CHALMERS
UNIVERSITY OF TECHNOLOGY

Utveckling av en applikation i förstärkt verklighet som stöd för problemlösning

Examensarbete inom Data- och Informationsteknik

Jan Rasmussen
Emir Zivcic

Institutionen för Data- och Informationsteknik
CHALMERS TEKNISKA HÖGSKOLA, GÖTEBORGS UNIVERSITET
Göteborg 2020

Kandidatarbete 2020

CHALMERS
UNIVERSITY OF TECHNOLOGY

Institutionen för Data- och Informationsteknik
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg 2020

Examinator: Jonas Duregård

Institutionen för Data- och Informationsteknik
Chalmers Tekniska Högskola/Göteborgs Universitet
412 96 Göteborg
Telefon 031-772 1000

Abstract

Augmented reality (AR) is a rapidly growing research area with potential for application in a wide range of fields. In contrast to Virtual reality (VR), AR captures reality as part of the view and adds additional layers of information. The additional layers have the potential to enable optimization of problem solving using visual aid, potentially saving time and money. The famous puzzle Rubik's cube is a perfect example of such a problem, where the user is required to solve the puzzle by rotating parts of the cube in the correct direction. This can be done in a series of steps using existing algorithms for solving arbitrary cubes. Using the Microsoft HoloLens, the solution steps can be displayed directly on the cube itself, turning anyone into a master of Rubik's cube!

Innehåll

Förord

HoloCube är ett examensarbete på kandidatnivå som utförts vid institutionen för Data- och informationsteknik på Chalmers. Detta examensarbete utgör 15 högskolepoäng utav 180 på dataingenjörsprogrammet.

Vi vill framföra ett stort tack till vår handledare Matthew Connors, från Cybercoms AR/VR-avdelning i Göteborg. Vi vill även tacka Gabriel Ibanez, ledare för Cybercom Innovation Zone i Göteborg. Sist men inte minst, ett stort och varmt tack till Sakib Sistek, vår handledare på Chalmers.

1
Inledning

Det finns behov utav att kunna hitta lösningar på vardagliga problem. En förstärkt verklighet (AR) skulle kunna vara lösningen och den fungerar genom att skanna in omgivningen som personen befinner sig i och låter sedan personen påverka den nya verkligheten.

1.1 Bakgrund

Cybercom Group har kontor i fem olika länder (Sverige, Finland, Danmark, Polen och Indien), med ungefär 1200 medarbetare totalt. Inom Cybercom så läggs det stort fokus på AR/VR (Augmented Reality/Virtual Reality), IoT (Internet of Things) och cloud-tjänster. Arbetet som berör denna rapport påbörjades på Innovation Zone som är en avdelning inom Cybercom där nya ideer utvecklas.

Inom avdelningen IZ har det tidigare genomförts ett projekt vid namn HoloCube med målet att påvisa hur HoloLensen skulle kunna användas för att underlätta genomföringen av processer med specifika steg. Ett tänkt scenario hade varit att en bilmekaniker behöver t.ex inte ha kunskap av alla bilmodeller och hur de repareras eftersom att HoloLens:en kan tillhandahålla denna information [1]. I fallet av projektet så innebar detta en handhållning genom lösningen av en Rubiks kub. Projektet hann ej slutföras på Innovation Zone och var vid tillfället av överföringen mellan parterna i tidigt stadie av utvecklingen. Nuvarande HoloCube har funktioner som att kunna använda sig av en kamera för att analysera och identifiera färgen av specifikt angivna områden inom kamerans vy. Mjukvaran hade sedan en egen uppfattning om hur Rubiks kuben såg ut.

De två primära målen med projektet som kvarstår som olöst är två grundläggande funktioner. HoloCube saknar en sanningsenlig representation av kubens uppsättning av färger då färgerna inte skannas med hög precision. Det saknas även en algoritm som kan förse användaren med instruktioner som leder till en löst Rubiks kub. En lösning till dessa mål skulle kunna påvisa hur man kan använda AR som ett stöd för att lösa praktiska problem i vardagen eller inom industrin.

