GHOST: Guided Holographic Overlays for Safe Teleoperation

Stijn De Busser
Universiteit Hasselt
Hasselt, Belgium
stijn.debusser@student.uhasselt.be

Dries Cardinaels
UHasselt - Digital Future Lab
Diepenbeek, Belgium
dries.cardinaels@uhasselt.be

Kris Luyten
UHasselt - Digital Future Lab
Diepenbeek, Belgium
kris.luyten@uhasselt.be

***Abstract*—Voor een efficiënte en veilige mens-robotinteractie in industriële omgevingen is het essentieel dat menselijke operatoren kunnen anticiperen op de acties van een robotarm. Bestaande robotbewegingen geven vooraf echter weinig signalen over hun geplande traject. De gevolgen van een commando, zoals het specifieke pad, de reikwijdte of mogelijke botsingen, worden hierdoor vaak pas zichtbaar wanneer de fysieke beweging al is ingezet, wat leidt tot onveiligheid en inefficiëntie.**

Deze paper beoogt dit gebrek aan voorspellende informatie op te lossen door een AR-gebaseerd feedforward-visualisatiesysteem te ontwikkelen. Met behulp van Unity en de Magic Leap 2 is een prototype gerealiseerd waarin via een virtuele, onzichtbare datakloon van de robotarm kinematische berekeningen worden uitgevoerd. Dit stelt het systeem in staat om drie hoofdvisualisaties over de werkruimte weer te geven: een dynamische trajectvisualisatie met botsingsbewustzijn, ghost-skeletons en spatial reach boundaries. Het ontwikkelde prototype reageert op interactieve manipulaties van de operator, waardoor botsingsrisico's en ruimtelijke limieten proactief in kaart worden gebracht nog vóór de fysieke arm in beweging komt.

I. Introductie

De inzet van robotarmen in industriële omgevingen kent een sterke transformatie en snelle toename, gedreven door de transitie naar intelligente productiesystemen [1]. Binnen deze evolutie is een efficiënte en veilige samenwerking tussen mens en robot essentieel geworden om flexibiliteit op de werkvloer te garanderen [2], [3]. Ondanks de aanzienlijke vooruitgang in autonome motion planning-algoritmes [4]–[6], blijft de menselijke operator wegens zijn flexibele probleemoplossende vermogen een cruciale rol spelen bij het instellen, programmeren en monitoren van deze systemen [7], [8]. Het uiteindelijke succes van deze synergie hangt bijgevolg sterk af van de intuïtieve interactie en de gegarandeerde veiligheid tijdens gezamenlijke industriële taken [3], [7].

Hoewel er in de literatuur reeds geavanceerde concepten worden onderzocht, zijn operatoren in de huidige industriële praktijk nog vaak aangewezen op traditionele 2D-schermen of beperkte statische feedback [9]. Hierdoor blijft het lastig om het volledige bewegingspad van de robotarm, potentiële botsingen en ruimtelijke relaties met de omgeving vooraf te overzien en tijdig op mogelijke risico's te anticiperen.

In eerdere literatuur is reeds onderzocht hoe Augmented Reality (AR) kan bijdragen aan het aanpakken van dit probleem door middel van predictieve grafische overlays. Zo beschrijft Fang et al. [10] een AR-systeem dat robottrajecten simuleert en vooraf valideert om de veiligheid te verbeteren, terwijl Quintero et al. [11] rapporteren dat het projecteren van interactieve, virtuele trajecten het programmeren van complexe armen significant vereenvoudigt. Daarnaast beschrijven Chong et al. [12] dat AR helpt bij het plannen van botsingsvrije paden, een concept dat door Matour en Winkler [13] verder is gespecificeerd naar een HoloLens-omgeving waarin de operator trajecten eerst virtueel kan testen op een gesimuleerd model alvorens de fysieke robot aan te sturen. Tot slot vergelijken Gruenefeld et al. [14] verschillende AR-visualisatievormen voor "motion intent". Hieruit blijkt met name dat een volumetrische driedimensionale weergave kan bijdragen tot een hogere ervaren veiligheid en een beter ruimtelijk bewustzijn bij de operator.

Hoewel deze bestaande systemen waardevolle eerste stappen zetten, blijft een intuïtieve, real-time visualisatie van toekomstige robotbewegingen en potentiële gevaren vaak beperkt. Hierdoor blijft het voor operatoren uitdagend om een accuraat mentaal model te vormen van het geplande traject en de mogelijke botsingsrisico's [15], [16]. Bovendien ontbreekt in veel gevallen een duidelijk visueel inzicht in de ruimtelijke relatie tussen de robot en diens directe omgeving, wat het proactief anticiperen op gevaarlijke situaties bemoeilijkt.

Om deze behoefte in te vullen, presenteert deze paper een visuele feedforward-omgeving ontwikkeld met behulp van AR op de Magic Leap 2. Het doel hiervan is om de operator proactief te ondersteunen tijdens het manipulatie-proces om inzicht te geven in het geplande traject en de mogelijke verplaatsing van de robotarm binnen de 3D-werkruimte. Wanneer de operator een handeling uitvoert ter beweging van een node, dan treden de AR-visualisaties onmiddellijk in werking. Hierbij ligt de focus op de technische realisatie en het grafisch ontwerp van deze interactieve overlays, die ontworpen zijn om toekomstige bewegingen, potentiële botsingen en ruimtelijke limieten van de robotarm in kaart te brengen alvorens de fysieke actie uitgevoerd wordt.

De concrete doelstelling van deze implementatie is het ontwikkelen van effectieve visualisaties die beogen de intuïtiviteit en veiligheid voor de operator vergroten. Dit hoofddoel is opgedeeld in drie specifieke onderdelen die direct inspelen op de onduidelijkheden in de bestaande literatuur: trajectvoorspelling, botsingsbewustzijn en ruimtelijk inzicht.

De voornaamste bijdrage van dit werk is het ontwerp en

de implementatie van een geïntegreerd AR-systeem dat drie vormen van feedforward combineert binnen één omgeving. Concreet gaat het om de volgende visualisatiecomponenten:

- **Een AR-gebaseerde trajectvisualisatie:** Het ontwerpen en implementeren van een visuele feedforward-omgeving die het volledige geplande traject van de robotarm weergeeft aan de operator.
- **Een proactief botsingsdetectiesysteem:** De ontwikkeling van visuele hulpmiddelen en waarschuwingsmechanismen die gericht zijn op het verhogen van het risicobewustzijn van de operator door naderende botsingszones te markeren.
- **Een ruimtelijke contextomgeving (Ruimtelijk inzicht):** De integratie van driedimensionale AR-visualisaties die gericht zijn op het vergroten van het ruimtelijk inzicht en het omgevingsbewustzijn van de operator.

II. Literatuurstudie

Een geïntegreerde feedforward-visualisatie brengt drie aspecten samen: de visualisatie van het traject, de perceptie van botsingsrisico's en het ruimtelijk inzicht van de operator. In de literatuur worden deze aspecten elk afzonderlijk onderzocht, maar een aanpak die de drie combineert ontbreekt, het gat dat dit werk wil dichten.

A. Visualisatie van Robottrajecten in AR

Het visualiseren van toekomstige bewegingen van een robotarm (feedforward) wordt in de literatuur benaderd via twee verschillende vormen van operator-ondersteuning: pre-operationele simulatie en realtime procesbegeleiding. Binnen de pre-operationele benadering biedt het RPAR-II systeem feedforward door trajecten vooraf grafisch te simuleren [10]. Dit stelt operatoren in staat om dynamische aspecten zoals overshoot en snelheid vooraf te evalueren. Een vergelijkbare methode zet AR in om botsingsvrije paden te genereren op basis van demonstraties door de operator [12]. Daarnaast stelt het projecteren van een virtueel robotmodel via een head-mounted display (HMD) operators in staat het volledige geplande traject eerst virtueel te testen en te verifiëren voordat de fysieke robot daadwerkelijk in beweging komt [13].

