MÄLARDALENS HÖGSKOLA
AKADEMIN FÖR INNOVATION, DESIGN OCH TEKNIK
VÄSTERÅS, SVERIGE

DVA331 – Examensarbete för kandidatexamen i datavetenskap

INSTRUKTIONER VIA FÖRSTÄRKT VERKLIGHET FÖR INDUSTRIELLT UNDERHÅLL

Rickard Lundberg
rlg15001@student.mdh.se

Pontus Wedén
pwn15002@student.mdh.se

Handledare: Caroline Uppsäll
Mälardalens högskola, Västerås, Sverige

Handledare: Lotta Lindberg
byBrick Interface AB, Badhusgatan 8, 722 15 Västerås, Sverige

Examinator: Gordana Dodig-Crnkovic
Mälardalens högskola, Västerås, Sverige

2018-05-22

Sammanfattning

Förstärkt verklighet (hädanefter förkortat AR efter engelskans augmented reality) är när virtuellt blandas med verkligt, och används till exempel för mobilspel och inom det militära. Utvecklingen av AR-applikationer har under de senaste åren förenklats, vilket gett upphov till nya möjligheter för tillämpning av AR-tekniken. Syftet med vårt arbete är att undersöka om instruktioner givna via AR bidrar till att användare gör färre fel när de utför industriellt underhåll. Vi har valt att jämföra AR-instruktioner med pappersmanualer, då det senare är det som oftast används inom industrin idag.

Det finns ett intresse inom industrin för att integrera AR-lösningar i deras verksamhet, inte minst för att det potentiellt kan minska tiden det tar att åtgärda driftstopp, men även för att kunna hjälpa oerfarna och nya operatörer. Idag finns det ett flertal hjälpmedel och programmeringsbibliotek som har gjort utveckling av AR-applikationer enklare, vilket öppnar upp för fler möjligheter när det gäller AR:s tillämpningsområden.

För att undersöka om AR-instruktioner kan bidra till att användare gör färre fel har vi utvecklat en AR-applikation till mobiltelefon. I ett experiment jämför vi applikationen mot en pappersmanual på en egenkonstruerad industriell underhållsuppgift. Vi har visat att med hjälp av vår AR-applikation gjorde användare i snitt drygt 60% färre fel. Vi genomförde även ett test för användarvänlighet där applikationen fick ett resultat som var ett poäng under det som anses vara helt tillfredställande.

Abstract

Augmented reality (AR) is when the virtual is mixed with the real, AR is used in for example mobile games and the military. The development of AR-applications has been simplified during recent years, which has opened possibilities for new applications of the AR-technology. The purpose of our work is to research whether instructions given via AR leads to the users making fewer errors when performing industrial maintenance. We chose to compare AR-instructions to paper manuals since this is the most commonly used instruction medium today.

Companies have shown interest in the development of AR-solutions for industrial maintenance in their day-to-day operations. Particularly since it can potentially lead to decreased downtime and help new or inexperienced operators. Today there exists several aids and programming libraries that has simplified the development of AR-applications, which opens for more possibilities when it comes to the areas of application for AR.

To examine if AR-instructions leads to fewer errors made by the users, we have developed an AR-application for mobile phones. In our experiment we compare the AR-application with a paper manual on a custom-built maintenance task. Our results show that the users with the AR-application perform just over 60% fewer errors. To verify the usability of the application we conducted a usability test, the results from this test puts our application one point below the acceptable limit.

Innehållsförteckning

1. Introduktion **6**

2. Bakgrund **6**

2.1. AR *6*

2.2. Spårning *7*

2.3. Rendering *7*

2.4. Industriellt underhåll *7*

2.5. System Usability Scale *7*

3. Relaterat arbete **8**

3.1. AR-instruktioner vid samarbete *9*

4. Problemformulering **10**

4.1. Begränsningar *10*

5. Metod **10**

6. Forskningsetiskt ställningstagande **11**

7. Experiment **11**

7.1. Förstudie hos Volvo Powertrain AB *11*

7.2. Utformning av experiment *12*

7.2.1. Design av uppgift 12

7.2.2. Konstruktion av utrustning 13

7.2.3. Konstruktion av pappersmanual 14

7.2.4. Konstruktion av AR-applikation 14

7.2.5. Testning av AR-applikation 16

7.2.6. Förbättring av AR-applikation 16

7.3. Utförande av experiment *17*

8. Resultat **18**

8.1. Observationer *19*

9. Diskussion **20**

10. Slutsats **21**

11. Framtida arbete **21**

12. Referenser **21**

Bilaga A. Pappersmanual **24**

Bilaga B. Programkod för markörgenerering **26**

Bilaga C. Programkod för AR-applikation **28**

Bilaga D. Bilder från AR-applikationen **43**

Bilaga E. Svar från användartest **47**

1. Introduktion

Förstärkt verklighet (förkortat AR efter engelskans augmented reality) är en teknik som gör det möjligt att placera ut virtuella element i verkligheten. Grundidén har funnits länge, men det är inte förrän nu på 2010-talet som tekniken har börjat få en användning för gemene man [1]. Moderna exempel på tillämpningar av AR är mobilspelet Pokémon GO [2], Microsofts HoloLens [3] och hjälpmedel i stridsfordon [4]. I takt med att AR fått mer uppmärksamhet har också utvecklingen av AR-program förenklats. För mobilapplikationer finns det till exempel färdiga programmeringsbibliotek såsom Googles ARCore [5], Apples ARKit [6] och PTCs Vuforia [7]. Bibliotek som dessa har gjort AR-teknik tillgänglig för fler utvecklare, någon expertkunskap inom området krävs inte längre.

Tidigare arbeten har visat att det kan finnas fördelar med AR-instruktioner, men det finns också forskning som påvisar motsatsen. Sanna et al. [8] visar på att AR-instruktioner ger färre fel vid utförande av en underhållsuppgift, däremot visar Havard et al. [9] på att AR-instruktioner kan orsaka fel, då testdeltagare missade vissa instruktioner i deras test. I detta arbete har vi valt att undersöka huruvida AR-instruktioner kan minska antalet fel vid underhållsarbeten. I dagsläget förlitar sig företag ofta på arbetarnas expertis och instruktioner givna via konventionella instruktionsmetoder, såsom pappers- eller videoinstruktioner. För att undersöka frågan har vi skapat en mobilapplikation för AR-instruktioner. Applikationen har använts i användartest där vi undersökt om användare gör färre fel med den jämfört med pappersinstruktioner. Vår initiala hypotes var att AR kan erbjuda ett tydligare sätt att presentera instruktioner och därmed också minska antal fel vid industriellt underhållsarbete.

Rapporten är strukturerad enligt följande: Avsnitt 2 presenterar en bakgrund till området AR och vad det är, där definieras även för rapporten viktiga begrepp såsom industriellt underhåll och avsnittet avslutas med en översikt över den metod som kommer användas för användarvänlighetstestet. Avsnitt 3 går igenom relaterade arbeten inom AR-instruktioner för industriellt underhållsarbete. Problemformulering och frågeställningar presenteras i avsnitt 4, följt av avsnitt 5 där den metod som kommer användas för att besvara frågorna beskrivs. Forskningsetiska ställningstaganden presenteras i avsnitt 6. Avsnitt 7 respektive 8 beskriver arbetet och resultatet, vilket sedan följs av diskussion och slutsats i avsnitt 9 respektive 10. Slutligen presenteras framtida arbete i avsnitt 11.

2. Bakgrund

I detta avsnitt presenterar vi begrepp och tekniker vi använder i rapporten. Här presenteras vad AR innebär och tekniker för spårning och rendering samt en definition av industriellt underhåll. Sist i avsnittet beskriver vi metoden för användarvänlighetstestet.

2.1. AR

AR är ett system som har följande tre egenskaper: det kombinerar verkligt och virtuellt, är interaktivt i realtid och registreras i tre dimensioner [10]. AR är således en blandning av verkligheten och datorgenererad grafik. Med tekniken är det möjligt att placera ut virtuella objekt på till exempel en vägg eller ett golv. I Figur 1 illustreras detta med en virtuell stol placerad i ett rum med hjälp av AR-teknik. Virtuella objekt som placerats ut med hjälp av AR bibehåller sin position i relation till verkligheten. Stolen i Figur 1 står kvar på samma plats i rummet även när kameran flyttas eller riktas åt ett annat håll.

Figur 1. Virtuell stol placerad i ett verkligt rum med hjälp av AR-teknik.

AR-tekniken är inte bunden till någon specialanpassad hårdvara, det är till exempel möjligt att använda en vanlig dator [11] eller mobiltelefon [2]. För att användaren ska ha båda händerna fria, och alltid se AR-informationen så kan glasögon med inbyggd skärm [12] användas.

2.2. Spårning

För att kunna avgöra hur den genererade grafiken ska visas måste AR-programvaran känna till var kameran är placerad i förhållande till den omgivning användaren befinner sig i. Den information programvaran behöver för detta får den genom en process som kallas spårning [13]. Spårningen kan genomföras genom att programvaran söker efter markörer [14] eller objekt [15] i en bild från kameran. Dessa två spårningstekniker benämns generellt som markör respektive markörlös spårning. Genom att jämföra den funna markören i kamerabilden med originalbilden av samma markör kan programvaran avgöra kamerans placering i det tredimensionella rummet. Resultatet av detta är en transformationsmatris som kan användas för att placera ut och omforma virtuella objekt för att passa i verkligheten i förhållande till kamerans position.

Det finns olika tillgängliga utvecklingsverktyg för att förenkla arbetet med spårning i AR. Ett av dem är Vuforia [7] som tillåter spårning med markörer, objekt eller en blandning av dessa för mobiltelefoner. Utvecklingsverktyg för spårning är inte det enda som krävs för att kunna implementera AR. När kameran har hittat den plats i verkligheten ett objekt ska placeras på så måste det även renderas, något som till exempel Vuforia inte klarar av. Det är därför vanligt att man implementerar AR med hjälp av två tekniker: en spårare och en renderare.

2.3. Rendering

Renderarens uppgift är att rita ut den virtuella grafiken. För att göra detta används antingen bilder eller 3D-modeller. I de fall 3D-modeller används behöver renderaren information dels om objektets form och dels om objektets utseende, såsom färg och hur reflekterande objektet är. Genom att applicera en transformationsmatris på objektet som ska genereras kan renderaren få objektet att se ut att vara placerad i den verkliga miljön på ett trovärdigt sätt.

Använder man Vuforia som spårningsteknik är det vanligt att man använder spelmotorn Unity för att rendera objekten i verkligheten, eftersom spelmotorns ovanliggande plattform har Vuforia inbyggt [16]. Det inbyggda stödet för Vuforia förenklar även arbetet med att utveckla AR-applikationer. Unity erbjuder ett grafiskt gränssnitt tillsammans med möjligheten att skriva egna skript för att styra interaktionen mellan olika programelement såväl som mellan program och användare. Skripten kan programmeras med språken C# och JavaScript. Ett program som utvecklats med Unity kan köras på flera olika plattformar utan att behöva anpassas för respektive plattform.

2.4. Industriellt underhåll

Industriellt underhåll är en bred term som kan inkludera många olika aktiviteter inom industrin. Vår definition av industriellt underhållsarbete baseras på erfarenheter från en förstudie hos Volvo Powertrain AB som presenteras i avsnitt 7.1. Vi definierar industriellt underhåll dels som förebyggande underhåll och dels som reparation och service av materiel. Förebyggande underhåll innebär regelbunden kontroll och rutinmässigt utbyte av maskindelar. Reparation och service är sådant underhållsarbete som utförs vid behov, till exempel då ett verktyg eller en maskin går sönder.

2.5. System Usability Scale

System Usability Scale (SUS) [17] används för att utvärdera användbarheten hos system, däribland applikationer. Den består av ett formulär med fördefinierade påståenden på engelska där respondenterna har fem svarsalternativ mellan att starkt hålla med och starkt inte hålla med. Påståendena i formuläret som respondenterna ombeds ta ställning till är följande:

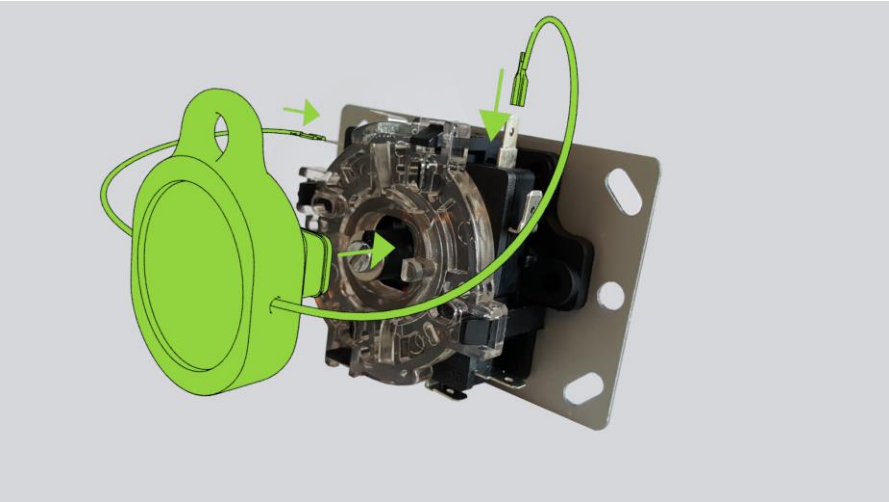
1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.