1.2 Syfte

Syftet med HoloCube är att utveckla en AR-applikation för att konstatera hur man kan använda AR som ett enkelt och bra stöd för att lösa praktiska problem av olika komplexitetsgrader.

1.3 Mål

Målet är att utveckla funktionerna som krävs för att få användaren att lösa en rubiks kub utan att själv faktiskt ha kunskapen om hur den ska lösas.

• Målet är att, med hjälp av en HoloLens, leda användaren från en blandad kub till en uniform kub utan att användaren har någon sorts förkunskap.

• Utgående från det befintliga systemet ska det programmeras en effektiv kublösningsalgoritm, dvs få ut en löst kub med så få instruktioner som möjligt(Gods number). Efter förstudien konstaterades att algoritmen ska kunna räkna ut en lösning till kuben inom en sekund och under 20 drag.

• HoloLens:en ska kunna hitta en kub vart som helst i sin vy med hjälp av kameran som finns på den. En del av en bra UX-design är att användarens handlingar åstadkommer deras förväntningar av vad som ska ske. För ett kvantifierbart mål frågas testgruppen om hur scenen jämförde med deras förväntningar.

• Användaren ska intuitivt förstå hur programmet används och detta uppnås genom att ha en bra UX-design. En grundpelare inom UX-design är att användaren kan gå från start till slut av programmet utan större hinder.

1.4 Givet System

Det givna system hade ingen algoritm som faktiskt löste en Rubiks kub utan den fortsatte bara att blanda om kuben. Inskanningen av färger skedde genom ett hårdkodat rutnät som användaren behövde orientera kuben inom på ett specifikt sätt för att korrekt skanning skulle ske.

1.5 Avgränsningar

Inget adaptivt ljussystem kommer att utvecklas. Detta betyder att programmet kommer bara kunna fungera i ett specifikt ljusförhållande; extremt mörkt eller extremt upplyst rum kommer inte kunna användas utan bara ljus anpassat för kontorsmiljö kommer kunna fungera.

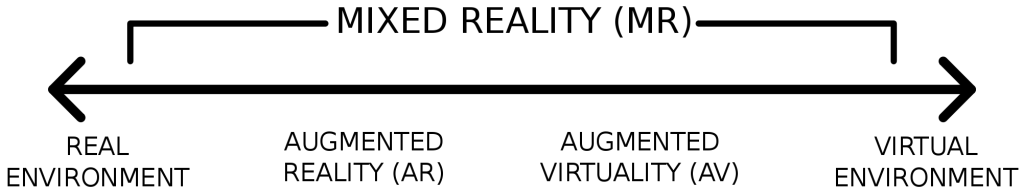
2
Teknisk Bakgrund

Augmented Reality (AR) har ingen ordentlig svensk översättning utan kallas ibland för förstärkt verklighet [2]. AR-tekniken fortsätter ständigt att utvecklas och presentera nya möjligheter i vårt sätt att lära oss, utföra uppgifter och interagera med vår omgivning. AR tillåter användaren att tillgå information utan att vara beroende av andra hjälpmedel, så som manualer eller skärmar.

2.1 Augmented Reality

Definitionen för AR är kombinationen av omgivningen med virtuell information som i sig är interaktiv i realtid och stämmer överens geometriskt [2]. AR kan appliceras i många olika fält så som underhållning, utbildning och inom industrin[källa].

För att upplevelsen faktiskt ska vara användbar krävs det en förståelse, av AR-enheten, för hur den fysiska världen användaren befinner sig i ser ut. Till skillnad från VR där tekniken utnyttjar förmågan att helt utesluta den verkliga världen så är AR på ett spektrum mellan verklighet och VR.