Tegenover deze statische validatiemethoden staat het onderzoek naar realtime feedforward tijdens de actieve beweging van de robotarm. Hiertoe worden visualisatievormen onderzocht zoals "Path" (het geprojecteerde pad van de end-effector) en "Preview" (een 3D-animatie van de toekomstige robotconfiguratie) [14].

Hoewel deze onderzoeken de fundamentele voordelen van trajectvoorspelling suggereren, identificeren zij tegelijkertijd enkele kritieke beperkingen. Binnen de pre-operationele systemen blijft de concrete grafische representatie van het pad, zoals de exacte vormgeving van de trajectlijnen, vaak onderbelicht of abstract. Aan de andere kant beperkt bestaand onderzoek naar realtime visualisaties zich hoofdzakelijk tot de nabije toekomst, waardoor het grotere overzicht van de volledige planning buiten beschouwing blijft. Dit duidt op de noodzaak voor een geïntegreerde oplossing die zowel het globale traject als de realtime status dekkend visualiseert.

B. Perceptie van Botsingsrisico's

Voor een operator binnen een gedeelde werkomgeving is het cruciaal om potentiële risico's proactief in te kunnen schatten. In de literatuur worden hiervoor zowel visuele als algoritmische oplossingen aangedragen om botsingsrisico's inzichtelijk te maken of te beheersen. Aan de visuele kant wordt het concept van 'collision-free volumes' (CFV) toegepast, waarbij operatoren vooraf veilige zones definiëren om botsingen te vermijden [12]. Dit principe is tijdens de operationele fase onderzocht met een zogenaamde "Volume"-visualisatie, waarbij de fysiek ingenomen ruimte van de robot wordt weergegeven als een vereenvoudigd driedimensionaal volume rondom de arm [14]. Hoewel deze weergave abstract is, rapporteert de bijbehorende gebruikersstudie dat deze methode leidde tot de hoogste ervaren veiligheid bij de operatoren.

Aan de algoritmische zijde focust dit onderzoek zich op het mathematisch inkapselen van de robotgeometrie. Hiertoe bestaat er een collision avoidance-systeem dat realtime geometrische capsules rondom zowel de mens als de robot berekent om het traject live aan te passen [17]. Om deze wiskundige buffers beter vertaalbaar te maken naar de operator, maakt een andere benadering gebruik van een expliciete representatie van driedimensionale 'danger zones' [18]. Deze gevarenzones representeren de fysieke volumes rondom de robotsegmenten waarin veiligheidsnormen worden overschreden op basis van de actuele snelheid en remweg. Dit sluit aan bij recente ontwikkelingen waarin dynamische bounding-volumes online worden geschaald om de afstand tussen operator en machine continu te bewaken en de procesvloeiendheid te ondersteunen [19].

Hoewel deze algoritmische benaderingen er veiligheid technisch op gericht zijn de veiligheid te waarborgen, blijven ze vaak volledig reactief en ontbreekt in veel gevallen een intuïtieve visuele interface; de machine handelt intern zonder directe communicatie naar de mens. Een vergelijkbaar probleem doet zich voor bij systemen waarbij weliswaar een 'AR-based warning system' via projectoren of head-mounted displays wordt ingezet [20], maar waarbij niet wordt gespecifieerd welke informatie vooraf getoond wordt. Het blijft in dergelijke benaderingen onduidelijk of toekomstige gevaren proactief worden gevisualiseerd, of dat het systeem pas waarschuwt wanneer een kritieke zone reeds is betreden. Concrete visuele feedbackelementen, zoals kleurgecodeerde proximity-zones of graduele waarschuwingen, blijven hierdoor in de bestaande literatuur onderbelicht.

C. Ruimtelijk inzicht van de operator

Het effectief samenwerken met een robotarm vereist dat de operator een accuraat mentaal model vormt van de gedeelde driedimensionale werkruimte. AR-systemen proberen dit ruimtelijk inzicht te faciliteren door digitale contextinformatie rechtstreeks over de fysieke realiteit te projecteren. Vroege systemen zoals RPAR-II [10] suggereren dat pre-operationele

AR-simulaties operatoren helpen bij het ruimtelijk evalueren van procesparameters. Met de opkomst van geavanceerde head-mounted AR-displays is deze focus verschoven naar realtime ondersteuning op de werkvloer. Dit wordt ondersteund door bevindingen die erop wijzen dat het direct projecteren van de ruimtelijke intentie van de robot via een HMD bijdraagt tot het tijdig en accuraat anticiperen op machinebewegingen [16].

Recenter onderzoek geeft verdere inzichten in de voordelen van deze benadering voor het ruimtelijk model. Zo is er gerapporteerd dat gerichte AR-cues de efficiëntie bij complexe ruimtelijke planning kunnen verhogen en fysieke interferentie tussen mens en machine helpen minimaliseren [21]. Daarnaast wijst recent werk [15] op een positief effect van ruimtelijke veiligheidszones op het situationeel bewustzijn van de operator, wat in lijn ligt met de praktijkbevindingen waarin een subjectieve meerwaarde voor de algehele mens-robotinteractie wordt gerapporteerd [20].

De kritieke tekortkoming in de huidige literatuur is dat geen artikel alle drie de kerncomponenten: trajectvoorspelling, botsingsbewustzijn en ruimtelijk inzicht combineert in één geïntegreerde visualisatiemethode. De meeste onderzoeken focussen op één of twee aspecten, waarbij trajectvoorspelling visueel gezien het minst is uitgewerkt. Er is bijgevolg nood aan een allesomvattende, real-time 3D-feedforward op een head-mounted display (zoals de Magic Leap 2) om de operator te helpen ondersteunen.

III. Systeem Architectuur

Een feedforwardvisualisatie vereist dat virtuele trajecten, gebruikersinteractie en fysieke robotbewegingen consistent op elkaar afgestemd zijn. De systeemarchitectuur vormt hiervoor de technische basis door AR-weergave, robotaansturing en ruimtelijke uitlijning te combineren.

Systeemconfiguratie en technische opzet. De technische architectuur van het feedforward-systeem is opgebouwd rondom een fysieke robotarm-opstelling gekoppeld aan een bijbehorende digitale scène binnen de Unity-engine (versie 2022.3.62f3) als centraal softwareplatform. De logica van dit prototype is geïmplementeerd in C# als primaire programmeertaal, waarbij voor de AR-weergave gebruikgemaakt wordt van het Magic Leap 2 head-mounted AR-platform. Binnen deze opzet is de functionele logica opgedeeld in drie complementaire hoofdcomponenten, zoals schematisch en chronologisch weergegeven in de interactieloop van Figuur 1. Dit systeem bestaat uit een *interactie-component* voor het registreren van de operatorinput, een *trajectcontroller* voor het databeheer van de ruimtelijke doelpunten, en een *visualisatiemodule* die frame-per-frame de synchronisatie en AR-projectie aanstuurt. De koppeling tussen deze virtuele AR-omgeving en de fysieke werkruimte wordt gewaarborgd door een *fiducial marker system* op basis van de AprilTag 36h11-familie, wat een snelle bepaling van de camerapositie ten opzichte van de robotarm garandeert.