5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

SUS ger ett resultat i intervallet 0–100, där högre poäng tyder på högre användbarhet. Ett resultat på minst 70 anses vara acceptabelt och 68 är medelvärdet [18].

3. Relaterat arbete

Ett sätt att använda AR-instruktioner är att presentera de olika steg och moment som finns i motsvarande pappersmanual. Användaren får då instruktioner i form av text, symboler och 3D-modeller, till exempel som i Figur 2 där de gröna elementen är datorgenererade instruktioner. Sanna et al. [8] har skapat ett sådant system för AR-instruktioner och har gjort tester som innefattade att byta hårddisk i en bärbar dator. Deras tester visar att systemet är mest gynnsamt för användare som inte har tidigare erfarenhet av den uppgift som ska utföras. En testgrupp som inte hade tidigare erfarenhet genomförde uppgiften snabbare med AR-instruktioner jämfört med när pappersmanual användes. För en annan testgrupp bestående av studenter inom datavetenskap hade AR-systemet motsatt effekt på tidsåtgången. Oavsett tidigare erfarenhet visar användartesterna som Sanna et al. gjort att testdeltagarna gör färre fel när AR-instruktioner används i jämförelse med pappersmanual.



Figur 2. Bild som visar hur ett AR-system kan se ut.

Aromaa et al. [19] har skapat ett liknande system som Sanna et al. men genomförde tester på två erfarna tekniker med uppgifter de var vana att arbeta med. Resultaten visar att testdeltagarna kände att AR-instruktionerna var bättre bland annat för att AR är mer visuellt och inte har den språkbarriär som pappersmanualer kan ha. Däremot uttryckte testdeltagarna en viss oro över att utrustningen som används kan vara för känslig för att användas i en industriell miljö.

Quint et al. [20] har genomfört en studie med AR-instruktioner för underhåll på branddörrar i en industriell miljö. Testdeltagarna menar att systemet var effektivt för underhållsarbete eftersom det möjliggör för oerfarna operatörer att utföra arbetet. Deltagarna ansåg att arbetet med att uppdatera AR-instruktioner är för krävande, därför tyckte de att systemet endast passar för processer som sällan ändras. De nämner också att AR inte kan ersätta expertis hos erfarna operatörer.

AR-tekniken gör det möjligt för användaren att röra sig fritt och titta på objekt från olika håll. Samtidigt som användaren rör på sig sitter instruktionerna, som till exempel kan vara pilar, kvar på rätt plats. Havard et al. [9] visar att detta inte alltid är en fördel. Deras tester visar bland annat att flera testdeltagare missade delar av ett instruktionssteg när de använde AR. Vid ett steg skulle tre slangar kopplas bort, två på högra sidan och en på den vänstra sidan om ett objekt. Flera deltagare missade då den vänstra slangen och i intervjuer efter testet hänvisade de till att de inte sett den.

Re och Bordegoni [21] har skapat ett liknande system som de ovan nämnda för AR-instruktioner till en ensamarbetande person. Det som skiljer sig i deras system är att de utöver att ge instruktioner även låter AR-utrustningen skicka vidare videoströmmen till en server som lagrar den. Det är också möjligt för en annan person att övervaka arbetet i realtid via denna videoström. Detta system kräver en nätverksuppkoppling vilket de andra system vi nämnt inte kräver. Re och Bordegoni menar dock att övervakningen är en effektiv metod för att kontrollera arbetet och ger en möjlighet att hitta misstag. Systemet gör det också möjligt att i efterhand gå tillbaka och kontrollera hur ett arbete har utförts.

Ovan nämnda studier visar både potentiella fördelar och nackdelar med AR som instruktionsmedium. Sanna et al. visar att det kan minska antalet fel en användare gör medan Havard et al. visar att AR kan försvåra för användaren. Enligt Aromaa et al. finns det en viss oro över hållbarheten på utrustningen som krävs för AR när den ska användas i en industriell miljö. Quint et al. belyser också en negativ aspekt med AR då deltagarna nämner att kontinuerlig uppdatering av AR-instruktioner kan vara för krävande och därför inte passar alla typer av arbeten. Aromaa et al. beskriver att användare av AR-instruktioner upplever det som ett bra medium då det är visuellt och möjligtvis inte kräver samma mängd förklarande text som det kan krävas i en pappersmanual. Quint et al. nämner att en fördel med AR-instruktioner är att det kan möjliggöra för oerfarna att utföra arbeten.

3.1. AR-instruktioner vid samarbete

Om en tekniker som utför underhållsarbete stöter på ett problem hen inte kan lösa är det vanligt att personen kan kontakta en expert, antingen hos tillverkaren eller inom det egna företaget. Kommunikationen mellan en tekniker som är på plats och en expert sker vanligtvis via mobiltelefon idag. Ett alternativ till kommunikation via mobiltelefon är att använda AR för samarbete mellan tekniker och expert. För detta har vi funnit i huvudsak två vägar att gå: direktkontakt med en expert och en metod som kan beskrivas som beställning av hjälpinstruktioner.

För att kunna ge instruktioner i realtid via AR behöver experten få information om den utrusning teknikern arbetar med. För att experten ska kunna placera ut virtuella instruktionselement krävs också att hen får information om den omgivning teknikern befinner sig i. Detta kan ske genom en videoström [22-25] eller fotografi [26]. Informationen är också en viktig del för att experten ska kunna diagnostisera och inspektera utrustningen som behöver underhåll [26].

Realtidssystemen skapade av [22, 23, 25, 26] grundar sig i samma idé: en expert tar emot en videoström från en tekniker och lägger sedan in instruktioner via ett program på en dator. När en instruktion är skapad skickas den till teknikern. Instruktionerna kan bestå av text eller grafik i form av pilar eller bilder på verktyg som ska användas. I dessa system kan de inblandade även prata med varandra under tiden instruktionerna skapas och utförs. Det är inget nämnvärt som skiljer lösningarna från [22, 23, 25, 26] i hur tekniker och expert kommunicerar. Däremot låter Zenati-Henda et al. [24] experten, istället för att ge instruktioner genom grafik, ge instruktioner via handgester. Handgesterna tolkas av systemet som sedan skickar dem som virtuella representationer av handgesterna till användaren.

Porcelli et al. [22] menar att kommunikationen blir effektiv med deras realtidssystem för AR-samarbete. Detta är något som bekräftas av test utförda av Zenati et al. [23]. Däremot lyfter Porcelli et al. specifikt en kommentar från en deltagare om att det kan bli problem att få kunder att vilja koppla upp sina maskiner mot Internet. Författarna menar att det kan vara nödvändigt att utveckla ett protokoll för säker

datatransport, innan AR kan användas i produktion. Wang et al. [25] fokuserar stora delar av sin forskning på hur kommunikationen ska lösas, däremot är deras fokus på att organisera och minska mängden data som skickas.

Mourtzis et al. [27] har tagit fram ett AR-system för samarbete som inte är i realtid. Det påminner mer om ett konventionellt ärendehanteringssystem. En tekniker som är på plats för att utföra ett arbete kan samla in data i form av beskrivning av problemet, bilder och ljud. Denna data skickas sedan som ett ärende till experter som skapar en specialanpassad AR-instruktion baserat på den data som skickats. När teknikern får AR-instruktionen genomför hen arbetet utan direkt kommunikation med en expert.

4. Problemformulering

Målet med vårt arbete är att undersöka användbarheten av AR-teknik som instruktions- och stödmaterial vid industriellt underhåll, då tidigare resultat från liknande studier har visats skilja sig. Vi kommer närmare bestämt undersöka om AR-instruktioner kan bidra till att användarna gör färre fel i jämförelse med när pappersmanual används. Vidare vill vi undersöka om AR-instruktioner via en mobiltelefon upplevs användarvänligt. För att undersöka användarvänligheten kommer vi använda oss av System Usability Scale (SUS) [17]. Skalan går från 0–100 poäng, där högre är bättre och minst 70 poäng ses som acceptabelt [18].

Frågorna vi vill besvara i arbetet är:

- Gör användarna färre fel när de använder AR-instruktioner på mobiltelefon jämfört med pappersmanualer vid industriellt underhåll?
- Är AR-tekniken på mobiltelefon tillräcklig för att uppnå 70 poäng på SUS med vår implementation?

4.1. Begränsningar

Vi kommer inte kunna utföra vårt arbete i en industriell miljö eller med manualer och utrusning från industrin. Därför begränsas vårt arbete till vår förmåga att skapa material som återspeglar den som används inom industrin. Vårt arbete begränsas också av den tid vårt arbete pågår vilket innebär att vi begränsat oss till tio testdeltagare.

5. Metod

För att kunna mäta om AR-instruktioner bidrar till att användarna gör färre fel skapade vi en applikation för AR-instruktioner. Då arbetets fokus var AR-instruktioner via en mobiltelefon valde vi att skapa en mobilapplikation. Denna applikation var inte ett komplett system för AR-instruktioner, utan funktionaliteten var begränsad till det som krävdes för att genomföra vårt experiment.

Då målet inte var att undersöka utvecklingstekniker för AR, fokuserade vi på att hitta enkla utvecklingsverktyg. Detta resulterade i att vi valde Vuforia och Unity, eftersom de erbjuder ett användarvänligt gränssnitt för att utveckla AR-applikationer. Vuforia används som ett insticksprogram i Unity för hantering av spårning. Utvecklingen i Unity skedde med programmeringsspråket C#. Unity är plattformsoberoende, däremot har vårt fokus varit att skapa en applikation för Android.

För att mäta om det finns en skillnad i det antal fel en användare gör när AR-instruktioner används mot om pappersmanualer används har vi genomfört ett användartest. Testdeltagarna fick använda sig av en mobiltelefon där instruktionerna presenterades via AR. Testuppgiften bestod i att genomföra underhåll i ett apparatskåp. Som kontroll genomförde vi samma test med andra deltagare som fick instruktioner via en pappersmanual. För att minimera missförstånd och överraskningsmoment fick de testdeltagarna som använde AR-instruktioner en introduktion till AR-tekniken innan testet genomfördes.

Både apparatskåpet och pappersmanualen skapade vi på ett sådant vis att de återspeglar ett möjligt verkligt scenario inom industriellt underhåll. Under utvecklingsprocessen har vi konsulterat olika personer

som arbetar inom industrin, har erfarenhet av industriellt underhåll eller är kunniga inom teknisk dokumentation.

För att kunna skapa en adekvat pappersinstruktion har vi byggt upp alla delar av vår testutrustning som 3D-modeller. Vi skapade modellerna i Blender [28], som är ett 3D-modellerings- och renderingsprogram. Vår erfarenhet är att det är vanligast att CAD-program används för att återskapa modeller till teknisk dokumentation. Blender är inte ett CAD-program, trots detta valde vi Blender då vi visste att vi kunde få det resultat vi eftersökte med det programmet. En viktig faktor i valet var också att vi har stor erfarenhet i att arbeta med Blender och att de modeller vi skapade i det programmet gick att använda i Unity.

Vi har observerat alla test och registrerat antalet fel testdeltagarna gjorde samt den tid uppgiften tog för dem. För att ge en bredare grund vid senare analys ställde vi frågor om uppgiftens komplexitet och användarnas upplevelse av AR-applikationen respektive pappersmanualen. Testerna spelades in med videokamera för att i efterhand kunna säkerställa att testet utförts korrekt. De deltagare som genomfört testet med AR-applikationen fick fylla i ett formulär för System Usability Scale. Detta formulär använde vi för att mäta användbarheten av vår AR-applikation.

6. Forskningsetiskt ställningstagande

För våra användartester så har deltagarna informerats om följande punkter:

- Allt deltagande är frivilligt och kan avbrytas när som helst.
- Inga uppgifter som lämnas kommer kopplas till uppgiftslämnaren.
- Inga enskilda uppgifter kommer föras vidare till tredje part.
- Uppgifterna kommer enbart användas i vår forskning.
- Syftet med användartestet och SUS-formuläret.
- Användartestet kommer att filmas och enbart vi kommer att se filmen.

De företag och personer vi sammarbetat med har givits möjligheten att vara anonyma i vårt arbete.

7. Experiment

Detta avsnitt redogör för teorin bakom utformningen av experimentet samt det praktiska arbetet för att utföra det. Vi inleder med resultatet av en förstudie vilket följs av utformningen av vårt experiment. Slutligen presenterar vi genomförandet av experimentet.