Figur 2.1: Spektrumet mellan verklighet och en helt virtuell miljö

2.2 Microsoft HoloLens

HoloLens är ett par smarta glasögon med förmågan att förstå geometrin i omgivningen och användarens relation till denna och på så sätt förse användaren med en immersiv AR-upplevelse där olika hologram kan placeras ut i omgivningen. Även olika ljud som spelas upp för användaren sker i 3D-miljö och kan därmed ge intrycket av att ljudet har ett hologram eller ett fysiskt objekt som källa.
Glasögonen använder sig av fyra stycken omgivningsmedvetna sensorer, en djupledskamera med en 120° x 120° vy, en 2,4 megapixel fotografisk videokamera, fyra mikrofoner och en ljussensor [3].
gestinmatning på HoloLens:en kan liknas vid musklick på en dator. Användaren tillämpar blicken för att indikera vart de vill att klicket ska ske och med en nypning i luften, i glasögonens vy, så sker klicket där användaren hade sin blick [4].

2.3 Unity3D

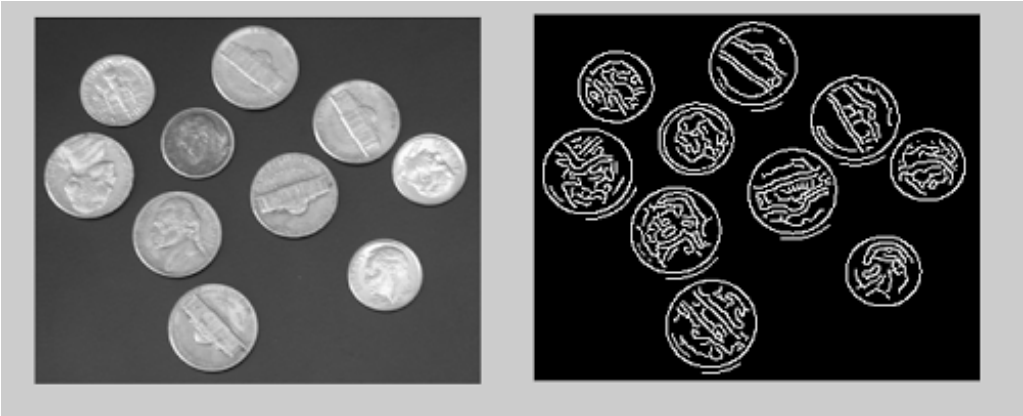
Unity är en spelmotor som kan användas för att utveckla applikationer för AR/VR till bland annat Xbox, iPhone, Android och Windows. Unity använder sig utav 3D-scener där olika objekt så som spelobjekt, ljuskällor samt användarens position och perspektiv kan placeras ut. För att interagera och avancera mellan scener och objekt utnyttjas olika skript som bestämmer vad som ska ske vid olika interaktioner. Unity har använts för många kända AR-spel; en av dessa är Pokemon Go [5].

2.4 HoloToolKit

HoloToolKit är ett Software Development Kit(SDK) utvecklat av Microsoft för underlätta med utveckling av programvara för HoloLens:en. I detta unity-paket så finns det en antal fördefinierade funktioner så som spårning av blicken och att tolka gester som användaren gör. Till exempel när användaren nyper i luften så tolkar glasögonen det som en tryckning där användaren kollar [6].

2.5 OpenCV

OpenCV är ett programvarubibliotek som används för bildanalys. Biblioteket har metoder för att kunna hitta exempelvis cirklar, ansikten och linjer i en bild. Metoder för att filtrera och förstärka önskade objekt utnyttjas för att effektivisera analysmetoderna. Fördelar med OpenCV i just detta projekt är att biblioteket underlättar för att skanna in kubens fyrkanter då det finns färdiga algoritmer för att hitta linjer och andra kvadratliknande föremål i bild. En av dessa fördefinierade algoritmer är att kunna hitta kanter i kamerabilden [7].



Figur 2.2: OpenCVs canny edge detection

2.6 Rubiks kub

En Rubiks kub är en kub med sidor av uniforma färger. En sida har nio stycken bitar och är ordnade i ett 3x3 rutnät. Varje ruta kan vridas i x- y- eller -z-led. Med hjälp av vridningar så skall man komma fram till en kub med lika färgade rutor på en och samma sida. Kuben har en permutation på $4.3x10^9$ möjliga blandningar. Om man har en kub för varje permutation så skulle man kunna täcka jorden 300 gånger om [8].