Op hoofdlijnen werken de invoer, robotdata en visualisaties samen via een gestructureerde datastroom. Het proces start zodra de operator via een *pinch*-gebaar een doellocatie vastlegt,

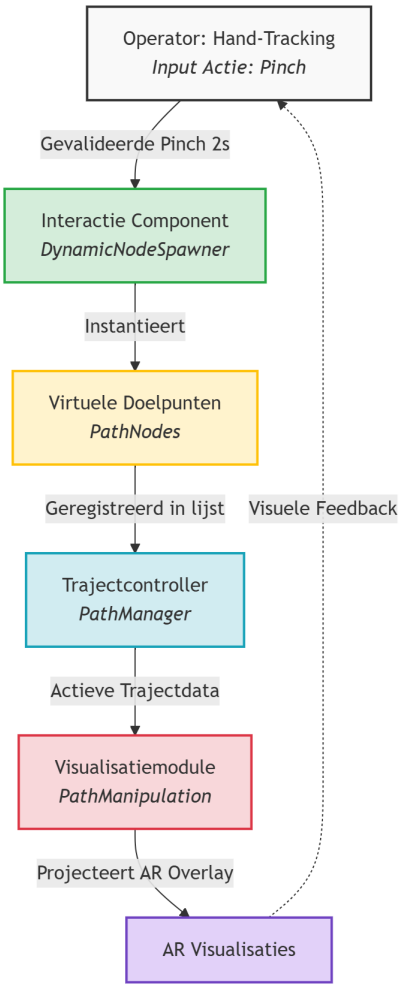


Fig. 1: Architectuur van de interactie- en visualisatieloop.

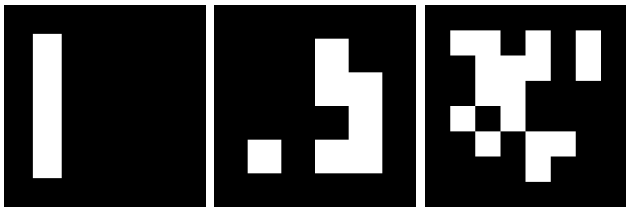
wat essentieel is voor het waarborgen van een handsfree workflow binnen een industriële setting. Om de actuele status van de fysieke robotarm te betrekken, haalt het systeem de huidige gewrichtshoeken op. Een cruciale ontwerpkeuze binnen deze architectuur is het gebruik van een onzichtbare, virtuele datakloon van de robotarm. Telkens wanneer een nieuwe node wordt geregistreerd, voert deze kloon direct via Inverse Kinematics (IK) en Forward Kinematics de simulaties en geometrische interpolaties uit in slechts één frame. Hierdoor stroomt de ruimtelijke data (3D-coördinaten en berekende ashoeken) door het systeem naar de visualisatiecomponenten, nog voordat de fysieke controller de asynchrone beweging in gang zet.

Hoewel de fysieke robotarm over zes vrijheidsgraden (6DOF) beschikt, is de architectuur bewust begrensd tot de eerste drie vrijheidsgraden (3DOF) voor de positionele IK. Tot slot vindt er een algoritmische veiligheidscontrole plaats waarna het traject een bereikbaarheids- en botsingscontrole doorloopt voordat de visualisatie wordt opgebouwd. Dit geheel maakt de uiteindelijke feedforward-visualisaties in AR mogelijk aangezien de gegenereerde trajectdata wordt

geprojecteerd als een richtinggevende curve die in real-time meebuigt bij handmatige manipulatie en bij een gedetecteerde botsing onmiddellijk transformeert tot een dikke, rode waarschuwingslijn.

Ruimtelijke uitlijning. Om de virtuele AR-scène correct te projecteren, wordt gebruikgemaakt van een *fiducial marker system*. Hoewel deze markers visueel gelijkenissen vertonen met QR-codes, zijn ze specifiek geoptimaliseerd voor Computer Vision. In tegenstelling tot QR-codes, die primair gericht zijn op data-opslag, zijn fiducial markers ontworpen voor een snelle en nauwkeurige bepaling van de 3D-camerapositie en -oriëntatie ten opzichte van het fysieke object. Binnen de beschikbare trackingsystemen is voor deze applicatie gekozen voor de AprilTag-technologie boven alternatieven zoals ArUco. Ter illustratie toont Figuur 2 het visuele verschil tussen een standaard ArUco-marker en twee verschillende AprilTag-configuraties.

De uiteindelijke keuze voor de AprilTag 36h11-familie is gebaseerd op de hoge mate van robuustheid tegen beeldruis en foutieve detecties in industriële omgevingen [22]. Dit specifieke formaat bestaat uit een 6×6 grid van 36 bits, waarbij de minimale Hamming-afstand gegarandeerd 11 bits bedraagt (zie Figuur 2c). Dit houdt in dat het herkenningsalgoritme minstens 11 bits foutief moet interpreteren voordat het systeem de tag per abuis verwart met een ander ID uit de database. Dit minimaliseert de kans op foutieve detecties significant in vergelijking met families met een lagere Hamming-afstand, zoals de 16h5-variant (Figuur 2b).



(a) ArUco (b) Tag16h5 (c) Tag36h11

Fig. 2: Visuele vergelijking van fiducial markers voor ruimtelijke uitlijning: (a) ArUco, (b) AprilTag 16h5 en (c) AprilTag 36h11.

IV. Systeem Implementatie

De implementatie van de AR-applicatie steunt op verschillende ondersteunende componenten die gebruikersinteractie, nodebeheer en robotaansturing met elkaar verbinden. Deze componenten vormen de technische basis waarop de feedforwardvisualisaties worden opgebouwd.

A. Dataverwerking en herberekeningslogica

De functionele verbindingen tussen de verschillende componenten en de bijbehorende datastroom zijn reeds vastgelegd in de systeemarchitectuur (zie Figuur 1). Tijdens de runtime monitort de visualisatiemodule continu en frame-per-frame de actuele status van de trajectcontroller om te analyseren of de operator de ruimtelijke informatie van een node heeft gemanipuleerd. Pas wanneer er een relevante verandering in de data optreedt, triggert het systeem een herberekening. Een dergelijke verandering vindt plaats in drie specifieke scenario's:

1) **Activatie van een nieuwe node:** De interne index verschuift, waardoor het systeem de datastroom moet omleiden naar het opeenvolgende doelpunt in de reeks.
2) **Fysieke verplaatsing van de node:** Zodra de ruimtelijke coördinaten (x, y, z) van de actuele doellocatie wijzigen — bijvoorbeeld wanneer de operator de virtuele node handmatig versleept — volgt onmiddellijk een trigger om de geometrische transformaties te herberekenen.
3) **Modificatie van de benaderingshoek:** Indien de node zich op dezelfde geografische locatie bevindt, maar de invalshoek ten opzichte van de robotarm (*angle of attack*) wijzigt, dient de configuratie van de robotarm hierop te worden aangepast.

Hoewel de onderliggende logica ontworpen is om alle drie de condities te verwerken, treden in de huidige systeemconfiguratie primair de eerste twee scenario's op. Om de procescomplexiteit te beheersen, wordt de parameter voor de invalshoek bij de generatie van een nieuwe node standaard geïnitialiseerd op een statische waarde van `0.0f`, waardoor deze momenteel niet handmatig door de operator wordt gecorrigeerd.

Wanneer aan een van de voorwaarden voor een herberekening is voldaan, bepaalt het systeem via inverse kinematica de benodigde gewrichtshoeken om het nieuwe doelpunt te bereiken. Om deze toekomstige beweging veilig te kunnen voorspellen, genereert het systeem op dat moment een onzichtbare kinematische kloon van de robotarm. Deze kloon fungeert als een geïsoleerde momentopname van de actuele startpositie.

Vervolgens worden de berekende doelhoeken iteratief toegepast op deze virtuele kopie, waardoor de beweging in de achtergrond wordt gesimuleerd. De coördinaten die de kloon tijdens deze simulatie doorloopt, worden vastgelegd als een trajectreeks. Omdat deze processtappen volledig onafhankelijk en geïsoleerd van de daadwerkelijke robotarm plaatsvinden, fungeert deze resulterende data uitsluitend als de fundering voor de realtime feedforward-visualisaties.