7.1. Förstudie hos Volvo Powertrain AB

För att undersöka vilka utmaningar som finns när AR används inom industrin inledde vi arbetet med en förstudie hos Volvo Powertrain AB i Köping. De har börjat experimentera med AR och hur det kan tänkas tillämpas inom industrin.

Under förstudien gav Volvo en presentation av hur deras arbete med AR ser ut och deras erfarenheter av arbetet. Problemet Volvo ville lösa med hjälp av AR var att till sådan stor mån som möjligt minska tiden det tar att åtgärda ett driftstopp. De gav siffror på att ett driftstopp i värsta fall kan innebära kostnader upp mot en miljon kronor i timmen, så att minska tiden det tar att åtgärda dessa är av stort intresse för dem. För att åstadkomma detta hade Volvo ett par områden där de tänkte sig att AR kunde tillämpas. Deras främsta mål var att möjliggöra samarbete mellan en expert och en operatör, där experten kunde ge instruktioner på distans med hjälp av AR. De arbetade även väldigt mycket med checklistor vid förebyggande underhåll, de tänkte sig att AR kunde förenkla arbetet genom att operatören kan gå runt inspektionsobjektet och checka av kontrollpunkter virtuellt. Slutligen nämnde de att AR kunde vara fördelaktigt för att lära upp nyanställda eller inhyrd personal från bemanningsföretag.

Den utmaning Volvo såg var att faktiskt kunna tillämpa AR i deras industri. De ansåg att tekniken finns och är nära att kunna tillämpas, men deras industrimiljö tillåter inte operatörer att använda sig av den AR-teknologi som Volvo försöker implementera. Volvo har valt att använda Microsoft HoloLens, vilket innebär att en operatör använder sig av glasögon för att se virtuella element, för samarbete med expert använde de även så kallade in-ear-hörlurar. Detta blir ett problem då glasögonen minskar användarens synfält, och hörlurar inte får användas i industrilokalerna.

En viktig del av förstudien var att vi fick studera förhållandena i industrilokalerna. Volvo redogjorde för vilken typ av arbete som finns inom industriellt underhåll och hur det går till hos dem, vilket visade att industriellt underhåll är ett brett område. Det sträcker sig från mekanik till elektronik. De som utför underhållsarbetet gör det antingen direkt på plats eller tar med sig utrustningen till ett avskilt rum. Underhållsarbete kan innebära att byta ut utrusning eller att reparera den. Utbyte av utrusning kan ske rutinmässigt eller vid oväntade driftstopp.

7.2. Utformning av experiment

Utformningen av experimentet delas in i stegen design, konstruktion och testning av AR-applikationen, pappersmanualen och experimentuppgiften. Vi inleder avsnittet med att beskriva arbetet bakom våra designval och beskriver sedan konstruktionen av experimentutrustning, pappersmanual och AR-applikation. Sist i avsnittet beskriver vi ett test av AR-applikationen och de förbättringar som vi gjorde efter det.

7.2.1. Design av uppgift

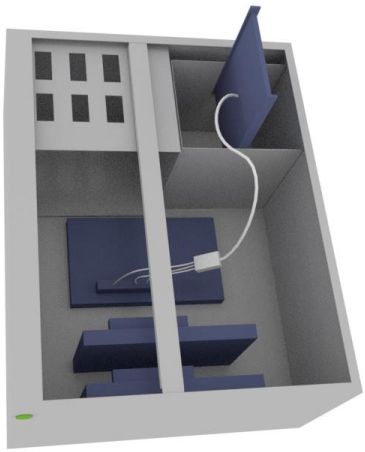
För att kunna skapa en realistisk uppgift för experimentet så inledde vi designfasen med att granska manualer för industriellt underhåll. Utifrån dessa manualer har vi främst baserat experimentuppgiften och underhållsstegen på en manual från ABB Automation Technologies AB [29]. Med erfarenheter från förstudien hos Volvo och med hjälp av manualen från ABB definierade vi 13 steg som användarna skulle utföra, dessa återfinns i Tabell 1.

STEG	BESKRIVNING	INSTRUKTION
1	Ta bort servicelucka	Skjut serviceluckan åt vänster.
2	Ta bort stödpelaren	Dra nedre del av pelaren rakt utåt från skåpet.
3	Koppla bort tregrenskontakten	Tryck in sprint och lossa vardera gren på kontakten.
4	Skruva bort anslutningskabeln från komponentsläden	Lossa båda tumskruvarna från kontakterna och dra ur kontakten.
5	Ta loss anslutningskabeln från klämmorna	Dra anslutningskabeln utåt från klämman.
6	Ta bort komponentsläden	Dra släden rakt ut i handtaget.
7	Skjut in ny komponentsläde	
8	Koppla in ny tregrenskontakt	Anslut respektive kontakt till motsvarande färgkodat uttag.

9	Skruva fast anslutning till den nya komponentsläden	Anslut kontakten till släden och spänn tumskruvarna.
10	Fäst anslutningskabeln i klämmorna	Tryck in kabeln i klämmorna hela vägen.
11	Koppla om läge på tre-grenskortet	
12	Montera stödpelare	Skjut i pelaren i spåret på den övre sidan av apparatskåpet, fäll stödpelaren mot apparatskåpet och fäst den i nedre monteringsspår.
13	Montera servicelucka	Placera luckan mot apparatskåpet med utrymme under, skjut sedan ned luckan för att fästa.

Tabell 1. Steg i experimentet.

Med utgångspunkt i stegen från Tabell 1 och ett antal insamlade referensbilder över olika sorters elektronik och apparatskåp utformande vi den design för apparatskåpet som återfinns i Figur 3. Komponentsläden syns i det övre högra hörnet på bilden och är ansluten via tregrenskabeln till apparatskåpet.



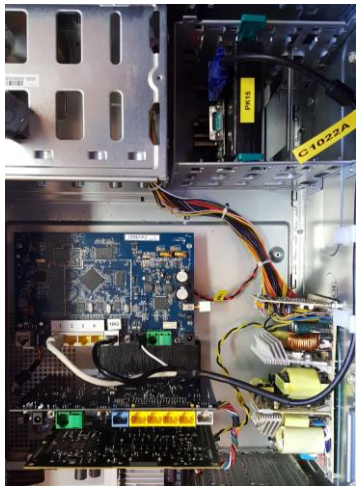
Figur 3. 3D-rendering av vår design för apparatskåpet.

7.2.2. Konstruktion av utrustning

Fundamentet i apparatskåpet bestod av ett datorchassi. För att kunna göra en detaljerad plan över skåpets inredning började vi med att tömma hela chassit. Den utrusning vi hade för att bygga upp skåpets innehåll bestod av originalkomponenterna från chassit, diverse kretskort och olika typer av kablar. Med detta utbud av utrustning kunde vi planera hur skåpet skulle inredas för att passa experimentet. Detta lät oss även se vad som saknades i form av byggmaterial och verktyg.

Det var viktigt att apparatskåpet efterliknade den utrustning som används inom industrin. Därför lade vi stor vikt i att allt som placerades i skåpet hade ett syfte. Inledningsvis byggdes de delar som var direkt involverade i experimentet. Det innefattade två specialgjorda kablar och två slädar samt ett kretskort med uttag för RJ45- och RJ12-kontakter. Båda kablarna hade två RJ45 och en RJ12 i ena änden och en VGA- respektive DVI-kontakt i andra. Slädarna konstruerades så att de gick att skjuta in i hårddiskracket i datorchassit. På ena släden monterade vi en serieport (DE-9) och en VGA-port, på den andra släden

satte vi en VGA-port och en DVI-port. Vi anpassade kablarnas längd så de passade avståndet mellan kretskortet och slädarna och samtidigt kunde fästas i en kabelhållare som fanns i datorchassit.



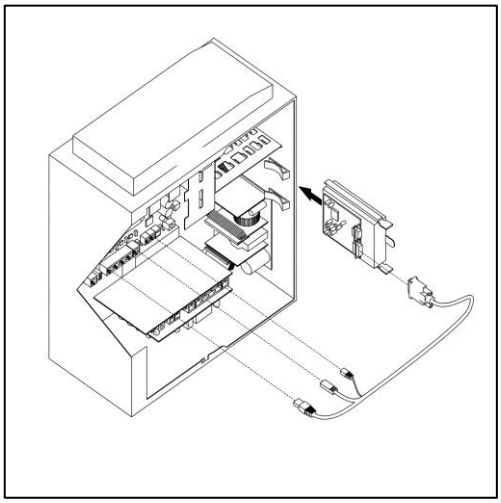
Figur 4. Det färdiga apparatskåpet.

När de viktigaste delarna var färdigställda fortsatte vi med att montera delar som dels skulle få apparatskåpet att se mer komplett ut och dels kunna förväxlas med de tidigare monterade delarna. Skälet till detta var att inte göra det för uppenbart för testdeltagarna var kablar skulle sättas. De kompletterande delarna innehöll dels samma sorts uttag som det kretskort som deltagarna skulle utföra uppgifter på och dels kablar som vi valde att dra till olika delar av skåpet. Det färdiga apparatskåpet visas i Figur 4.

7.2.3. Konstruktion av pappersmanual

För att kunna jämföra AR-instruktioner med pappersinstruktioner så skapade vi en pappersmanual som efterliknar den referensmanual [29] som användes för att utforma underhållsuppgifterna. För att skapa en manual modellerades all vår utrustning i 3D-modelleringsprogrammet Blender. Modellerna renderades sedan på ett sätt som efterliknade bilderna i referensmanualen, Figur 5 visar en av dessa renderingar, den kompletta manualen redovisas i Bilaga A.

För att kunna beskriva de olika uppgifterna manualen innehöll namngav vi alla komponenter. Denna information fanns i manualen så att användaren skulle kunna identifiera de komponenter manualen refererade till. Uppgiften som skulle utföras var uppdelad i nio moment. Respektive moment hade en procedurbeskrivning som innehöll en eller flera uppgifter. Manualen bestod av en sida med bilder och komponentförklaring och en sida med alla moment som skulle genomföras.



Figur 5. Rendering som använts i pappersmanual.

För att säkerställa att manualen motsvarade en teknisk manual som används inom industrin så har vi, som tidigare beskrivits, använt en manual från ABB som inspiration. Utöver detta granskades den färdiga manualen av en person med mångårig erfarenhet av teknisk dokumentation och instruktioner inom industrin, och justerades efter denna persons återkoppling. Den justerade manualen kontrollerades sedan av ytterligare en person med erfarenhet av industriellt underhåll och teknisk dokumentation.

7.2.4. Konstruktion av AR-applikation

Utvecklingsverktyget Vuforia som vi använde för AR-applikationen erbjuder både spårning med markörer och markörlös spårning. Vid markörspårning krävs högkvalitetsmarkörer, dessa ska främst ha tydliga kontraster och väldefinierade former för att AR-kameran från Vuforia ska kunna identifiera dem [30]. Vi valde att använda markörspårning för vår AR-applikation, vilket innebar att vi inledde med att skriva ett program för att generera markörer av hög kvalitet. Detta program, vars kod återfinns i Bilaga B, är skrivet i Java för utvecklingsmiljön Processing [31]. Programmet har använts för att generera samtliga markörer åt AR-applikationen, ett exempel på en genererad markör ges i Figur 6.

Figur 6. Markör genererad av vårt program.

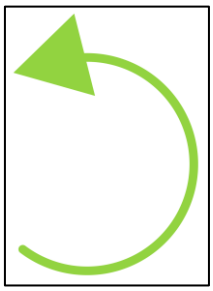
Vid markörspårning kan man ytterligare göra valet om man vill ha statisk spårning eller utökad spårning i Vuforia. Statisk spårning innebär att de 3D-modeller som är associerade med markören bara syns om markören är i kamerans synfält. Utökad spårning innebär att Vuforia kartlägger området runt markören i realtid för att skapa en referenskarta som den kan spåra på, detta innebär i sin tur att markören inte behöver vara i kamerans synfält för att 3D-modeller ska genereras. Vi valde att använda utökad spårning i huvudsak och statisk spårning för detaljrika instruktioner, då utökad spårning var markant sämre när små detaljrika instruktioner skulle visas.

Spårningsproblematiken ledde till att vi använde flera markörer: en stor markör med utökad spårning centralt i apparatskåpet och två mindre markörer med statisk spårning för att visa detaljrika instruktioner på komponentslädarna, se Figur 7. Utifrån inledande experiment med att testa storlek på markörer i relation till deras spårningsegenskaper kom vi fram till storleken 8x8cm och 2,5x2,5cm för den stora respektive de små markörerna. Storleksvalet var en avvägning mellan det avstånd som markören känns igen från och vad som fick plats i apparatskåpet.