Figur 2.3: En Rubiks kub

3
Metod

3.1 Förstudier

I och med att utvecklingsområdet är relativt nytt för utvecklarna anses det passande att tid spenderas på att utforska existerande applikationer och lösningar. Detta ska ge en bättre förståelse för användaren interagerar med HoloLens:ens fördefinierade funktioner som t.ex. att klicka i vyn och orientering av digitala objekt. Det kommer även att undersökas om olika ramverk finns definerade för hur användargränsnitt ska utformas.
Ett annat stort fält att tacklas med är bildigenkänning och vilka alternativ det finns vid utveckling för Hololens i Unity. Även utbildning inom det nya ramverket för bildanalys och hur annan programvara löst olika problem ska utforskas.
Osäkerheter finns kring hur lätt det skulle vara att automatisera lösning av kuben och om det redan finns algoritmer som kan utföra lösningen och därför investeras tid i att hitta en algoritm som kan implementeras istället för att komma på en egen.

3.2 Handledarmöte

Veckovisa handledarmöten sker med en handledare från Chalmers för att diskutera framgångar, svårigheter och idéer inom projektets gång. Det har även försetts handledare ifrån företaget Cybercom som hjälper till med olika användarvänlighetsfrågor. Dessa möten har tänkts ske i och med varje iteration och vid behov då handledaren finns tillgänglig på plats.

3.3 Agilt arbetsflöde

I och med flertalet nya tekniker som tas upp under projektets gång uppkommer det svårigheter i att tidsuppskatta implementeringen av önskad funktionalitet. En utgångspunkt kommer väljas att börja utveckla från och återevalueras varje dag på den dagliga stand-upen. Därmed jobbas det utifrån en agil arbetsmetod. Avsikten med dessa standups är att vi dagligen orienterar oss tidsmässigt och går igenom vad som gjordes dagen innan och vad man hoppas åstadkomma under dagen. Vi får möjligheten att dagligen kommunicera med produktägaren ansikte-mot-ansikte och kan anpassa oss till snabb förändring och intryck. Följaktligen blir vi under tidens lopp bättre anpassade till att uppskatta egna förmågor och dessa standups blir med

tiden mer och mer värdefulla med realistiska dagliga målsättningar, leveranser och informationsförmedling.

3.4 Testning och verifiering

Övervakning av resurser såsom processorkraft och minne sker m.h.a given HoloLens och dess "Device Portal". Microsoft förser även utvecklare med en Hololens emulator där man kan testa holografiska applikationer utan en fysisk HoloLens [9]. En stor fördel med att jobba i Unity är att man kan testa de olika scenernas funktionalitet samt utseende för snabb verifiering av förbestämt beteende utan den holografiska funktionen. Verifiering kommer att ske direkt i HoloLens:en.

3.5 Testgrupp/Användarvänlighet

Testgruppen bestod av tre utvecklare på Cybercom som inte hade erfarenhet av att lösa en rubiks kub eller erfarenhet av att vara i en AR-miljö. Upplevelsen av att ta sig igenom programmet bedömdes genom att testgruppen fick svara på frågor kring användarvänligheten i programmets olika delar. Testgruppen består av tre personer som kommer att få testa applikationen vid tre olika tillfällen. En bedömmelse av applikationens upplevelse kommer att ske genom det s.k. *Tänka-högt-protokollet* där testpersonerna bes ta sig igenom programflödet medan de talar om alla intryck de får [10]. en online-enkät samt en dialog kort efter genomförandet. På enkäten blev testgruppen frågad om hur intuitivt programflödet var och de kunde även dela med sig av sina egna tankar om programmet.
Denna konstruktiva kritik hoppas till att klara upp oklarheter med vissa funktioner i applikationen.
Det är väsentligt att ha en formell enkät för att komplettera testgruppens direkta muntliga feedback om hur deras upplevelse var för att uppnå så kvalitativa och kompletta svar som möjligt. I och med online-enkäten undviker man en personlig dialog där testpersonen kanske inte vågar uttrycka sig till fullo och ger personen tid till att tänka och formulera sina tankar. En direkt feedback kan vara minst lika viktig i fall då testpersonen kan ha en ändrad upplevelse vid den senare tidpunkten enkäten fylls i. För att undvika kvarstående kunskap från tidigare test i gruppen så bestämdes det att det tredje testet skulle utökas med en ny person som inte har sett programmet för att bedömma flödet. Risken med att fortsätta med samma testgrupp är att tidigare erfarenheter med programvaran kan komma att påverka hur de upplever uppdaterade versioner och gör det svårt att mäta om programmet har blivit bättre eller om det är kvarstående kunskap som gör att de löser kuben.