B. Gebruikersinteractie en Node Spawning

De primaire datastroom binnen de applicatie start bij de fysieke handelingen van de operator. Binnen de softwarearchitectuur is het `DynamicNodeSpawner`-component specifiek ingericht om deze ruimtelijke interacties storingsvrij af te vangen, te valideren en om te zetten in bruikbare trajectdata.

Pinch-gesture. De initiële invoer voor het positioneren van een trajectnode verloopt via hand-tracking, ondersteund door het Unity Input System en de XR Interaction Toolkit (XRI). De fysieke interactie is gekoppeld aan een selectie-actie, wat zich op het Magic Leap 2-platform vertaalt naar een intuïtieve "pinch"-beweging met de linkerhand van de operator.

Bij het ontwerpen van deze ruimtelijke interactie vormde *tracking jitter* het kortstondig incorrect interpreteren van de

handpositie door de camera's, wat een risico vormt op redundante data-instantiatie. Om te voorkomen dat minieme haperingen of onbewuste handbewegingen resulteren in een ongewenste overproductie van nodes, is een tijdsgebonden validatiefilter geïmplementeerd. De operator dient de pinch-beweging onafgebroken vast te houden, waarbij een interne timer vereist dat deze actie minimaal 2,0 seconden aanhoudt alvorens de 3D-coördinaat definitief wordt geregistreerd. Ter verdere foutpreventie is er na een succesvolle registratie een tijdelijke cooldown actief om accidentele dubbele invoer uit te sluiten.

UI Feedback. Vanwege de ingebouwde validatietijd van twee seconden is continue visuele feedback essentieel om de actuele systeemstatus naar de operator te communiceren. Bij de initialisatie van het systeem wordt er een grafische interface gegenereerd die direct aan de hoofdcamera gekoppeld blijft. Hierdoor is gewaarborgd dat de display-elementen op een vaste afstand binnen het gezichtsveld zweven en altijd zichtbaar blijven, onafhankelijk van de kijkrichting van de operator in de fysieke ruimte.

De opbouw van deze interface is gecentreerd rond twee datagestuurde weergavecomponenten:

- **Statusmelding:** Een tekstcomponent dat de actuele status van het invoerproces reflecteert, met een dynamische transitie van *"Hold pinch to spawn node"* naar *"Spawning Node..."*.
- **Voortgangsbalk:** Een visuele indicator die lineair opschaalt van 0 tot 100%. Deze schaling is direct gekoppeld aan de ratio tussen de verstreken tijd en de vereiste tijdsdrempel, wat de operator een vloeiende animatie biedt tijdens het vasthouden van de pinch-beweging.

Ten slotte is binnen de runtime-logica vastgelegd dat de generatie van nodes pas operationeel beschikbaar is zodra de initiële ruimtelijke scène volledig in het geheugen is ingeladen.

V. Visualisaties Implementatie

De geïmplementeerde visualisaties zijn direct afgeleid van de probleemstelling en de daaruit voortvloeiende onderzoeksvragen. Voor elke onderzoeksvraag is een specifiek visualisatiecomponent ontwikkeld dat gericht is op het ondersteunen van de interactie met de robotarm. Om deze visualisaties aan te sturen zonder de fysieke robot te gebruiken, is er een onderliggend simulatiesysteem geïmplementeerd.

TABLE I: Overzicht van de geïmplementeerde visualisaties

Visualisatie	Visuele weergave
Trajectlijn	Vloeiende, smaller wordende curve
Botsingswaarschuwing	Traject wordt rood en 3× dikker
Ghost Skeletons	Grijze skeletten, oplopende transparantie
Proximity Shield	Gebogen vlak, oranje → rood
Error Tether	Rode lijn van grenspunt naar node

A. De virtuele gekloonde arm

Aan de basis van het systeem is ervoor gekozen om een virtuele datakloon van de gewrichten te integreren. Telkens wanneer een nieuw doel wordt geregistreerd, activeert de applicatie de trajectgeneratie. Binnen dit proces wordt een onzichtbare rekenkloon van de arm gesimuleerd die exact over de geometrie van de fysieke arm wordt geplaatst. Dit subsysteem dient als onzichtbaar testmodel voor alle wiskundige berekeningen. Hierdoor kunnen de visualisaties volledig onafhankelijk vooruitberekend worden in slechts één frame, terwijl de echte arm in asynchrone tijd zijn fysieke beweging nog moet uitvoeren. De datalijst van eerder berekende trajectpunten (`trajectoryPoints`) wordt hierbij gewist en klaargezet voor de nieuwe simulatiecyclus.

B. Bepalen van de doelhoeken en veiligheidsmarges

Voor trajectvoorspelling haalt de applicatie eerst de actuele gewrichtshoeken (`currentAngles`) van de robotarm op. Vervolgens berekent de kinematische transformatielaag de finale doelconfiguratie aan de hand van de `targetNode` (de 3D-coördinaat van het einddoel). Omdat de Inverse Kinematics (IK)-berekeningen in deze toepassing uitsluitend de hoeken tot aan het polsgewricht bepalen, moet de doelcoördinaat eerst ruimtelijk gecompenseerd worden. Zonder deze compensatie zou de robot proberen het doelwit aan te raken met de pols in plaats van met de werkelijke gereedschapspunt (*end-effector*). Door de lengte van het gereedschap (`L4`) af te trekken van de doelcoördinaat, wordt exact berekend op welke 3D-positie de pols moet eindigen (de *`wristTarget`*).

Vooraleer deze doellocatie definitief wordt vrijgegeven, vindt er een algoritmische veiligheidscontrole plaats. De fysieke armsegmenten (basis `L1`, bovenarm `L2`, onderarm `L3`) worden geometrisch geëvalueerd. Ligt de `wristTarget` verder dan de maximale reikwijdte van de arm (`L2 + L3`), dan wordt deze coördinaat intern begrensd (*geclampt*) op 99% van het maximum. Dit voorkomt singulariteiten en wiskundige berekeningsfouten door fysieke overstrekking. De calculator module rekent vervolgens de pure 3-DOF doelhoeken uit, waarna deze teruggemapt worden naar een 6-DOF array (`targetAngles`) door de niet-gebruikte oriëntatie-assen vast te zetten op nul.

C. Trajectsimulatie en interpolatie

Met de startpositie (`currentAngles`) en de berekende eindpositie (`targetAngles`) vastgesteld, genereert het simulatiecomponent het voorspelde pad. Hierbij wordt de benodigde beweging niet geleidelijk over tijd verdeeld, maar direct iteratief vooruitberekend en opgeknipt in een vaste resolutie van 150 stappen. De keuze van deze resolutie is proefondervindelijk vastgesteld aangezien het een vloeiende, ononderbroken lijn biedt.

Tijdens elke stap worden de hoeken geïnterpoleerd en verschoven van start naar doel op basis van gewrichtssnelheden. De virtuele kloonarm neemt bij elke stap deze opeenvolgende tussenhoeken aan, waarna via Forward Kinematics de wereldcoördinaten van de tooltip op dat specifieke moment worden opgeslagen. *Forward kinematics fungeert hierbij als het omgekeerde van IK, aangezien het hoeken verwerkt en een resulterende positie in de ruimte teruggeeft.* Deze punten

vormen samen het volledige verloop van de beweging, wat de datafunderingslaag vormt voor de rendering van de trajectlijn.

Om de fysieke beweging van de robotarm getrouw na te bootsen, moeten alle gewrichten gelijktijdig hun rotatie voltooien. In de simulatie wordt dit gerealiseerd door de gewrichtssnelheden proportioneel aan elkaar te berekenen. Wanneer bijvoorbeeld één gewricht een rotatie van 30 graden moet maken en een ander slechts 10 graden, wordt de snelheid van het tweede gewricht verlaagd naar één derde van de snelheid van het eerste. Het gewricht met de grootste af te leggen afstand krijgt hierbij de maximale snelheid toegewezen. Deze aanpak is essentieel voor een realistische visualisatie die synchroon loopt met de werkelijkheid.