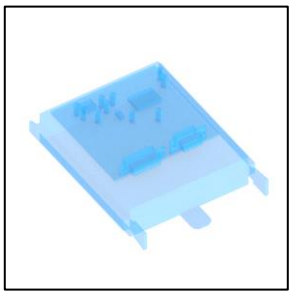
Figur 7. Stor markör centralt i apparatskåpet (vänster) och liten markör på släde (höger).

För att visa instruktioner använde vi oss av både 2D- och 3D-modeller av pilar och 3D-modeller av komponenter och kablar. De olika varianter av pilar som användes och ett exempel på 3D-modell illustreras i Figur 8 till Figur 11. Modellerna blandades i vissa instruktioner, till exempel användes modellerna i Figur 8 och Figur 10 samtidigt då en kontakt skulle skruvas loss och sedan tas bort, se Figur 12.

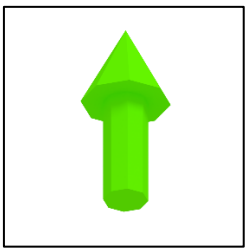
Kärnlogiken i applikationen sköttes av ett överliggande skript som styrde vilken instruktion som skulle visas. Varje instruktion låg som ett eget objekt med associerade pilar och modeller. Navigationen mellan de olika instruktionerna styrdes med en framåt- och en bakåtknapp, skriptet bestämde utifrån vilket steg användaren navigerat till vilka objekt som skulle renderas. Kärnlogiken hade två specialfall, ett för när luckan på apparatskåpet var monterad så centralmarkören inte syntes, och ett för när modeller skulle renderas från de små slädesmarkörerna. När luckan var monterad fanns det en extra markör för att kunna visa att luckan skulle tas av. För att hantera de små markörerna krävdes det en mer avancerad lösning. Instruktionselementens position var alltid i relation till en markör, vilket innebar att när man gick från centralmarkören till en slädmarkör ändrades kamerans centreringspunkt och vad som renderades. Det andra specialfallet hanterade då man gick från centralmarkören till en slädmarkör och sedan tillbaka. I dessa fall stängde vi av renderingen från centralmarkören när en slädmarkör registrerades av AR-kameran, och instruktioner visades med slädmarkören som centreringspunkt. När slädmarkören inte längre registrerades stängdes dess rendering av och centralmarkören blev centreringspunkt igen.



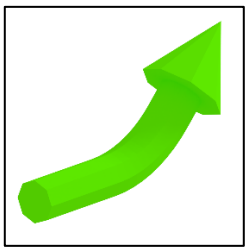
Figur 8. 2D-pil som visade när något skulle skruvas.



Figur 9. 3D-modell av komponentsläde som användes i instruktionerna.



Figur 10. 3D-pil som visade riktning.



Figur 11. 3D-pil som visade när något skulle dras i en riktning och sedan en annan.

Figur 12. Kombination av 2D- och 3D-instruktioner.

7.2.5. Testning av AR-applikation

För att säkerställa att vår applikation hade tillfredställande kvalitet och att instruktionerna givna med applikationen var enkla att följa genomförde vi ett test. En testdeltagare, som inte haft insyn i vårt arbete eller varit delaktig i utvecklingen av vår applikation, fick genomföra experimentuppgiften med hjälp av AR-applikationen. Vi övervakade testet och tog anteckningar om vad deltagaren sa och gjorde.

Innan testet fick deltagaren information om vad uppgiften var. Vidare informerade vi om att vi inte kommer blanda oss i vad personen gjorde under tiden testet pågick och att vi inte kommer svara på frågor under testet. För oss var det viktigt att se om instruktionerna i applikationen var tillräckliga för att genomföra uppgiften.

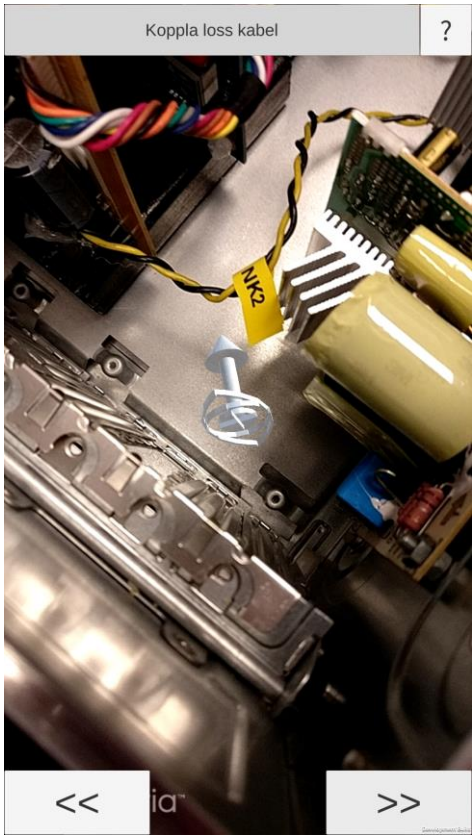
Resultatet av testet var att flera instruktioner var undermåliga, testdeltagaren förstod inte vissa moment och hoppade över steg eller chansade på vad som menades. Det visade sig också att de animerade objekt vi hade för att visa att man som användare skulle gå närmare med telefonen inte uppfattades korrekt av testpersonen. Personen bortsåg helt från den symbolen, vilket ledde till att vissa steg inte presenterades överhuvudtaget. Vi uppmärksammade även under testets gång att ljusförhållandena spelade stor roll när det kom till spårning, detta orsakade att instruktioner blev felplacerade när en markör var för mörk.

7.2.6. Förbättring av AR-applikation

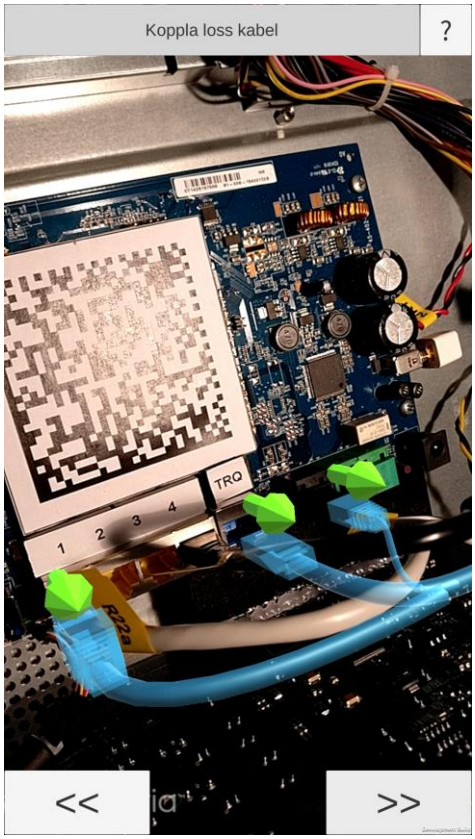
Efter att ha gått igenom våra anteckningar från testet kom vi fram till ett antal förbättringar. För att en användare skulle hitta var ett moment skulle utföras lade vi till en riktningspil som pekade mot delen i apparatskåpet där arbetet i ett givet steg skulle utföras. Vi valde att visa riktningspilen enbart när de momentbeskrivande objekten inte syntes i applikationen. Detta för att riktningspilen inte skulle blockera något en användare behövde se för att utföra en uppgift. Problemet med att användaren inte förstod exakt vad som skulle utföras löste vi med att lägga in en textruta med en kort förklaring av uppgiften i

varje moment. Detta var något vi tidigare sett i arbeten med AR-instruktioner [8, 21]. Det förbättrade gränssnittet visas i Figur 13.

Utöver de ovannämnda två större förändringarna genomförde vi vissa förändringar på hur vi presenterade en del steg. Bland annat lade vi till animerade objekt i fler steg för att tydligare visa vad som skulle utföras, exempel på ett sådant objekt ges i Figur 14. Vi gjorde det också möjligt att visa en hjälpsida under tiden man använde applikationen. Hjälpsidan innehöll förklaringar av vad de olika objekten och symbolerna som användes i applikationen innebar. Den slutgiltiga koden för implementationen och bilder på applikationens alla steg återfinns i Bilaga C respektive Bilaga D.



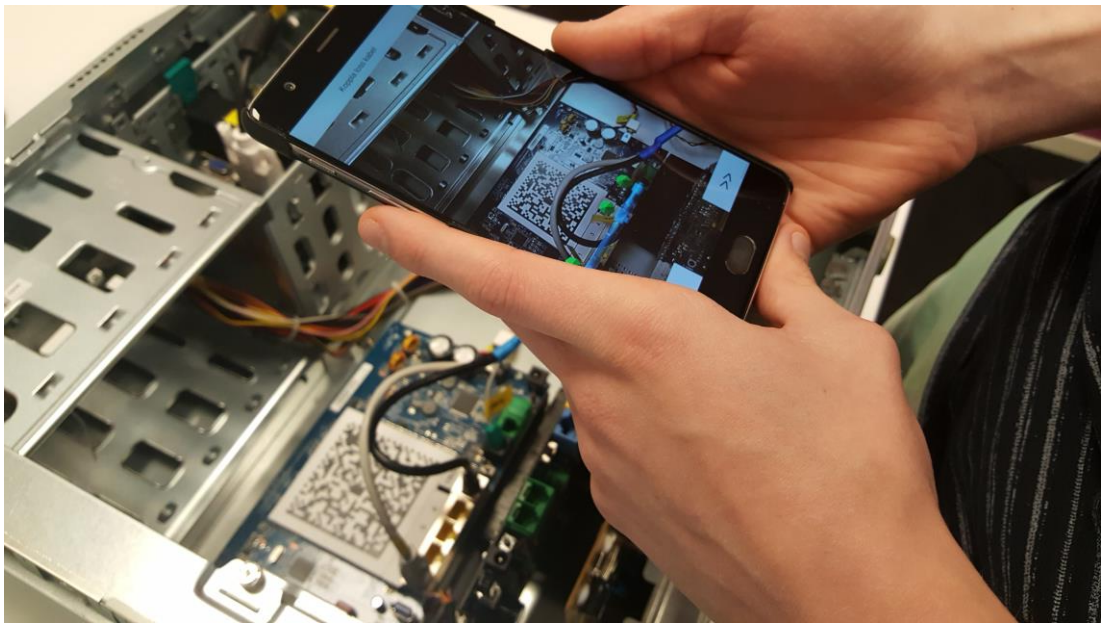
Figur 13. AR-applikationens gränssnitt med synlig riktningspil.



Figur 14. Instruktion i AR-applikationen.

7.3. Utförande av experiment

Tio personer från företaget byBrick Interface AB i Västerås deltog i experimentet som bestod i att utföra ett arbete i apparatskåpet som beskrivits i avsnitt 7.2. Hälften av deltagarna fick instruktioner via AR och för att göra det möjligt att jämföra resultaten med pappersmanual fick den andra hälften av deltagarna instruktioner via en utskrift av den manual som återfinns i Bilaga A. AR-applikationen kördes på en mobiltelefon av modellen OnePlus 3T med operativsystemet Android 8.0. De personer som utförde testet med AR fick telefonen med applikationen startad och använde den för att få instruktioner, se Figur 15. Innan experimentet inleddes fick de deltagare som fick instruktioner via AR en kort introduktion till vad AR är och hur det fungerar. För att kunna utvärdera och analysera resultaten valde vi att spela in samtliga deltagare på film. Under arbetet med uppgiften mätte vi tidsåtgången och antecknade vad personen sade och gjorde.



Figur 15. AR-applikationen används för instruktioner.

För att minimera yttre påverkan fick alla deltagare arbeta enskilt i ett avskilt rum. Vi fanns med i rummet och observerade men hade förklarat för deltagarna att vi inte skulle lägga oss i arbetet eller svara på frågor under tiden experimentet pågick. Efter att uppgiften var utförd fick deltagarna svara på frågor om uppgiftens komplexitet och hur de uppfattat instruktionerna. De deltagare som använt sig av AR fick även fylla i SUS-formuläret.

AR-testpersonerna fick svara på följande frågor efter att uppgiften var slutförd:

- På en skala från ett till tio, där ett är enkel och tio är avancerad, var skulle du sätta uppgiften du nyss utförde?
- Hur upplevde du att få instruktioner via en mobiltelefon?
- Beskriv hur du upplevde instruktioner via AR.

De som utförde uppgiften med pappersinstruktioner fick svara på följande frågor:

- På en skala från ett till tio, där ett är enkel och tio är avancerad, var skulle du sätta uppgiften du nyss utförde?
- Hur upplevde du instruktionerna?

8. Resultat

Tabell 2 och Tabell 3 visar antalet fel och den tid det tog för testpersonerna att slutföra underhållsuppgiften. I genomsnitt utförde AR-testpersonerna uppgiften på 7 minuter och 43 sekunder med en median på 7 minuter och 59 sekunder, medan de med pappersmanual i genomsnitt klarade uppgiften 26 sekunder snabbare med en median på 7 minuter och 24 sekunder. Till antalet fel sett gjorde AR-testpersonerna i genomsnitt ett fel, i jämförelse med 2,6 fel för de som använde sig av pappersmanual. Maxtiden och minimitiden för AR var 11:36 respektive 04:30, för pappersmanualen var de 11:15 respektive 04:39.