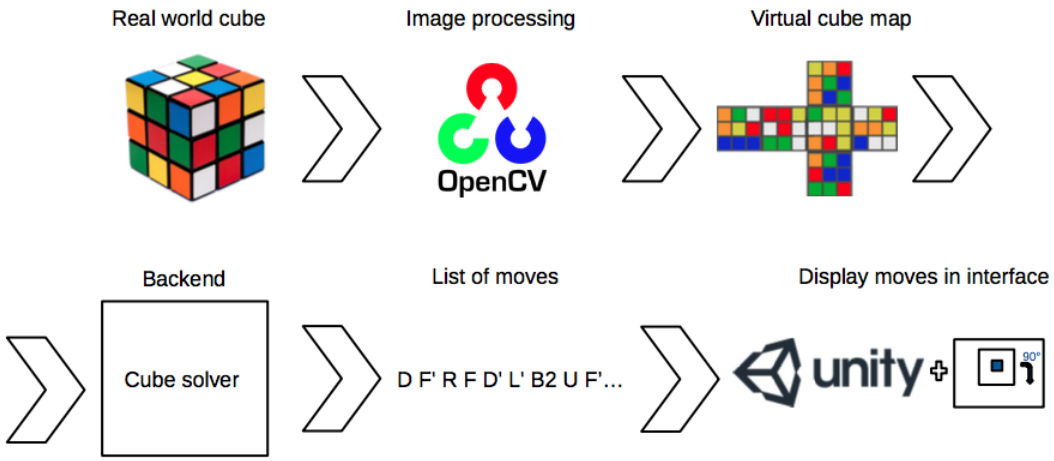
4

Genomförande

Här beskrivs de olika delmodulerna av projektet och vad som utvecklades.

4.1 Utvecklingsområden

Projektet har bestått utav sex stycken olika utvecklingsområden.



Figur 4.1: Applikationens flöde

4.2 Förstudie

Gruppen valde att göra en förstudie för utveckling av hololens-applikationer som ledde till ett preliminärt progamflöde.

1. Läsa av hololensens kamera
2. OpenCV analyserar bilderna och försöker hitta nio stycken kvadrater som utgör en sida av kuben
3. Skapa en inre representation av kubens permutation
4. Analysera permutationen av kuben och löser denna med så få drag som möjligt
5. En lista av drag skapas för att nå en löst kub
6. Visa dessa drag i användarens vy i HoloLens:en

4.3 Läsa av HoloLens kamera

Det finns en huvudkamera på hololensen som kan ta högupplösta bilder och spela in videoklipp. HoloToolKit förser inte utvecklarna med funktionen att läsa av video i realtid [11] utan sparar endast videoklipp på fil och därför krävdes det ett tillägg för att kunna analysera video i realtid. "HoloLensCameraStream" gör HoloLens-videokamerabilderna tillgängliga i realtid för Unity applikationen. Detta gör det enkelt för utvecklare att använda bildigenkänning i realtid [12].

