yoko Yamakata, Koh Kakusho, and Michihiko Minoh. Smart kitchen: A user centric cooking support system. 01 2008.

[8] Isaias Majil, Mau-Tsuen Yang, and Sophia Yang. Augmented reality based interactive cooking guide. *Sensors*, 22(21):8290, October 2022.

[9] Jaewook Lee, Andrew D. Tjahjadi, Jiho Kim, Junpu Yu, Minji Park, Jiawen Zhang, Jon E. Froehlich, Yapeng Tian, and Yuhang Zhao. Cookar: Affordance augmentations in wearable ar to support kitchen tool interactions for people with low vision. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, UIST '24, page 1–16. ACM, October 2024.

[10] Tushar Nagarajan and Lorenzo Torresani. Step differences in instructional video. 2024.

[11] Daniele Di Mitri, Jan Schneider, and Hendrik Drachsler. Keep me in the loop: Real-time feedback with multimodal data. *International Journal of Artificial Intelligence in Education*, 32(4):1093–1118, November 2021.

[12] Rainer Winkler, Sebastian Hobert, Antti Salovaara, Matthias Söllner, and Jan Marco Leimeister. Sara, the lecturer: Improving learning in online education with a scaffolding-based conversational agent. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–14. ACM, April 2020.

[13] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. pages 779–788, 06 2016.

[14] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and OTHERS. Scaling egocentric vision: The epic-kitchens dataset. 2018.

[15] Fernando De la Torre, Jessica Hodgins, Javier Montano, and Sergio Valcarcel Macua. Detailed human data acquisition of kitchen activities: the cmu-multimodal activity database (cmu-mmac). 04 2009.

[16] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, and OTHERS. Ego4d: Around the world in 3,000 hours of egocentric video. 2021.

[17] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, and OTHERS. Ego-exo4d: Understanding skilled human activity from first- and third-person perspectives. 2023.

[18] Saelyne Yang, Jo Vermeulen, George Fitzmaurice, and Justin Matejka. Aqua: Automated question-answering in software tutorial videos with visual anchors. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '24, page 1–19. ACM, May 2024.

[19] Fatemeh Sarshartehrani, Elham Mohammadrezaei, Majid Behravan, and Denis Gracanin. *Enhancing E-Learning Experience Through Embodied AI Tutors in Immersive Virtual Environments: A Multifaceted Approach for Personalized Educational Adaptation*, page 272–287. Springer Nature Switzerland, 2024.

[20] Aditi Singh, Saket Kumar, Abul Ehtesham, Tala Talaei Khoei, and Deepshikha Bhati. Large language model-driven immersive agent. November 2024.

[21] Peihua Ma, Shawn Tsai, Yiyang He, Xiaoxue Jia, Dongyang Zhen, Ning Yu, Qin Wang, Jaspreet K.C. Ahuja, and Cheng-I Wei. Large language models in food science: Innovations, applications, and future. *Trends in Food Science & Technology*, 148:104488, June 2024.

[22] Daniel Gorman, Simon Hoermann, Robert W. Lindeman, and Bahareh Shahri. Using virtual reality to enhance food technology education. *International Journal of Technology and Design Education*, 32(3):1659–1677, May 2021.

[23] Michael I. Posner Paul M. Fitts. *Fitts, P. M., & Posner, M. I. (1967). Human Performance. Belmont, CA Brooks/Cole.* Prenlice-Hall of India Private Limited, 1967.

[24] Petronela Savin and Diana Trandabat. Ethnolinguistic audio-visual atlas of the cultural food heritage of bacău county — elements of methodology. *BRAIN. Broad Research in Artificial Intelligence and Neuroscience*, 9(1):125–131, 2018.

[25] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. 2019.

[26] Richard Mayer and Roxana Moreno. A cognitive theory of multimedia learning: Implications for design principles. 91, 01 2005.

[27] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context. 2014.

[28] Dima Aldamen, Davide Moltisanti, Evangelos Kazakos, Hazel Doughty, Jonathan Munro, William Price, Michael Wray, Tobias Perrett, and Jian Ma. Epic-kitchens-100, 2020.

[29] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. 2017.