Hoewel de fysieke robotcontroller gebruikmaakt van specifieke versnellings- en vertragingsprofielen, is ervoor gekozen om deze niet op te nemen in de simulatie. Aangezien de geometrische vorm van het traject bij gewrichtsinterpolatie onveranderd blijft ongeacht de versnelling, volstaat een lineaire berekening voor een accurate visualisatie van het ruimtelijke pad. Nu de trajectdata is gegenereerd en opgeslagen, kan deze data direct worden aangewend door de visualisatiecomponenten.

D. Trajectvisualisatie en Dynamische Rendering

Technische Realisatie en Rendering-Parameters. Nadat de lijst met trajectpunten is gegenereerd, wordt deze vertaald naar een visuele interface met behulp van een `LineRenderer`-component. Dit component is bij uitstek geschikt voor dit doeleinde, omdat het in staat is om door een reeks driedimensionale coördinaten een continue lijn te trekken die eenvoudig kan worden aangepast qua kleur, dikte en vorm.

De technische realisatie van dit pad gebeurt door voor elk berekend punt in de lijst een hoekpunt te definiëren. Het component spant vervolgens automatisch een segment tussen deze opeenvolgende punten. Omdat deze punten zeer dicht op elkaar staan, transformeert de aaneenschakeling van korte, rechte segmenten visueel tot een vloeiende curve die de beweging van de robot accuraat volgt.

Aangezien de beweging van een robotarm zelden in een rechte lijn verloopt, volgt de visualisatie de natuurlijke curvatuur van de gewrichtsbewegingen. Om te voorkomen dat dit pad er hoekig of gefragmenteerd uitziet, is het component geconfigureerd met extra vertices voor een vloeiende weergave:

- **`numCornerVertices`:** Deze parameter bepaalt de ronding bij de knikpunten in de lijn, waardoor de overgang tussen de 150 segmenten vloeiend oogt.
- **`numCapVertices`:** Deze component zorgt voor een halfronde afwerking van het begin- en eindpunt van de lijn, wat een zachtere visuele integratie in de AR-omgeving geeft.

De visuele presentatie is verder verfijnd door de breedte van de lijn variabel te maken; de lijn wordt gaandeweg slanker naarmate deze het doel nadert. Dit creëert een directioneel beeld dat de aandacht van de operator intuïtief naar het eindpunt leidt.

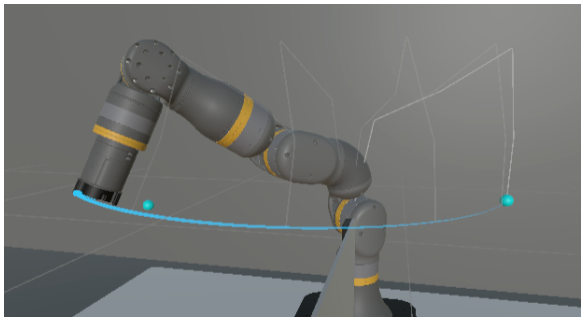


Fig. 3: Realtime feedforward-visualisatie van het geprojecteerde robottraject via een vloeiende `LineRenderer`-curve.

Dynamische Voortgangsindicatie. Naast deze esthetische eigenschappen is de rendering dynamisch: het systeem monitort continu welk punt op het traject zich het dichtst bij de fysieke robot bevindt. Punten die de robot reeds gepasseerd is, worden direct uit de visualisatie verwijderd. Hierdoor lijkt de robot de lijn als het ware te ”consumeren”, wat directe visuele feedback geeft over de voortgang van de beweging. Dit wordt gerealiseerd door de actuele positie van de end-effector te spiegelen aan het eerstvolgende aanstaande punt in de datareeks, waarna het gepasseerde trajectdeel direct wordt afgesneden.

Interactiviteit en Responsiviteit. Een cruciaal aspect van de visualisatie is de interactieve feedback bij manipulatie. Het systeem controleert binnen een continue runtime-cyclus of de positie van een `PathNode` door de operator is gewijzigd. Indien er een verandering wordt gedetecteerd, wordt de volledige trajectsimulatie onmiddellijk opnieuw uitgevoerd. Hierop reageert de visuele lijn door direct mee te buigen met de handbewegingen van de operator. De hoge verversingssnelheid van deze simulatie-loop garandeert een realtime update van de `LineRenderer`. Hierdoor ontstaat een responsieve gebruikerservaring waarbij de visuele voorspelling synchroon loopt met de interactieve manipulatie van de `PathNodes`. Het resultaat van deze realtime trajectvoorspelling is te zien in Figuur 3.

De visualisatie maakt gebruik van één enkele renderercomponent die de gehele lijst van trajectpunten in één keer verwerkt. Hoewel het resulterende traject oogt als een vloeiende curve, is er in de implementatie geen gebruikgemaakt van complexe curve-wiskunde (zoals splines of Bézier-curves). De visualisatie is puur gebaseerd op een aaneenschakeling van 150 lineaire segmenten tussen de berekende datapunten. Door de hoge dichtheid van deze punten worden de scherpe hoeken tussen de segmenten onzichtbaar voor het oog, wat resulteert in een visueel vloeiend pad dat de complexe beweging van de robotarm accuraat benadert. Het toont hiermee effectief aan hoe het systeem een traject kan weergeven vooraleer de fysieke actie is gestart (feedforward).

Botsingdetectie. Net zoals de trajectpunten vooraf in hun geheel worden berekend, voert de *veiligheidsmodule*

(`CheckTrajectoryCollisions`) een volledige scan uit op potentiële botsingen voordat de beweging start. Voor deze detectie is gebruikgemaakt van een `Physics.SphereCastAll`-behandeling. In tegenstelling tot een standaard laserstraal (*Raycast*), projecteert deze methode een virtuele volumetrische bol over elk segment van het geplande traject.

De keuze voor deze volumetrische scan is fundamenteel voor de nauwkeurigheid: een bolvormig volume representeert de fysieke omvang en dikte van de robotarm vele malen beter dan een eendimensionale lijn. Hierdoor worden ook objecten gedetecteerd die zich vlak naast het ideale pad bevinden, maar die door de fysieke breedte van de robotarm alsnog geraakt zouden worden. De straal van deze bol (`detectionRadius`) is instelbaar en wordt nauwkeurig afgestemd op de werkelijke afmetingen van de robotcomponenten.

Hoewel het subsysteem de mogelijkheid biedt om de exacte coördinaten en geraakte objecten op te slaan in een interne datalijst voor verdere analyse, is er in de gebruikersinterface voor gekozen om de feedback puur visueel en intuïtief te houden. Het algoritme bevat bovendien specifieke geometrische filters (`IsNodeCollider` en `IsRobotCollider`) om foutpositieven te voorkomen. Hierdoor negeert het systeem botsingen met de eigen robotonderdelen of met de vooraf gedefinieerde `PathNodes`.

Zodra de simulatie een intersectie met een extern object detecteert, verandert de kleur van het traject direct naar rood en verdrievoudigt de lijndikte.

Tijdens de ontwikkelingsfase is geëxperimenteerd met het genereren van een extra visuele marker, zoals een rode bol, op de exacte locatie van de gedetecteerde botsing. Hoewel dit technisch nauwkeurige informatie verschafte, leek de meerwaarde voor de operator in de praktijk beperkt. Dit is echter niet formeel geëvalueerd en zou nader onderzocht kunnen worden. Sterker nog, extra 3D-objecten in de AR-omgeving zorgden voor visuele ruis en konden het zicht op het werkelijke obstakel belemmeren. Er is daarom besloten om te vertrouwen op de kleurverandering en verdikking van het volledige traject. Deze visuele waarschuwing bij een naderende botsing is weergegeven in Figuur 4.