AR-INSTRUKTIONER		
	Tid (mm:ss)	Fel
PERSON 1	08:13	0
PERSON 2	06:17	2
PERSON 3	04:30	0
PERSON 4	11:36	3
PERSON 5	07:59	0

Tabell 2. Tider och antal fel med AR-instruktioner.

PAPPERSMANUAL		
	Tid (mm:ss)	Fel
PERSON 6	11:15	5
PERSON 7	05:02	0
PERSON 8	07:24	3
PERSON 9	08:04	0
PERSON 10	04:39	5

Tabell 3. Tider och antal fel med pappersmanual.

Vår definition av fel var när användaren gjorde ett moment i fel ordning, missade att utföra ett moment eller utförde momentet felaktigt, till exempel ansluta en kontakt i fel uttag. Oavsett om slutresultatet blev rätt, så har vi registrerat alla fel deltagarna gjorde under testet. De testpersoner som använde AR-instruktioner åtgärdade totalt 60% av felen de gjorde, medan testpersonerna med pappersmanual åtgärdade totalt 15,4% av felen.

På frågan om uppgiftens komplexitet var det genomsnittliga resultatet 3,35 på en skala där ett var enkel och tio avancerad. Individuella svar på denna fråga tillsammans med användarnas svar på hur de upplevde AR- respektive pappersinstruktionerna återfinns i Bilaga E.

Det sammanlagda resultatet av svaren på SUS-utvärderingen var 69 poäng. De individuella värdena för applikationens SUS-poäng presenteras i Tabell 4**.**

FRÅGA	PERSON 1	PERSON 2	PERSON 3	PERSON 4	PERSON 5
1	5	3	4	4	5
2	4	2	2	3	2
3	4	3	4	2	4
4	1	2	1	2	1
5	3	4	4	3	3
6	5	4	1	2	2
7	5	5	3	5	4
8	3	3	1	4	2
9	5	2	4	1	5
10	2	2	1	3	1
SUS-POÄNG	67,5	60	82,5	52,5	82,5

Tabell 4. SUS-poäng för AR-applikationen.

8.1. Observationer

För de testpersonerna som använde AR-applikationen märkte vi att de hade väldigt svårt med att få applikationen att registerna slädesmarkörerna. Detta resulterade i sin tur i att det tog längre tid för dem att få instruktionerna, då applikationen inte kan visa instruktionerna om markören inte har registrerats.

Generellt var det största problemet med AR-applikationen att användarna inte kom tillräckligt nära markörerna eller att markörer var skymda av objekt i apparatskåpet, vilket i sin tur påverkade spårningen och positioneringen av de virtuella objekten negativt. Vi observerade att alla användare av applikationen använde sig av riktningspilen för att hitta till nästa instruktion.

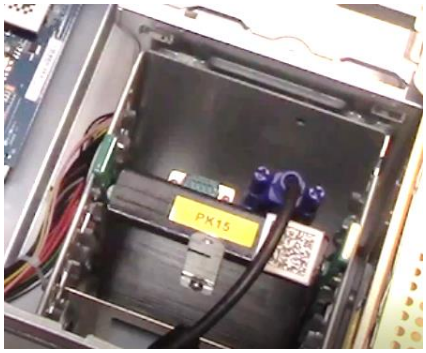
Vi kunde se att de som använde pappersmanualen arbetade mer på eget initiativ i jämförelse med de som använde AR-applikationen. Detta resulterade i att de gjorde fler saker i fel ordning vilket också återspeglas i antalet registrerade fel. Ett tydligt exempel på detta var en testdeltagare som genomförde uppgiften fram till steg sju i manualen enbart genom att studera sprängskisserna på manualens första sida. Detta exempel gav ett felaktigt slutresultat som aldrig åtgärdades även när testpersonen började läsa stegen i manualen.

9. Diskussion

Vi ville undersöka om antalet fel minskar när användarna utför en uppgift med AR-instruktioner jämfört med en pappersmanual. Som vi kan se i resultatet så gjorde testdeltagarna märkbart färre fel när de använde sig av AR vilket går i linje med tidigare resultat från Sanna et al. [8]. Som några av testdeltagarna påpekade så tror vi att de färre antalen fel testdeltagarna gör beror på att AR-instruktioner är mer visuella och kan visa exakt vart en uppgift ska utföras. Att det visuella egenskaperna hos AR är en fördel är även något Aromaa et al. upplevde [19].

Skälet till att bygga en unik underhållsuppgift var för att försäkra oss om att ingen av testdeltagarna skulle ha tidigare erfarenhet av uppgiften. Alla komponenter i apparatskåpet kom från befintlig teknik, så möjligheten finns att vissa testdeltagare kände igen delar av materialet i uppgiften, men vi känner oss säkra på att uppgiften i sin helhet var unik. Begränsningen med att ha en unik uppgift är att vi inte kan dra några slutsatser om huruvida AR-instruktioner hade fått samma resultat för en befintlig underhållsuppgift med en erfaren underhållsoperatör. Under förstudien hos Volvo Powertrain AB nämnde de att AR-instruktioner kunde vara fördelaktigt för att lära upp nyanställda eller inhyrd personal från bemanningsföretag, något som även Quint et al. [20] tar upp. Vi tror att vårat experiment återspeglar ett scenario som kan uppstå när en oerfaren operatör i dagsläget ställs inför en helt ny underhållsuppgift.

Tiden det tog att slutföra uppgiften med AR-applikationen jämfört med pappersinstruktioner skiljer sig inte markant. Vi tror att tiden för AR påverkades negativt av svårigheterna med att hitta de små markörerna. Dels på grund av deras storlek som kräver att användaren kommer väldigt nära med kameran och dels för att de i vissa fall var skymda av en kabel såsom i Figur 16. Precis som problemet Havard et al. [9] upplevde så missade vissa testdeltagare uppgifter, vilket skulle kunna lösas om användaren bättre förstår vad den ska leta efter. Vår lösning med riktningspilen som visar vart uppgiften är fungerar för detta ändamål, men skulle kunna förbättras med att också visa vad användaren ska leta efter.



Figur 16. Markör skymd av kabel.

Deltagarna i vårt användartest hade varierande grad av erfarenhet av materialet i apparatskåpet, trots detta kan vi se från data i Bilaga E att nio av tio deltagare gav uppgiften 3–3,5 poäng på en tiogradig komplexitetsskala, där tio var avancerad. Vi ser då medelbetyget på 3,35 poäng som bekräftelse på att uppgiften hade en korrekt avvägd komplexitetsgrad. Utifrån detta uppfattar vi att resultatet från experimentet ger en rättvis jämförelse av AR- och pappersinstruktioner, då testdeltagarna inte var hindrade av uppgiftens komplexitet, men uppgiften var inte enkel nog att slutföras korrekt utan instruktioner.

Vi ville även se om vår applikation kunde uppnå en poäng på 70 eller över på SUS-skalan. Resultatet på 69 poäng når inte riktigt upp till målet på 70 poäng. Vi anser att 69 poäng är tillräckligt för att bekräfta applikationens användarbarhet då det fortfarande är inom den 50:e percentilen [18]. I sin nuvarande form har AR-tekniken på mobiltelefoner brister, det krävs god ljussättning och stora tydliga markörer

för att spårningen ska fungera optimalt. Vi observerade vid flera tillfällen att AR-applikationen tappade spårningen vilket störde testpersonerna.

Eftersom det inte finns ett etablerat designspråk för AR-instruktioner än så är det en utmaning att utveckla användarvänliga gränssnitt. På grund av den begränsade utvecklingstiden så genomförde vi enbart ett test av vår applikation och förbättrade den utifrån det. Mer tid hade tillåtit fler iterationer av vårt gränssnittsarbete och applikationen hade möjligen då också fått högre SUS-poäng. Så vitt vi känner till är lösningen med en riktningspil för att visa vart nästa instruktion är ny inom området AR-instruktioner. Vi kunde observera att användarna drog nytta av den, men som vi tidigare nämnt finns det möjlighet för förbättringar.

10. Slutsats

Vi har i detta arbete jämfört AR-instruktioner med pappersinstruktioner inom industriellt underhåll. Syftet med vårt arbete var att undersöka om användare av AR-instruktioner gör färre fel jämfört med användare av pappersinstruktioner på en uppgift inom industriellt underhåll. Vi genomförde ett experiment på en underhållsuppgift med tio deltagare där hälften fick AR-instruktioner och hälften fick en pappersmanual. Resultatet från experimentet visar att de personer som använde pappersinstruktioner i genomsnitt gjorde 2,6 gånger fler fel än de som använde AR-instruktioner. Resultaten visar även att det inte fanns någon signifikant skillnad på tidsåtgången för respektive instruktionsmedium. Vidare ville vi undersöka om AR-tekniken på mobiltelefoner är tillräckligt för att uppnå 70 poäng på SUS-skalan, vår implementation fick ett resultat på 69 poäng.

11. Framtida arbete

Vi tror att mycket arbete fortfarande kvarstår i utvecklingen av gränssnitt för AR-applikationer. Vi föreslår ytterligare forskning i att översätta instruktioner som normalt sett skrivs med text till objekt och symboler utan att förlora förståelse för uppgiften. Vi ser också att mer arbete krävs med utvecklingen av vår riktningspil med målet att tydligare visa var instruktioner ska utföras.

12. Referenser

[1] A. Javornik, "The Mainstreaming of Augmented Reality: A Brief History," *Harvard Business Review*, 4 oktober 2016. [Online], Tillgänglig: https://hbr.org/2016/10/the-mainstreaming-of-augmented-reality-a-brief-history. [Använd 14 februari 2018].

[2] J. Paavilainen, H. Korhonen, K. Alha, J. Stenros, E. Koskinen, and F. Mayra, "The Pokémon GO experience: A location-based augmented reality mobile game goes mainstream," i *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 2017, pp. 2493-2498.

[3] Microsoft Corporation, "Microsoft HoloLens," *Microsoft Corporation*, 2018. [Online]. Tillgänglig: https://www.microsoft.com/en-us/hololens. [Använd 14 februari 2018].

[4] J. Edgren, "Hägglunds nya stridsfordon slåss med augmented reality," *Ny Teknik*, 31 januari 2018. [Online], Tillgänglig: https://www.nyteknik.se/fordon/hagglunds-nya-stridsfordon-slass-med-augmented-reality-6896274. [Använd 6 mars 2018].

[5] Google Developers, "ARCore - Google Developer," *Google LLC*. [Online]. Tillgänglig: https://developers.google.com/ar/. [Använd 22 februari 2018].

[6] Apple Inc., "ARKit - Apple Developer," *Apple Inc.*, 2018. [Online]. Tillgänglig: https://developer.apple.com/arkit/. [Använd 22 februari 2018].

[7] PTC Inc., "Vuforia | Augmented Reality," *PTC Inc.*, 2018. [Online]. Tillgänglig: https://www.vuforia.com/. [Använd 22 februari 2018].

[8] A. Sanna, F. Manuri, F. Lamberti, G. Paravati, and P. Pezzolla, "Using handheld devices to support augmented reality-based maintenance and assembly tasks," i *International Conference on Consumer Electronics (ICCE)*, 2015, pp. 178-179.

[9] V. Havard, D. Baudry, X. Savatier, B. Jeanne, A. Louis, and B. Mazari, "Augmented Industrial Maintenance (AIM): A Case Study for Evaluating and Comparing with Paper and Video Media Supports," i *AVR 2016: Augmented Reality, Virtual Reality, and Computer Graphics*, 2016, pp. 302-320.

[10] R. T. Azuma, "A survey of augmented reality," *Presence: Teleoperators & Virtual Environments*, vol. 6, nr. 4, pp. 355-385, 1997.

[11] B. M. Khuong, K. Kiyokawa, A. Miller, J. J. La Viola, T. Mashita, and H. Takemura, "The effectiveness of an AR-based context-aware assembly support system in object assembly," i *Virtual Reality (VR)*, 2014, pp. 57-62.

[12] J. Sääski, T. Salonen, M. Liinasuo, J. Pakkanen, M. Vanhatalo, and A. Riitahuhta, "Augmented reality efficiency in manufacturing industry: a case study," i *DS 50: Proceedings of NordDesign 2008 Conference*, 2008, pp. 99-109.

[13] S. You, U. Neumann, and R. Azuma, "Hybrid inertial and vision tracking for augmented reality registration," i *Proceedings IEEE Virtual Reality*, 1999, pp. 260-267.

[14] H. Kato and M. Billinghurst, "Marker tracking and hmd calibration for a video-based augmented reality conferencing system," i *Augmented Reality, 1999. (IWAR'99) Proceedings. 2nd IEEE and ACM International Workshop on*, 1999, pp. 85-94.