4.4 Bildanalys

För att analysera realtidsbilderna utnyttjas biblioteket OpenCV; CV står för "Computer Vision". OpenCV konverterar
För att minimera datamängden vid varje videobild och på så sätt underlätta för bildanalys utnyttjades b.la. Cannys Edge detection. Denna kantidentifieringsmetod presterar bättre än många andra metoder inom olika aspekter så som naturlig anpassning, brusiga bilder och felaktiga kantidentifieringar [13]. Den fungerar på så sätt att den filtrerar brus och jämför kontraster i färger. För att sedan på den bearbetade bilden hitta och identifiera Rubiks kuben och dess bitar krävs det först att man hittar Rubiks kuben. Detta görs genom att söka efter fyra konturer med en 90-gradig vinkel i relation till varandra. Detta antogs vara en av de nio bitarna på den sidan av kuben. För att hjälpa användaren och programmet att identifiera rätt objekt i bilden ritas det upp ramar kring vad som tros vara en sidbit.
Programvaran hämtar ständigt nya bilder från kameran och OpenCV analyserar bilden genom att identifiera kanter, vinklar och storlek på rutorna. Om en kub skulle placeras inom kamerans vy så ska varje kub-bit kunna hittas.

4.5 Representation av kuben

En kubkarta genereras från bildanalysens uppfattning av kuben.
Mittenbiten på varje sida av kuben är unik och bestämmer vilken färg som kopplas till vilken bokstav. Om mittenbiten på kubens framsida(Front) är röd kommer alla röda bitar att hänvisas till som "F" i strängen. Färgerna skickas vidare till backend som en uppsättning av bokstäver(sträng). Denna sträng består av sex olika bokstäver (F, B, R, L, D eller U), för att kombineras till en sträng av 54 bokstäver, en bokstav för varje kubbit.
vilken färg resten av kubbitarna på den sidan ska ha för att vara uniform. Det vill säga om mittenbiten på kubens framsida (Front) är röd så kommer alla röda bitar att hänvisas som “F” i strängen.
Användaren som håller i kuben bestämmer vilken sida som ska hänvisas som till Front
Ordningen spelar roll så det är viktigt att skanna kuben som programmet frågar efter. Först skannas en valfri framsida, från denna sida vrider användaren kuben till höger och skannar tills den kommer tillbaka till framsidan och skannar sedan ovansidan och till sist undersidan.

4.6 Kublösningsalgoritm

Algoritmen som har använts i projektet är en så kallad två-fas algoritm som är utvecklad av Kociemba [14] och han har lyckats få ut en lösning för en slumpmässig kub under 200 ms och under 20 drag. Den var implementerad i Java och det enda som behövdes var att översätta den till C# för att kunna fungera i Unity.

4.7 Lista av drag

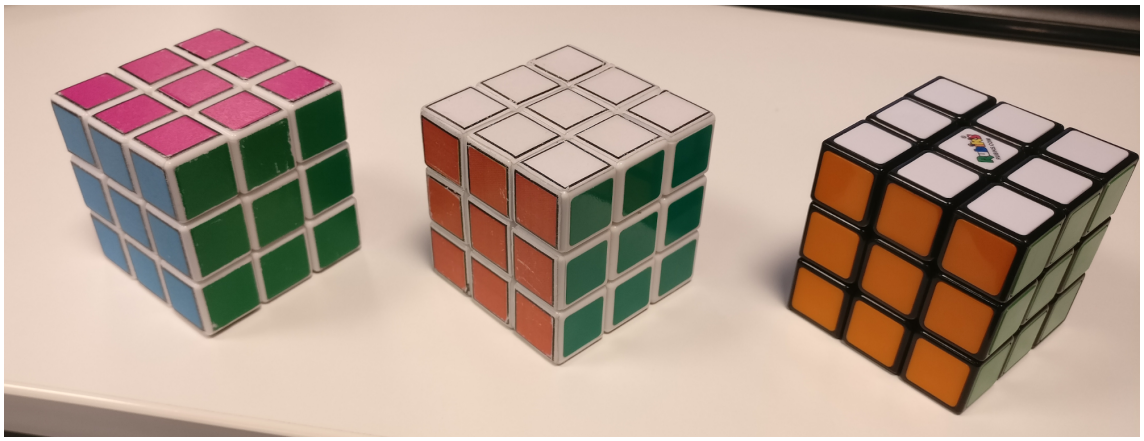
Lösningsalgoritmen efter att ha tagit in strängen med 54 bokstäver kommer att returnera dragen som krävs för att lösa kuben. Dessa drag kommer bestå av de 18 drag som användaren kan utföra på kuben. Varje sida av kuben kan vridas 90° åt höger, 90° åt vänster eller 180° åt båda håll. Om draget är klockvis 90° så används bokstaven för den sidan som ska vridas annars adderas “2” för att vrida 180° eller ett primtecken för att den ska vridas 90° motsols.