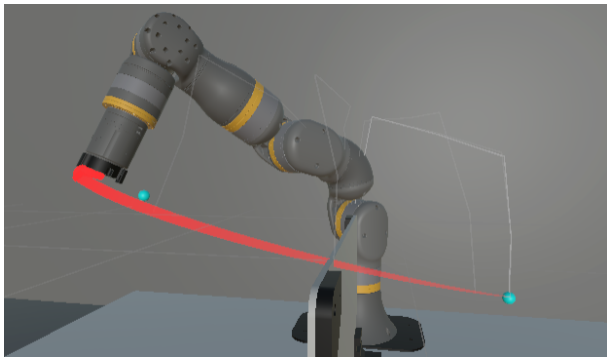


Fig. 4: Visuele feedback bij botsingsdetectie: het traject kleurt onmiddellijk rood en verdrievoudigt in dikte zodra een intersectie met een extern object wordt berekend.

E. Ghost Skeletons: Predictieve Pose-visualisatie

De ”Ghost Skeletons” (of predictieve schaduwmodellen) visualiseren de volledige configuratie van de robotarm op tien specifieke intervallen langs het berekende traject. Waar de trajectlijn enkel de verplaatsing van de *end-effector* toont, geven deze skeletten de operator inzicht in de toekomstige houding van alle zes gewrichten.

Technisch wordt dit gerealiseerd door tijdens de simulatie van 150 stappen op vaste intervallen pose-snapshots te maken van de gewrichtsposities. Voor elke snapshot wordt een datastructuur gevuld met de 3D-coördinaten van de zes rotatieassen en de tooltip. Bij een standaardresolutie van 150 stappen genereert het algoritme elke 15 stappen een pose-snapshot. Dit resulteert in tien individuele skeletmodellen die proportioneel over het traject verdeeld zijn.

De weergave van de skeletten is direct afhankelijk van de totale simulatieduur. Indien een traject extreem kort is, bijvoorbeeld wanneer de doel-node zich zeer dicht bij de huidige positie van de robot bevindt, blijft de logica van de stapgrootte (met een interval van 15) behouden. Indien het totale traject bijvoorbeeld slechts 45 stappen beslaat, worden er bijgevolg slechts drie ghosts getoond. Hierdoor blijft de hoeveelheid visuele informatie op een klein oppervlak beperkt.

Hoewel de aansturing van de robotarm gebaseerd is op een 3DOF + 1DOF-model (waarbij hoofdzakelijk de eerste drie gewrichten en de invalshoek van de pols worden berekend), bevat elk ghost-skelet desondanks zes knikpunten. Dit is mogelijk doordat de simulatie gebruikmaakt van een volledige hiërarchische kopie van de robotarm via *Forward Kinematics*. Zelfs gewrichten die tijdens de beweging niet actief van hoek veranderen (zoals de rotatie van de bovenarm), hebben een vaste fysieke positie in de ruimte. Door deze volledige kinematische keten te renderen, krijgt de operator een waarheidsgetrouw beeld van de volledige omvang van de robotarm, wat cruciaal is voor het inschatten van de veiligheid en de benodigde werkruimte.

Visuele Opmaak en Cognitieve Ergonomie. De Ghost Skeletons zijn functioneel geïmplementeerd als onafhankelijke rendercomponenten met een hoekig profiel. Momenteel zijn er maximaal 10 instanties hiervan mogelijk. Deze hoeveelheid bleek proefondervindelijk voldoende om de vloeiendheid van de beweging te begrijpen zonder dat er sprake is van *cognitive overload* of visuele obstructie van de omgeving.

De skeletten hebben een neutrale grijze kleur (RGB: 0.65, 0.65, 0.65), wat een duidelijk onderscheid maakt met de actieve blauwe of rode trajectlijn. Hierbij is een dynamische transparantiegradiënt (*Alpha Blending*) toegepast:

- De ghosts nabij de huidige positie van de robot hebben een lage opacity (ca. 5%).
- De ghosts nabij het einddoel hebben een hogere opacity (tot 50%).

Dit verloop creëert een diepte-effect en legt de nadruk op

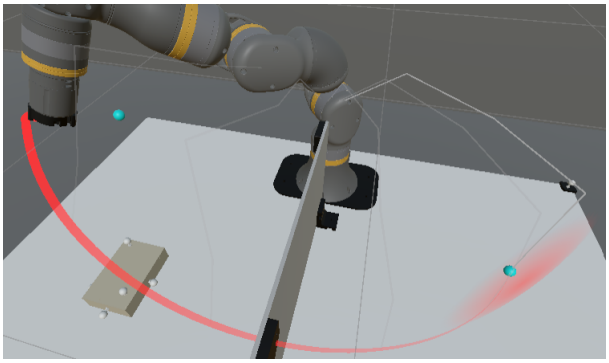


Fig. 5: De 'Ghost Skeletons' tonen de toekomstige configuratie van de robotgewrichten op opeenvolgende intervallen langs het pad.

de uiteindelijke configuratie van de robot. Net zoals bij de trajectlijn is er sprake van een dynamisch consumptie-effect: zodra de fysieke robotarm een ghost passeert, wordt het betreffende weergavecomponent uitgeschakeld. Dit geeft de operator intuïtieve feedback over welke delen van de geplande beweging reeds succesvol zijn voltooid. Deze predictieve pose-visualisatie is geïllustreerd in Figuur 5.

Systeemprestaties en Object Pooling. Aangezien het per frame berekenen en renderen van tien afzonderlijke skelet-modellen rekenintensief kan zijn, is er een optimalisatieslag doorgevoerd middels een **Object Pool** [23]. In plaats van bij elke frame-update nieuwe weergavecomponenten in het geheugen te creëren en te vernietigen, houdt het systeem een vaste lijst van tien objecten paraat. Wanneer de operator een node versleept, worden deze bestaande architectuurcomponenten simpelweg naar de nieuwe coördinaten verplaatst en hun ruimtelijke eigenschappen bijgewerkt, wat een aanzienlijke prestatiewinst oplevert.

F. Spatial Reach Boundaries

Oorspronkelijk was het idee om het ruimtelijk bewustzijn en het bereik van de robotarm te visualiseren door middel van een volledige, semi-transparante sphere met zichtbare omtreklijnen (wireframes). De gebruiker kon dan, in combinatie met tekstlabels ("Out of Reach"), zien of een node zich al dan niet binnen het bereik bevond.

Dit concept is echter geschrapt vanwege de visuele drukte (*clutter*) die hiermee werd gecreëerd. AR werkt het best met minimalistische en doelgerichte visuele feedback. Daarom is er gekozen voor een dynamische, reactieve visualisatie die uitsluitend zichtbaar wordt wanneer de fysieke limieten daadwerkelijk bereikt dreigen te worden. Dit vernieuwde bereikscomponent is opgesplitst in twee samenwerkende visualisaties: een dynamisch waarschuwingsschild (*Proximity Shield*) en een *error tether*.

Dynamic Proximity Shield. Het waarschuwingsschild functioneert als een dynamisch, oplichtend oppervlak dat exact op de grens van de maximale reikwijdte van de robotarm ligt. Deze fysieke maximale grens wordt wiskundig gedefinieerd als $L2 + L3$, oftewel de opgetelde lengte van de twee voornaamste robotarmdelen (de bovenarm en onderarm). Om dit schild wiskundig te genereren zonder gebruik te maken van geïmporteerde externe 3D-modellen, bouwt het systeem procedureel een gebogen *mesh* op (vergelijkbaar met de bolling van een contactlens).