[15] A. I. Comport, É. Marchand, and F. Chaumette, "A real-time tracker for markerless augmented reality," i *Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2003, p. 36.

[16] Unity Technologies, "Vuforia SDK overview," *Unity Technologies*. Tillgänglig: https://docs.unity3d.com/Manual/vuforia-sdk-overview.html. [Använd 27 mars 2018].

[17] J. Brooke, "SUS-A quick and dirty usability scale," *Usability evaluation in industry*, vol. 189, nr. 194, pp. 4-7, 1996.

[18] J. Brooke, "SUS: a retrospective," *Journal of usability studies*, vol. 8, nr. 2, pp. 29-40, 2013.

[19] S. Aromaa, I. Aaltonen, E. Kaasinen, J. Elo, and I. Parkkinen, "Use of wearable and augmented reality technologies in industrial maintenance work," i *Proceedings of the 20th International Academic Mindtrek Conference*, 2016, pp. 235-242.

[20] F. Quint, F. Loch, and P. Bertram, "The Challenge of Introducing AR in Industry-Results of a Participative Process Involving Maintenance Engineers," *Procedia Manufacturing*, vol. 11, pp. 1319-1323, 2017.

[21] G. M. Re and M. Bordegoni, "An Augmented Reality Framework for Supporting and Monitoring Operators during Maintenance Tasks," i *VAMR 2014: Virtual, Augmented and Mixed Reality. Applications of Virtual and Augmented Reality*, 2014, pp. 443-454.

[22] I. Porcelli, M. Rapaccini, D. B. Espíndola, and C. E. Pereira, "Technical and organizational issues about the introduction of augmented reality in maintenance and technical assistance services," *IFAC Proceedings Volumes*, vol. 46, nr. 7, pp. 257-262, 2013.

[23] N. Zenati, S. Benbelkacem, M. Belhocine, and A. Bellarbi, "A new AR interaction for collaborative E-maintenance system," *IFAC Proceedings Volumes*, vol. 46, nr. 9, pp. 619-624, 2013.

[24] N. Zenati-Henda, A. Bellarbi, S. Benbelkacem, and M. Belhocine, "Augmented reality system based on hand gestures for remote maintenance," i *Multimedia Computing and Systems (ICMCS), 2014 International Conference on*, 2014, pp. 5-8.

[25] J. Wang, Y. Feng, C. Zeng, and S. Li, "An augmented reality based system for remote collaborative maintenance instruction of complex products," i *Automation Science and Engineering (CASE), 2014 IEEE International Conference on*, 2014, pp. 309-314.

[26] R. Masoni *et al.*, "Supporting Remote Maintenance in Industry 4.0 through Augmented Reality," *Procedia Manufacturing*, vol. 11, pp. 1296-1302, 2017.

[27] D. Mourtzis, V. Zogopoulos, and E. Vlachou, "Augmented Reality Application to Support Remote Maintenance as a Service in the Robotics Industry," *Procedia CIRP*, vol. 63, pp. 46-51, 2017.

[28] Blender Foundation, "blender.org - Home of the Blender project - Free and Open 3D Creation Software," *Blender Foundation*. Tillgänglig: https://www.blender.org/. [Använd 20 april 2018].

[29] ABB Automation Technologies AB, "Installation, service och underhållsinstruktion för lågspånningsbrytare typ ACB," *ABB Automation Technologies AB*, [Online], Tillgänglig: http://new.abb.com/low-voltage/sv/support/genvagar-for-konsult-och-konstruktor/manualer. [Använd 16 april 2018].

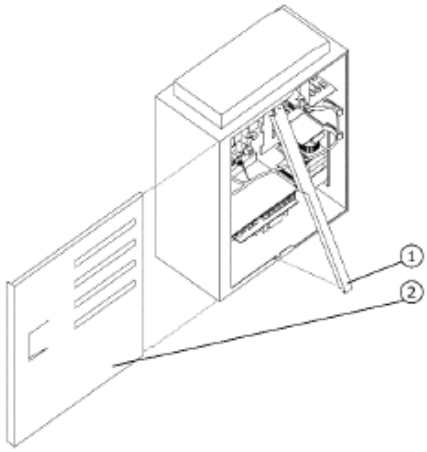
[30] PTC Inc., "Optimizing Target Detection and Tracking Stability," *PTC Inc.*, 2018. Tillgänglig: https://library.vuforia.com/articles/Solution/Optimizing-Target-Detection-and-Tracking-Stability.html#optimize. [Använd 4 april 2018].

[31] Processing Foundation, "Processing.org," *Processing Foundation*. Tillgänglig: https://processing.org/. [Använd 8 maj 2018].

Bilaga A. Pappersmanual

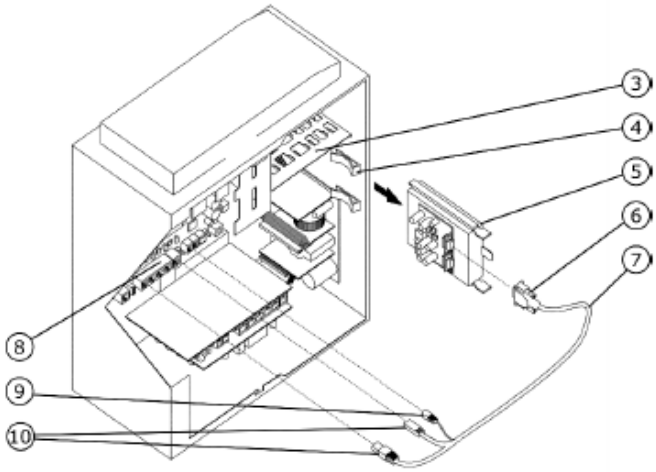
Utbyte av PK-enhet

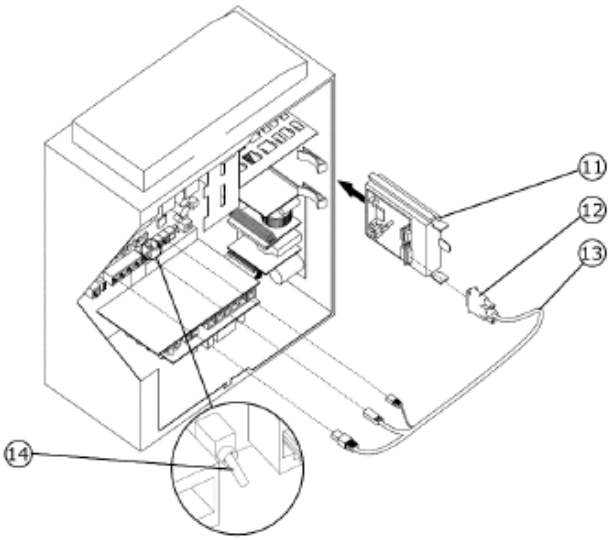
Instruktion för utbyte av modul PK15 till PK75 i apparatskåp STS 2070/B.



Förklaring

1. Stödpelare
2. Servicelucka
3. PK-hållare
4. Kabelhållare
5. Modul PK-15
6. PK/A-kontakt
7. Kabel C1022A
8. Kopplingsplint för PK-moduler
9. R18-kontakt
10. R22-kontakt
11. Modul PK-75
12. PK/D-kontakt
13. Kabel C1022D
14. Vippomkopplare





	Moment	Procedur
1	Ta bort servicelucka	- Skjut serviceluckan (2) åt vänster.
2	Ta bort stödpelaren	- Dra nedre del av stödpelaren (1) rakt utåt från skåpet. - Dra ut stödpelaren nedåt.
3	Koppla bort kabel C1022A	- Lossa respektive R-kontakt (9,10) genom att trycka in sprint och dra ut. - Lossa båda skruvarna från PK/A-kontakt (6). - Dra ur kontakten ur PK-modul. - Lossa kabel från kabelhållare (4).
4	Ta bort PK-modul	- Dra ut modul PK-15 (5) rakt ut i handtaget.
5	Montera ny PK-modul	- Skjut in modul PK-75 (11) i PK-hållare (3).
6	Anslut kabel C1022D	- Anslut R22a (10) i kopplingsplint, anslutning 1. (8) - Anslut R22b (10) i kopplingsplint, anslutning TRQ. - Anslut R18D (9) i kopplingsplint enligt färgkod. - Anslut kontakt PK/D (6) i PK-modul. - Dra åt båda kontaktskruvarna för att fästa kontakten. - Fäst kabel i kabelhållare (4).
7	Koppla om läge på kopplingsplint.	- Ställ vippomkopplare (14) i D-läge (nedåt).
8	Montera stödpelare	- Skjut i stödpelaren i spåret på den övre sidan av apparatskåpet. - Fäll stödpelaren mot apparatskåpet och fäst den i nedre monteringsspår.
9	Montera servicelucka	- Placera luckan mot apparatskåpet. - Skjut luckan åt höger för att fästa.

Bilaga B. Programkod för markörgenerering

```
/*
/  QR Code Generator 1.0
/  Generates QR code images that yields five star quality rating,
/  according to Vuforia, for use in AR applications.
*/
final int squareCount = 30; // Number of rows and columns of squares
final int imageSize = 1000; // Image ratio is always 1:1 (square)
final int margin = 100;     // White margin around the marker
PGraphics img;              // Output image

final color black = color(0);
final color white = color(255);

void setup() {
  img = createGraphics(imageSize+margin*2, imageSize+margin*2, JAVA2D);
  img.beginDraw();
  printSquares(squareCount);
  img.endDraw();
  img.save(year() + "" + month() + "" + day() + "" + hour() + "" + minute()
+ "" + second() + ".jpg");
  println("Done!");
}

void printSquares(int count) {
  int rectSize = imageSize / count;
  img.noStroke();
  img.background(255);
  for(int i = 0; i < count; i++) {
    for(int j = 0; j < count; j++) {
      img.fill(squareColor(i,j,count));
      img.rect(i*rectSize+margin, j*rectSize+margin, rectSize, rectSize);
    }
  }
}

color squareColor(int i, int j, int count)
{
  // Default Color is Black
  color fillColor = black;

  // Randomly set to White
  if(random(1) < 0.5)
    fillColor = white;

  // Top row every other
  if(j == 0)
    if(i % 2 == 0)
      fillColor = black;
    else
      fillColor = white;

  // Right column every other
  if(i == count-1)
    if(j % 2 == 1)
      fillColor = black;
```

```
    else
      fillColor = white;

  // Left column black
  if(i == 0)
    fillColor = black;

  // Bottom row black
  if(j == count-1)
    fillColor = black;

  return fillColor;
}
```

Bilaga C. Programkod för AR-applikation

Animate.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public enum Axis {
    X,
    NegX,
    Y,
    NegY,
    Z,
    NegZ
}

public class Animate : MonoBehaviour {

    public Axis axis;
    public float distance = 1.0f;
    public int easeInFactor = 2;
    public float speed = 0.01f;

    private Vector3 origin;
    private float deltaPosition;

    /// <summary>
    /// Initialization of the instance.
    /// </summary>
    void Start () {
        origin = transform.position;
        deltaPosition = 0.0f;
    }

    /// <summary>
    /// Runs once every frame
    /// </summary>
    void Update () {
        var x = origin.x;
        var y = origin.y;
        var z = origin.z;

        float moveDistance = distance * Mathf.Pow(deltaPosition, easeInFac-
tor);

        // Scales moving distance by object scale and add distance to cho-
sen axis
        switch(axis)
        {
            case Axis.X:
                x += moveDistance * transform.localScale.x / 100.0f;
                break;
            case Axis.Y:
                y += moveDistance * transform.localScale.y / 100.0f;
                break;
            case Axis.Z:
                z += moveDistance * transform.localScale.z / 100.0f;
```

```
                break;
            case Axis.NegX:
                x -= moveDistance * transform.localScale.x / 100.0f;
                break;
            case Axis.NegY:
                y -= moveDistance * transform.localScale.y / 100.0f;
                break;
            case Axis.NegZ:
                z -= moveDistance * transform.localScale.z / 100.0f;
                break;
        }

        transform.position = new Vector3(x, y, z);
        deltaPosition += speed;
        if (deltaPosition > 1.0f) deltaPosition = 0.0f;
    }
}
```

ButtonBehaviour.cs

```
using UnityEngine;
using UnityEngine.UI;

public class ButtonBehaviour : MonoBehaviour {

    public ViewState state;
    public bool isNext;

    private Button btn;

    /// <summary>
    /// Initialization of the instance.
    /// </summary>
    void Start () {
        btn = GetComponent<Button>();
        btn.onClick.AddListener(() => ActionClick());
    }

    /// <summary>
    /// Runs every frame
    /// </summary>
    void Update () {
        ToggleInteractable();
    }