4.8 Visa drag för användaren

Programmet har ett mellanliggande system som översätter dessa instruktioner till mänskligt förståeliga instruktioner, t.ex blir instruktionen “F2”, “Turn the front Face 180°”.

4.9 Transformering av den Fysiska Kuben

Under utvecklingen kom det upp svårigheter med att identifiera färger och kvadrater i bilden. Detta ledde, istället för att försöka optimera koden mer, till att ändringar på den fysiska kuben gjordes.



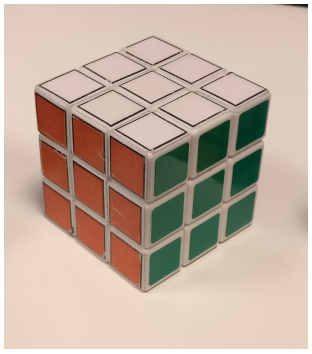
Figur 4.2: Kuberna

4.9.1 Version 1.0

Figur 4.3: Den första kuben

På denna kub identifierades aldrig de mörkare färgerna; blå, röd och grön. Detta beror på att OpenCVs “Edge Detection” hittar kanter genom att kolla på kontrasten i bilden. De blåa, röda och gröna bitarna har låg kontrast till de svarta kanterna på kuben. Däremot hittades vit gul och orange felfritt.

4.9.2 Version 2.0



Figur 4.4: Den andra kuben

Denna kub hade istället svårigheter med vit, gul och orange då kontrasten mellan vit och just dessa färger är väldigt låg och därmed hade "Edge Detection" problem med att urskilja mellan dem. Detta löstes genom att t.ex sätta vita klistermärken med en svart kant på den vita sidan för att öka kontrasten. Detta gav stora förbättringar och alla kuber kunde identifieras.

4.9.3 Version 3.0

Figur 4.5: Den tredje kuben

Den andra versionen hade fortfarande ett problem kvar. De vita, gula och orangea bitarna identifierades som varandra ibland då färgernas färgkoder överlappar med varandra. Denna iteration av kub ledde till att färgerna vit och orange byttes ut mot magenta och cyan för att öka kontraster mellan färgerna på HSV-spektrumet. Kuben var i detta stadie felfritt för bildanalys i särskilda ljusförhållanden.
De sex slutfärdiga färgerna baserades på en HSV skala där S(Saturation) och V(Value) sattes till maximum och färgerna 60 punkter ifrån varandra på en 360-skala i och med enkelheten att bara analysera en kanal med så stora kontraster som möjligt till skillnad från tre i RGB-spektrumet.