Aangezien Unity standaard geen ingebouwd object bezit voor een gebogen vlak, genereert het systeem deze vorm zelf via code. Dit gebeurt in de volgende stappen:

- **Generatie van het ruitennet:** Er wordt een plat tweedimensionaal raster van coördinaten berekend. Elk segment in dit raster bestaat uit twee driedimensionale driehoeken (en telt dus 6 hoekpunten per vierkant).
- **Normalisatie:** Omdat een plat vlak hoekpunten heeft die diagonaal verder weg liggen van het middelpunt, wordt elk hoekpunt geprojecteerd naar een vaste Z-diepte. Vervolgens wordt op elke vector wiskundige normalisatie (`.normalized`) toegepast. Deze bewerking dwingt elk punt om exact even ver van de oorsprong (het middelpunt) te liggen. Hierdoor plooit het platte vlak zich samen tot een oppervlak dat exact de curve van een sphere volgt.
- **Alpha Fading:** Naast de geometrie wordt er per hoekpunt een transparantiewaarde (alpha) berekend. Het middelpunt van de lens is ondoorzichtig, maar de randen vervagen vloeiend naar volledig transparant, wat een onnatuurlijk effen vlak voorkomt.

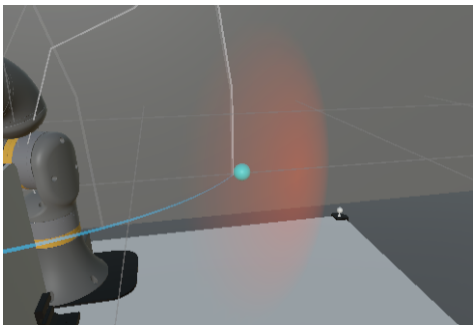
Hoewel het schild eenmalig wordt geconstrueerd bij de initialisatie, bepaalt de actieve monitoringlogica wanneer en hoe het wordt getoond. Het schild wordt enkel zichtbaar wanneer de node de buitenste 20% van het maximale armbereik betreedt. Het schild roteert voortdurend mee zodat het altijd richting de node wijst. Naarmate de node de absolute grens ($L2 + L3$) nadert, verandert de kleur via een wiskundige interpolatie (`Color.Lerp`) vloeiend van een gedimde oranje waarschuwingskleur naar een felrode kleur.

Error Tether. Wanneer de gebruiker de limiet volledig overschrijdt en de `PathNode` fysiek onbereikbaar wordt, volstaat een oplichtend schild alleen niet langer. Men moet in de 3D-ruimte kunnen inschatten hoe ver de node buiten het schild is geplaatst om dit efficiënt te kunnen corrigeren. Hiervoor is het *error tether*-component ontwikkeld.

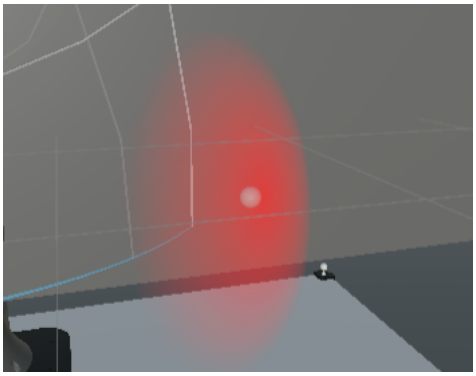
De error tether is een gerichte, visuele band die werkt als een gespannen rode laser. Het mechanisme werkt als volgt:

- **Intersectie berekenen:** Zodra het doel onbereikbaar is, berekent de rekenlaag een projectievector vanuit de schouder van de basis tot aan de fysieke limiet van het zojuist besproken schild. Dit bepaalt het exacte 3D-intersectiepunt op de buitengrens van de sphere.
- **Visuele Verbinding:** De laserlijn wordt getrokken vanaf dit vastgezette grenspunt direct naar het zwevende doel (de Node). Aan de kant van de grens is de lijn dik en felrood, waarna hij transparanter en dunner wordt richting de Node.

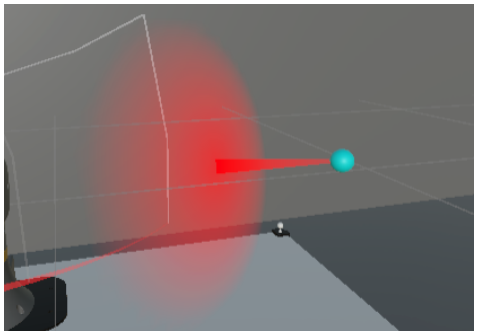
Wanneer de operator de waarschuwingen negeert en de node buiten de maximale reikwijdte plaatst, activeert de interface deze visuele verbindingslijn. Deze lijn loopt van het botsingspunt op de sphere tot aan de node en visualiseert exact de overschrijding. Door de lengte van deze lijn en de bijbehorende afstandsweergave krijgt de operator direct inzicht in de benodigde correctie om de node weer binnen de bereikbare zone te navigeren. Het stapsgewijze verloop van deze ruimtelijke waarschuwingen is samengevat in Figuur 6.



(a) Oranje schild



(b) Rood schild



(c) Error Tether

Fig. 6: Visualisaties voor de ruimtelijke reikwijdte: (a) het schild gloeit bij benadering rustig op en neemt gradueel toe in intensiteit als indicatie dat de grens genaderd wordt zonder dat er sprake is van een onmiddellijke bedreiging, (b) kleurt felrood op de maximale fysieke grens, en (c) activeert een lineaire error tether zodra de node onbereikbaar wordt geplaatst.

Kritische Bespreking en Evaluatie

Sterke punten

De kracht van de huidige implementatie ligt in de directe vertaling van de onderzoeksvragen naar functionele visualisaties. Voor elk van de drie pijlers: trajectvoorspelling, botsingsbewustzijn en ruimtelijk inzicht, is een specifieke visuele uitwerking ontwikkeld. Door de arm te clonen en het pad op te splitsen in twee aparte datastromen (`trajectoryPoints` voor de end-effector en `ghostPoses` voor de volledige armconfiguratie) wordt een hoge mate van visuele feedback gegenereerd op relatief simpele basis. Het gebruik van *object pooling* voor de ghost skeletons waarborgt bovendien een manier om het aanmaken van GameObjects zo min mogelijk te maken, al gebruikmakende van een simpele optimalisatietechniek.

Tekortkomingen en Limitaties

Hoewel de huidige visualisaties effectief zijn in hun eenvoud, is er ruimte voor meer creatieve of abstracte weergavevormen die de intuïteit en behulpzaamheid van het systeem verder kunnen verbeteren. Een significante technische beperking tijdens de tests was de *AR jitter*, die de accuraatheid en de professionele uitstraling van de interface op de Magic Leap 2 negatief beïnvloedde. Er is geëxperimenteerd met het verlagen van de herkalibratie-frequentie van de AprilTags (bijvoorbeeld enkel bij significante verplaatsing), maar deze oplossing bleek onvoldoende. Het is echter de vraag of de tracking-frequentie überhaupt de hoofdoorzaak was van de instabiliteit, of dat een andere factor de hoofdrol speelde.

Daarnaast vormt de reductie tot een 3-DOF (Degrees of Freedom) model een beperking voor de volledige benutting van de robotcapaciteiten. Hoewel de ontwikkeling van de `IK_Calculator` een behoorlijke tijd vergde om tot een stabiel resultaat te komen met beperkte kennis, beschikt de fysieke robotarm over een 6-DOF bereik. Door de focus te beperken tot de positionering van de end-effector, is de volledige bewegingsvrijheid van de robot, zoals polsoriëntaties, momenteel onbenut gebleven. Een volledige 6-DOF implementatie had potentieel geleid tot meer gedetailleerde visualisaties, waarbij ook de oriëntatiedynamiek en niet enkel de positionele IK van de robotarm een rol had kunnen spelen in de visuele feedforward.