    /// <summary>
    /// Change state
    /// </summary>
    void ActionClick()
    {
        if (isNext)
            state.Next();
        else
            state.Previous();
    }

    /// <summary>
    /// Toggle whether button should be interactable or not.
    /// </summary>
```

```
    void ToggleInteractable()
    {
        if (isNext)
            if(state.AtLastState())
                btn.interactable = false;
            else
                btn.interactable = true;
        else
            if (state.AtFirstState())
                btn.interactable = false;
            else
                btn.interactable = true;
    }
}
```

Caption.cs

```
using UnityEngine;
using UnityEngine.UI;


public class Caption : MonoBehaviour {

    public Canvas canvas;
    public string captionText;
    public float rightMargin = 160;
    public float height = 160;
    public int minFontSize = 20;
    public int maxFontSize = 52;

    /// <summary>
    /// Initialization of the instance.
    /// </summary>
    void Start () {
        canvas = Instantiate(canvas);
        var panel = canvas.transform.GetChild(0);

        var panelTransform = panel.GetComponent<RectTransform>();

        var width = canvas.GetComponent<RectTransform>().rect.width -
rightMargin;
        panelTransform.sizeDelta = new Vector2(width, height);

        var textField = panel.GetChild(0);
        textField.GetComponent<Text>().text = captionText;
        textField.GetComponent<Text>().resizeTextForBestFit = true;
        textField.GetComponent<Text>().resizeTextMinSize = minFontSize;
        textField.GetComponent<Text>().resizeTextMaxSize = maxFontSize;

        canvas.GetComponent<Canvas>().enabled = false;
    }

    /// <summary>
    /// Set whether to show or hide caption
    /// </summary>
    /// <param name="visible">Bool</param>
    public void SetVisibility(bool visible)
    {
        canvas.GetComponent<Canvas>().enabled = visible;
```

```
    }

}
```

DirectionGuide.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class DirectionGuide : MonoBehaviour {

    private Transform target;

    /// <summary>
    /// Initialization of the instance.
    /// </summary>
    void Start () {

    }

    /// <summary>
    /// Runs once every frame
    /// </summary>
    void Update () {
        if (target == null)
        {
            Hide();
        }
        else
        {
            transform.LookAt(target);
        }
    }

    /// <summary>
    /// Show direction guide.
    /// </summary>
    public void Show()
    {
        GetComponent<Renderer>().enabled = true;

        foreach (var r in GetComponentsInChildren<Renderer>())
        {
            r.enabled = true;
        }
    }

    /// <summary>
    /// Hide direction guide.
    /// </summary>
    public void Hide()
    {
        GetComponent<Renderer>().enabled = false;

        foreach (var r in GetComponentsInChildren<Renderer>())
        {
            r.enabled = false;
        }
```

```
    }

    /// <summary>
    /// Set target to guide direction to.
    /// </summary>
    /// <param name="target">The target</param>
    public void SetTarget(Transform target)
    {
        this.target = target;
    }
}
```

RenderControl.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Vuforia;
using System.Linq;

public class RenderControl : MonoBehaviour {

    private StateManager sm;
    private IEnumerable<TrackableBehaviour> trackables;
    private Renderer centerCubeRend;

    /// <summary>
    /// Initialization of the instance.
    /// </summary>
    void Start ()
    {
        centerCubeRend = GameObject.Find("CenterCube").GetComponent<Ren-
derer>();
    }

    /// <summary>
    /// Runs once every frame
    /// </summary>
    void Update ()
    {
        sm = TrackerManager.Instance.GetStateManager();
        trackables = sm.GetActiveTrackableBehaviours();

        if (trackables.Any())
            centerCubeRend.enabled = true;
        else
            centerCubeRend.enabled = false;

    }
}
```

RenderSmallMarkers.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Vuforia;
using System.Linq;

public class RenderSmallMarker : MonoBehaviour {
```

```
    public ViewState viewState;
    public GameObject tracker;
    public GameObject indicator;

    private Transform marker;
    private bool visible;

    /// <summary>
    /// Initialization of the instance.
    /// </summary>
    void Start()
    {
        marker = tracker.transform;
        ToggleSprites(false);
    }

    /// <summary>
    /// Runs once every frame
    /// </summary>
    void Update()
    {
        // If step should be visible, start looking if sleigh marker is ac-
tive
        if (visible)
        {
            StateManager sm;
            IEnumerable<TrackableBehaviour> trackables;

            sm = TrackerManager.Instance.GetStateManager();
            trackables = sm.GetActiveTrackableBehaviours();

            if (trackables.FirstOrDefault(GameObject => GameObject.name ==
tracker.name))
            {
                ToggleRenderer(false);
                ToggleSprites(visible);
            }
            else
            {
                ToggleRenderer(true);
                ToggleSprites(false);
            }
        }
        else

        {
            ToggleSprites(false);
        }
    }

    /// <summary>
    /// Toggles sprite renderer on or off.
    /// </summary>
    /// <param name="b">If sprite should render or not.</param>
    private void ToggleSprites(bool b)
    {
```

```
        for (int i = 0; i < marker.childCount; i++)
        {
            if (marker.GetChild(i).childCount == 0)
                marker.GetChild(i).GetComponent<Renderer>().enabled = b;
            else
                marker.GetChild(i).GetChild(0).GetComponent<SpriteRen-
derer>().enabled = b;
        }
    }

    /// <summary>
    /// Toggles mesh renderer on or off.
    /// </summary>
    /// <param name="b">If sprite should render or not.</param>
    private void ToggleRenderer(bool render)
    {
        if (indicator == null)
            return;

        foreach (var r in indicator.GetComponentsInChildren<Renderer>())
        {
            r.enabled = render;
        }
    }

    /// <summary>
    /// Set if a child should render or not.
    /// </summary>
    /// <param name="render">If child should render.</param>
    public void Render(bool render)
    {
        visible = render;
    }
}
```

Reset.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class Reset : MonoBehaviour {

    public ViewState viewState;

    private Button btn;

    /// <summary>
    /// Initialization of the instance.
    /// </summary>
    void Start () {
        btn = GetComponent<Button>();
        btn.onClick.AddListener(ExitApp);

        Hide();
    }
```

```
    /// <summary>
    /// Runs once every frame
    /// </summary>
    void Update()
    {
        if(viewState.AtLastState())
        {
            Show();
        } else
        {
            Hide();
        }
    }

    /// <summary>
    /// Close the application
    /// </summary>
    void ExitApp()
    {
        Application.Quit();
    }

    /// <summary>
    /// Hide the GUI elements
    /// </summary>
    void Hide()
    {
        btn.GetComponent<Image>().enabled = false;
        btn.GetComponent<Button>().enabled = false;
        btn.GetComponentInChildren<Text>().enabled = false;
    }

    /// <summary>
    /// Show the GUI elements
    /// </summary>
    void Show()
    {
        btn.GetComponent<Image>().enabled = true;
        btn.GetComponent<Button>().enabled = true;
        btn.GetComponentInChildren<Text>().enabled = true;
    }
}
```

RotationAnimate.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public enum RotationDirection
{
    Right,
    Left,
    Up,
    Down,
    Forward,
    Back
}
```

```
public class RotationAnimate : MonoBehaviour {

    public RotationDirection direction;
    public float rotateDegrees;
    public float speed;
    public int easeInFactor = 2;

    private Vector3 rotationAxis;
    private float deltaDegree;
    private float rotate;

    /// <summary>
    /// Initialization of the instance.
    /// </summary>
    void Start () {
        deltaDegree = 0.0f;
        rotate = 0.0f;

        switch(direction)
        {
            case RotationDirection.Left:
                rotationAxis = Vector3.left;
                break;
            case RotationDirection.Right:
                rotationAxis = Vector3.right;
                break;
            case RotationDirection.Up:
                rotationAxis = Vector3.up;
                break;
            case RotationDirection.Down:
                rotationAxis = Vector3.down;
                break;
            case RotationDirection.Forward:
                rotationAxis = Vector3.forward;
                break;
            case RotationDirection.Back:
                rotationAxis = Vector3.back;
                break;
        }
    }

    /// <summary>
    /// Runs once every frame
    /// </summary>
    void Update () {
        transform.Rotate(rotationAxis, -rotate);
        rotate = rotateDegrees * Mathf.Pow(deltaDegree, easeInFactor);
        deltaDegree += speed;
        transform.Rotate(rotationAxis, rotate);
        if (deltaDegree > 1.0f) deltaDegree = 0.0f;
    }

}
```

SetCameraFocusMode.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
```

```
using Vuforia;

public enum FocusModes
{
    FOCUS_MODE_NORMAL,
    FOCUS_MODE_TRIGGERAUTO,
    FOCUS_MODE_CONTINUOUSAUTO,
    FOCUS_MODE_INFINITY,
    FOCUS_MODE_MACRO
}

public class SetCameraFocusMode : MonoBehaviour {

    public FocusModes focusMode;

    private bool mVuforiaStarted = false;

    /// <summary>
    /// Initialization of the instance.
    /// </summary>
    void Start () {
        var vuforia = VuforiaARController.Instance;
        vuforia.RegisterVuforiaStartedCallback(OnVuforiaStarted);
        vuforia.RegisterOnPauseCallback(OnApplicationPause);
    }

    /// <summary>
    /// Set focus mode as soon as Vuforia is loaded.
    /// </summary>
    private void OnVuforiaStarted()
    {
        mVuforiaStarted = true;
        SetFocusMode();
    }

    /// <summary>
    /// Restore focus mode after applications has been paused.
    /// </summary>
    /// <param name="pause">If applications is paused.</param>
    private void OnApplicationPause(bool pause)
    {
        if (!pause)
        {
            if (mVuforiaStarted)
            {
                SetFocusMode();
            }
        }
    }

    /// <summary>
    /// Set focus mode to selected mode.
    /// </summary>
    private void SetFocusMode()
    {
        switch (focusMode)
        {
```

```
            case FocusModes.FOCUS_MODE_NORMAL:
                CameraDevice.Instance.SetFocusMode(CameraDevice.Focus-
Mode.FOCUS_MODE_NORMAL);
                break;
            case FocusModes.FOCUS_MODE_TRIGGERAUTO:
                CameraDevice.Instance.SetFocusMode(CameraDevice.Focus-
Mode.FOCUS_MODE_TRIGGERAUTO);
                break;
            case FocusModes.FOCUS_MODE_CONTINUOUSAUTO:
                CameraDevice.Instance.SetFocusMode(CameraDevice.Focus-
Mode.FOCUS_MODE_CONTINUOUSAUTO);
                break;
            case FocusModes.FOCUS_MODE_INFINITY:
                CameraDevice.Instance.SetFocusMode(CameraDevice.Focus-
Mode.FOCUS_MODE_INFINITY);
                break;
            case FocusModes.FOCUS_MODE_MACRO:
                CameraDevice.Instance.SetFocusMode(CameraDevice.Focus-
Mode.FOCUS_MODE_MACRO);
                break;
            default:
                CameraDevice.Instance.SetFocusMode(CameraDevice.Focus-
Mode.FOCUS_MODE_NORMAL);
                break;
        }
    }
}
```

ShowDirectionGuide.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ShowDirectionGuide : MonoBehaviour
{

    public GameObject directionObject;

    private DirectionGuide directionGuide;
    private float screenWidth, screenHeight;
    private float screenFraction = 5;
    private Camera cam;

    /// <summary>
    /// Initialization of the instance.
    /// </summary>
    void Start()
    {
        directionGuide = directionObject.GetComponent<DirectionGuide>();

        cam = Camera.main;
        screenHeight = Screen.height;
        screenWidth = Screen.width;
    }

    /// <summary>
    /// Runs once every frame
    /// </summary>
```

```
    void Update()
    {
        if (GetComponent<Renderer>().enabled)
        {
            Vector3 screenPos = cam.WorldToScreenPoint(transform.position);
            directionGuide.SetTarget(transform);

            float heightMargin = screenHeight / screenFraction;
            float widthMargin = screenWidth / screenFraction;

            if (screenPos.y > heightMargin && screenPos.y < screenHeight -
heightMargin && screenPos.x > widthMargin && screenPos.x < screenWidth -
widthMargin)
            {
                directionGuide.Hide();
            }
            else
            {
                directionGuide.Show();
            }

        }

    }
}
```

SwapCanvas.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class SwapCanvas : MonoBehaviour {

    public Canvas fromCanvas, toCanvas;

    /// <summary>
    /// Initialization of the instance.
    /// </summary>
    void Start () {
        Button btn = GetComponent<Button>();
        btn.onClick.AddListener(Swap);
    }

    /// <summary>
    /// Swap which canvas to render.
    /// </summary>
    void Swap()
    {
        fromCanvas.enabled = false;
        toCanvas.enabled = true;
    }
}
```

ToggleRendering.cs

```
using System.Collections;
```

```
using System.Collections.Generic;
using UnityEngine;

public class ToggleRendering : MonoBehaviour {

    public ViewState viewState;
    public int stepIndex;

    /// <summary>
    /// Toggles the renderer for a given component
    /// </summary>
    void Update () {
        if (viewState.GetState() == stepIndex)
        {
            foreach (var r in GetComponentsInChildren<Renderer>())
            {
                r.enabled = true;
            }
        }
        else
        {
            foreach (var r in GetComponentsInChildren<Renderer>())
            {
                r.enabled = false;
            }
        }
    }
}
```

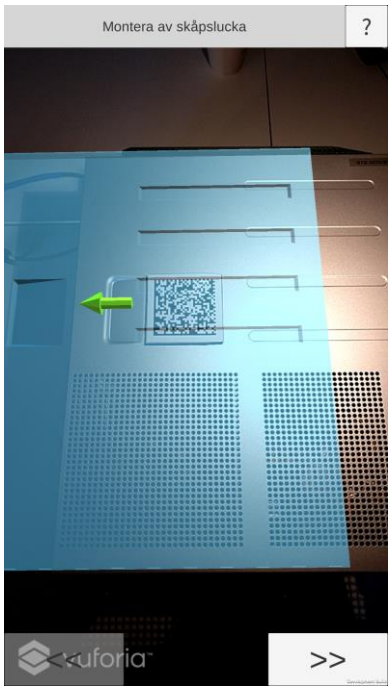
ViewState.cs

```
using System;
using System.Collections.Generic;
using UnityEngine;

public class ViewState : MonoBehaviour {

    int state;
    int stateCount;
    Transform marker;
    public GameObject tracker;
    public RenderSmallMarker renderStep3, renderStep8;
    public int sleighStepIndex, sleighStepIndexTwo;
    public GameObject directionObject;

    /// <summary>
    /// Initialization of the instance.
    /// </summary>
    void Start () {
        Initialize();
    }

    /// <summary>
    /// Registers the number of states in a given marker.
    /// </summary>
    public void Initialize()
    {
        state = 0;
```

```
        try
        {
            marker = tracker.transform;
        }
        catch (Exception e)
        {
            Debug.Log("No Tracker defined.");
            throw e;
        }
        stateCount = marker.childCount;
        Toggle();
        Debug.Log("View state starting ");
    }

    /// <summary>
    /// Go to next state.
    /// </summary>
    public void Next()
    {
        if (state >= stateCount - 1) return;
        state++;
        Toggle();
    }

    /// <summary>
    /// Go to previous state.
    /// </summary>
    public void Previous()
    {
        if (state <= 0) return;
        state--;
        Toggle();
    }

    /// <summary>
    /// Returns whether the current state the last.
    /// </summary>
    /// <returns>True if current state is last, false otherwise.</returns>
    public bool AtLastState()
    {
        return state == stateCount - 1;
    }

    /// <summary>
    /// Returns whether the current state the fist.
    /// </summary>
    /// <returns>True if current state is fist, false otherwise.</returns>
    public bool AtFirstState()
    {
        return state == 0;
    }

    /// <summary>
    /// Changes what to render based on the state.
    /// </summary>
    void Toggle()
    {
```

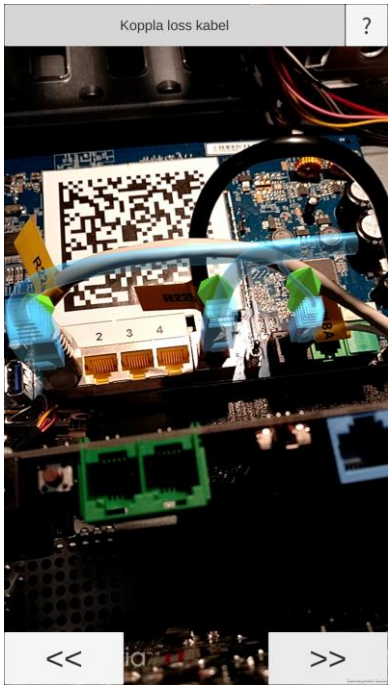
```
        directionObject.GetComponent<DirectionGuide>().Hide();
        for(var i = 0; i < stateCount; i++)
        {
            bool visible = i == state;
            var child = marker.GetChild(i);

            // Show/Hide caption for step
            Caption caption = null;
            if ((caption = child.GetComponent<Caption>()) != null)
            {
                caption.SetVisibility(visible);
            }

            // Set visibility for the respective child
            foreach (var r in child.GetComponentsInChildren<Ren-
derer>(true))
            {
                r.enabled = visible;
            }
        }

        // Special case for sled marker
        if (state == sleighStepIndex)
            renderStep3.Render(true);
        else if (state == sleighStepIndexTwo)
            renderStep8.Render(true);
        else
        {
            renderStep3.Render(false);
            renderStep8.Render(false);
        }

    }

    /// <summary>
    /// Returns the current state.
    /// </summary>
    /// <returns>The current state</returns>
    public int GetState()
    {
        return state;
    }

}
```

Bilaga D. Bilder från AR-applikationen



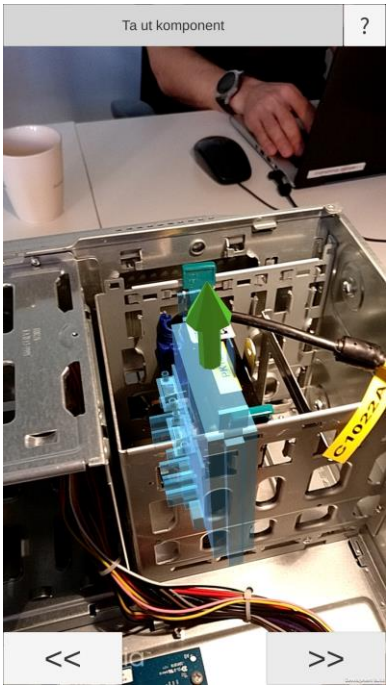
Steg 1.

Steg 2.

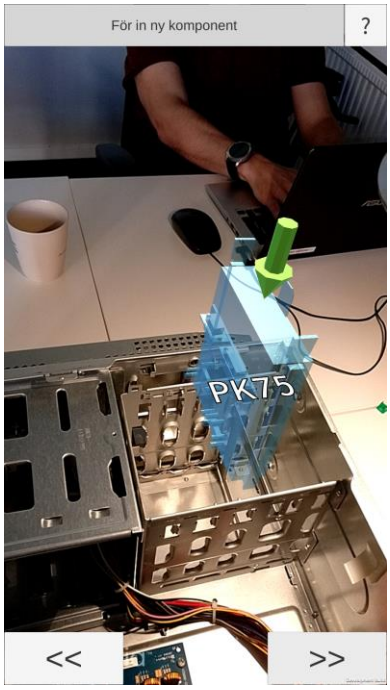


Steg 3.

Steg 4.



Steg 5.

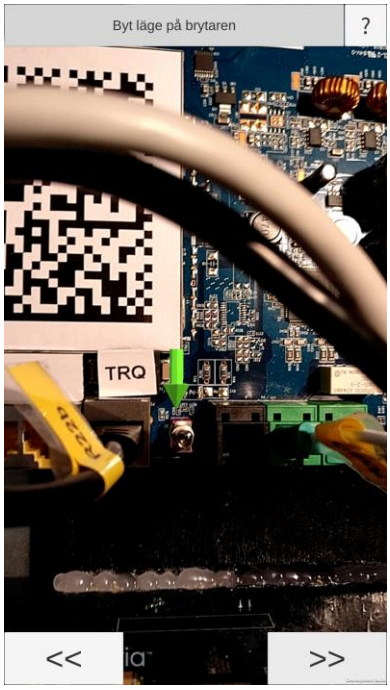


Steg 6.

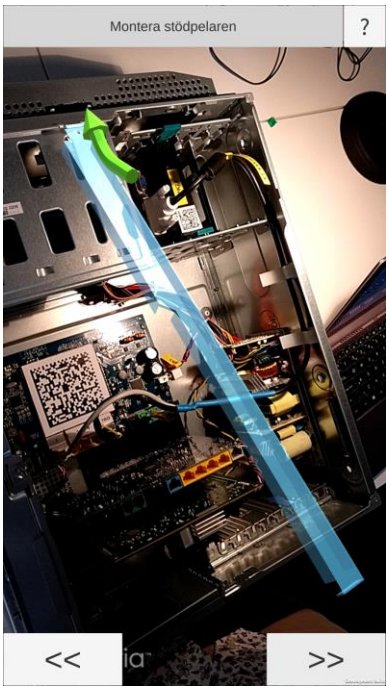
Steg 7.

Steg 8.

Steg 9.

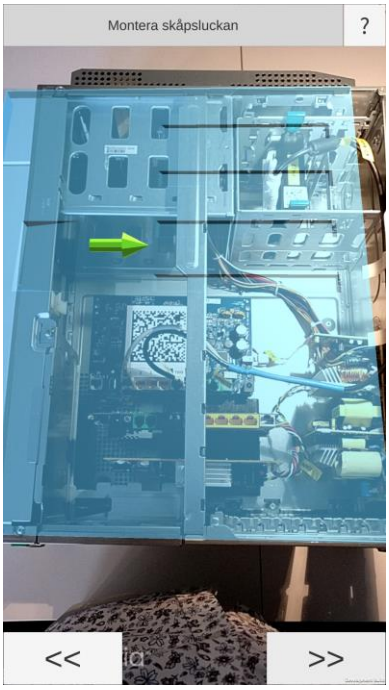


Steg 10.



Steg 11.

Steg 12.



Steg 13.

Hjälpsida som visas för användaren vid första användning och som de kan komma åt under testets gång.

Bilaga E. Svar från användartest

Deltagare med AR-instruktioner

Frågor

1. På en skala från ett till tio, där ett är enkel och tio är avancerad, var skulle du sätta uppgiften du nyss utförde?
2. Hur upplevde du att få instruktioner via en mobiltelefon?
3. Beskriv hur du upplevde instruktioner via AR.

Person 1

1. 3,5
2. Det var bra. Det kändes som att det var ganska… Det finns fördelar mot att läsa en manual. Sen känns det som att tekniken är lite knackig fortfarande. Det är det, när det väl funkar så känns det bra, jättebra. Men det är just det att den liksom, ibland så faller den bakom och så.
3. Dom är bra. Jag gillar det för de är så visuella, delen som ser ut så här ska sitta där. Det är extremt konkret och visuellt. Det känns som att det är närmre verkligheten än att läsa manual oftast.

Person 2

1. 3,5
2. När det fungerade så fungerade det skitbra. Sen så var den lite off ibland.
3. När det fungerade var det skitbra, annars var det lite svårt då fick man försöka gissa sig fram.

Person 3

1. 3
2. Jag tycker det funkar bra. Det är spännande eftersom man ser precis det där man ska göra.
3. Fördelen blir ju det att den visar precis där man ska vara och peta, så jag tycker det var häftigt.

Person 4

1. 3
2. Lite svårt, faktiskt. Man får ingen feedback, det får man inte på en instruktion i vanliga fall heller, men ja lite svårt.
3. Både bra och dåligt. Det bra för att man egentligen kan se lite så här exakt, men det är svårt för när man vinklar telefonen lite så ba', var det den här eller den här. Man får inte heller där så här... När jag flyttar telefonen då så försvinner det ju lite av det jag hade, så jag kan inte gå tillbaka egentligen och titta där jag står med fingrarna.

Person 5

1. 3
2. Det är bra. Det var jävligt bra.
3. Det var jättebra. Det var ännu bättre.

Deltagare med pappersinstruktioner

Frågor

1. På en skala från ett till tio, där ett är enkel och tio är avancerad, var skulle du sätta uppgiften du nyss utförde?
2. Hur upplevde du instruktionerna?

Person 6

1. 3
2. Det var tydligt men jag bommade på att byta kabel, för det tror jag inte stod. Eller, jo det står ju – koppla bort kabel. Jo det var tydligt. Tydliga instruktioner faktiskt.

Person 7

1. 3
2. Instruktionerna var väl goda ändå. Det var ju inte såhär IKEA-manual, alltså såhär bild för bild. Mittemellan. Det var inte Kinanivå men det var inte IKEA-nivå, så det var helt ok.

Person 8

1. 3
2. Det var ganska enkla. Man fick bläddra lite fram och tillbaks, men det funkar ju. Det var bara två sidor.

Person 9

1. 3
2. De var bra, de var tydliga. Sen letar man hela tiden såhär, är det någon som försöker lura mig. Hade jag gjort det där hemma så hade jag liksom… Alltså typ IKEA-möbel, det blir ju fel rätt så ofta. Det är ju för att, ja nu vet jag ju det ska ju vara likadant som förut. Jag försökte undvika den grejen. Var de verkligen lika, ja så.

Person 10

1. 5,5
2. Det är som jag tycker är svåra, språket, det tycker jag är lite svårt. Jag skulle egentligen velat ha en IKEA-manual nästan.