Conclusie en Reflectie

Wegens het ontbreken van een geïntegreerde aanpak om een visuele feedforward weer te geven, is er een AR systeem gerealiseerd met als doel de operator vooraf inzicht te geven in geplande robotbewegingen. Binnen deze oplossing zijn de drie vooropgestelde pijlers succesvol tot stand gebracht: de trajectvisualisatie, het botsingsbewustzijn en het ruimtelijk inzicht. De fundamentele kernbijdrage van dit werk ligt in de integratie van deze drie afzonderlijke pijlers binnen één samenhangende AR-omgeving.

Hoewel het functionele prototype hiermee technisch is gerealiseerd, is de daadwerkelijke effectiviteit voor de operator nog niet empirisch getoetst. Om deze beperking te overbruggen,

zou toekomstig werk zich in de eerste plaats kunnen richten op het uitvoeren van een dergelijke gebruikersstudie. Op technisch vlak is bovendien een uitbreiding van de visualisatie van 3-DOF naar een volledige 6-DOF aan de orde. Daarnaast zou de resterende tracking jitter rondom de fiducial markers nog verder onderzocht kunnen worden. Tot slot valt te denken aan een validatie die het voorspelde AR-traject vergelijkt met de werkelijk uitgevoerde traject.

Richting de toekomst neem ik waardevolle ervaring mee rondom het formele schrijfgedrag dat vereist is voor academische artefacten zoals een paper en poster. Op het gebied van planning heeft dit deze opdracht het belang aangetoond van tijdig starten met omvangrijke taken en het actief onderhouden van een constant feedbackmechanisme. Daarnaast is mijn technische kennis verbreed op het vlak van visuele feedforward en inverse kinematics.

Acknowledgments

Ik wil mijn promotor Prof. Dr. Kris Luyten en begeleider Dries Cardinaels bedanken voor hun begeleiding en feedback. Ook dank aan Digital Future Lab voor het ter beschikking stellen van de Magic Leap 2 hardware en de ondersteuning bij het project.

Gebruik van AI-tools

Tijdens het werken aan deze thesis is gebruikgemaakt van Gemini als ondersteunend hulpmiddel. In de schrijffase werd de tool ingezet voor taalondersteuning, zoals het controleren van grammatica en het verbeteren van de algemene leesbaarheid. Tijdens de implementatiefase is Gemini gebruikt om mogelijke code-implementaties te verkennen, foutmeldingen te begrijpen en de wiskundige achtergrond rondom inverse kinematics en de 3D-visualisaties in Unity te verduidelijken. De gegenereerde output werd nooit rechtstreeks overgenomen, maar steeds handmatig gecontroleerd, getest en aangepast. Alle inhoudelijke keuzes, de software-architectuur en de uiteindelijke conclusies blijven volledig mijn eigen verantwoordelijkheid.

References

[1] Y. Zhang, "The impact of mechanical arm applications on industrial intelligent transformation," *Applied and Computational Engineering*, 2025.

[2] S. Robla-Gómez, V. Becerra, J. R. Llata, E. Gonzalez-Sarabia, C. Torre-Ferrero, and J. Pérez-Oria, "Working together: A review on safe human-robot collaboration in industrial environments," *IEEE Access*, vol. 5, pp. 26754–26773, 2017.

[3] V. Villani, F. Pini, F. Leali, and C. Secchi, "Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications," *Mechatronics*, 2018.

[4] C. Zhou, B. Huang, and P. Fränti, "A review of motion planning algorithms for intelligent robots," *Journal of Intelligent Manufacturing*, vol. 33, pp. 387 – 424, 2021.

[5] M. G. Tamizi, M. Yaghoubi, and H. Najjaran, "A review of recent trend in motion planning of industrial robots," *International Journal of Intelligent Robotics and Applications*, vol. 7, pp. 253–274, 2023.

[6] J. Liu, H. Yap, and A. S. M. Khairuddin, "Review on motion planning of robotic manipulator in dynamic environments," *Journal of Sensors*, 2024.

[7] S. Kumar, C. Savur, and F. Sahin, "Survey of human–robot collaboration in industrial settings: Awareness, intelligence, and compliance," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, pp. 280–297, 2021.

[8] J. Wilhelm, V. Tibo, and M. Freitag, "Beyond supervision: Reviewing operator types and their requirements in the context of human-automation interaction," *Procedia CIRP*, vol. 130, pp. 510–515, 2024. 57th CIRP Conference on Manufacturing Systems 2024 (CMS 2024).

[9] E. Rosen, D. Whitney, E. Phillips, G. Chien, J. Tompkin, G. Konidaris, and S. Tellex, "Communicating and controlling robot arm motion intent through mixed-reality head-mounted displays," *The International Journal of Robotics Research*, vol. 38, pp. 1513 – 1526, 2019.

[10] H. Fang, S. Ong, and A. Nee, "Interactive robot trajectory planning and simulation using augmented reality," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 2, pp. 227–237, 2012.

[11] C. P. Quintero, S. Li, M. K. Pan, W. P. Chan, H. Machiel Van der Loos, and E. Croft, "Robot programming through augmented trajectories in augmented reality," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1838–1844, 2018.

[12] J. Chong, S. Ong, A. Nee, and K. Youcef-Youmi, "Robot programming using augmented reality: An interactive method for planning collision-free paths," *Robotics and Computer-Integrated Manufacturing*, vol. 25, no. 3, pp. 689–701, 2009.

[13] M.-E. Matour and A. Winkler, "Intuitive robot path planning through augmented reality," in *2023 27th International Conference on Methods and Models in Automation and Robotics (MMAR)*, pp. 27–32, 2023.

[14] U. Gruenefeld, L. Prädel, J. Illing, T. Stratmann, S. Drolshagen, and M. Pfingsthorn, "Mind the arm: realtime visualization of robot motion intent in head-mounted augmented reality," in *Proceedings of Mensch Und Computer 2020*, MuC '20, (New York, NY, USA), p. 259–266, Association for Computing Machinery, 2020.

[15] S. Chacko and V. Kapila, "Making robots understandable: Augmented reality for enhancing situational awareness in human–robot co-located environments," *Empathic Computing*, 01 2025.

[16] M. Walker, H. Hedayati, J. Lee, and D. Szafir, "Communicating robot motion intent with augmented reality," pp. 316–324, 02 2018.

[17] M. Safeea, P. Neto, and R. Bearee, "On-line collision avoidance for collaborative robot manipulators by adjusting off-line generated paths: An industrial use case," *Robotics and Autonomous Systems*, vol. 119, pp. 278–288, 2019.

[18] B. Lacevic, A. M. Zanchettin, and P. Rocco, "Safe human-robot collaboration via collision checking and explicit representation of danger zones," *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 2, pp. 846–861, 2023.

[19] H. Orsolits, A. Saliger, and A. Korn, "Interactive mixed reality visualization of dynamic safety zones in human-robot collaboration," in *2025 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, pp. 450–453, 2025.

[20] S. Demirtas, T. Cankurt, and E. Samur, "Development and implementation of a collaborative workspace for industrial robots utilizing a practical path adaptation algorithm and augmented reality," *Mechatronics*, vol. 84, p. 102764, 2022.

[21] W. W. Loy, J. Donovan, M. Rittenbruch, and M. Teixeira, "Exploring ar-enabled human–robot collaboration (hrc) system for exploratory collaborative assembly tasks," *Construction Robotics*, vol. 9, 08 2025.

[22] J. Wang and E. Olson, "Apriltag 2: Efficient and robust fiducial detection," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4193–4198, 2016.

[23] Unity Technologies, "Use object pooling to boost performance of C# scripts in Unity." https://learn.unity.com/course/design-patterns-unity-6/tutorial/use-object-pooling-to-boost-performance-of-c-scripts-in-unity, 2024. Geraadpleegd op: 17 mei 2026.