4.10 Testgruppens feedback

När en *Minimum Viable Product* utvecklats är det viktigt att mäta vilka och hur mycket av de uppsatta målen programmet nått upp till. En *Minimum Viable Product* innebär att primitiva versioner av basfunktionerna finns tillgängliga och att det finns något konkret att presentera och analysera. Vid detta tillfälle kunde programmet med hjälp av en kamera automatiskt hitta en rubiks kub i sin vy, skanna av uppsättning av kubbitar, lösa kuben samt ge användaren instruktioner på hur den löses. För att undvika förutfattade meningar om hur effektiva och användarvänliga funktionerna är togs en testgrupp in för att testköra produkten. Testindividerna kritiserade produkten i sin helhet men även varje delmoment från start till slut. Utvecklarna satt bredvid vid testkörningen för att anteckna vad som sades enligt tänka-högt-protokollet och testindividerna ombads att svara på frågan *Vad trodde du skulle hända där?* efter varje handling. Kort efter att programflödet gåtts igenom får nu individen möjlighet att kritisera programmet i sin helhet genom en dialog med utvecklarna. En online-enkät skickas ut till testgruppens medlemmar där det efterfrågas kompletterande synpunkter om programmet som de möjligen inte kunde formulera vid testtillfället och för att anonymt förmedla åsikter som de är rädda kan såra utvecklarna [15]. För att filtrera och kvalificera vilken kritik att följa upp utnyttjas *triangulering* som innebär att fenomen från minst två olika perspektiv undersöks [16].
Detta gav utvecklarna utomstående perspektiv inom vilka utvecklingsområden framtida resurser bör investeras.

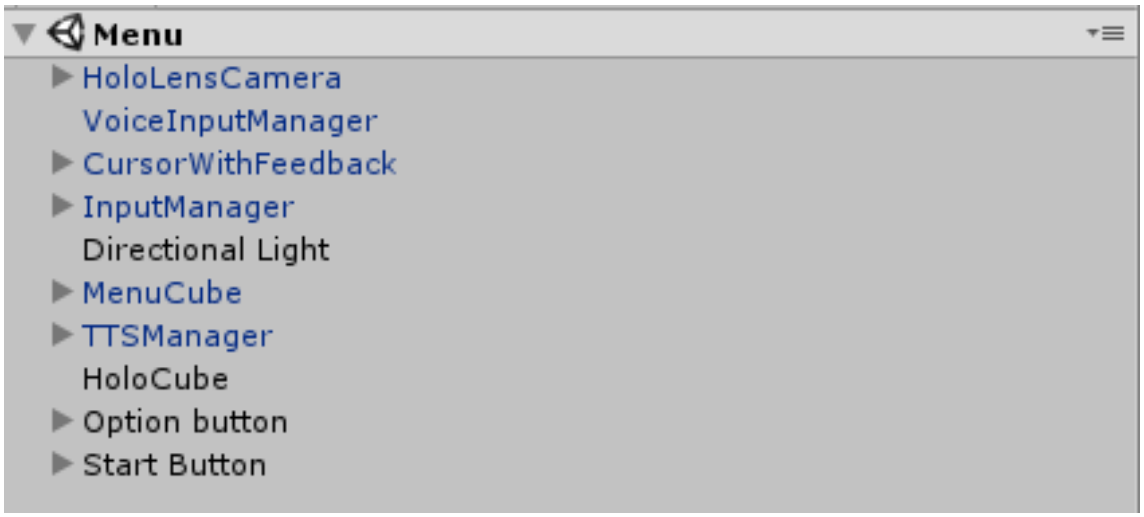
Efter första gången som programmet testades av gruppen så tyckte de att inskanningen var långdragen eftersom att en felskanning innebar att sidan behövde skannas om helt. Lösningen var att införa en verifieringsscen där gruppen kunde istället korrigera misstagen under skanningen. Även en animerad kub lades till i scenen för att förtydliga hur kuben ska skannas och roteras. Testindividerna kunde vid detta tillfälle ej lösa en rubiks kub oavsett mängd tid.

Efter andra iterationen tycks det bland testindividerna att grundfunktionerna av programmet utförs utan större problem. Det påpekades dock att kameran ofta blandade ihop färgerna på grund av den låga kontrasten mellan vissa färger. Detta ledde till att kuben transformerades till det som syns i figur 4.5. Individerna tycks även ha svårigheter med verifierings/lösningsccenen där instruktionerna presenterades med otydliga termer som 'D'. En snabb modul introducerades för att översätta dessa till mer tydliga instruktioner. Testindividerna kunde efter denna iteration lösa rubiks kuben med hjälp av programvaran inom en rimlig tid av 10 till 15 minuter.

Under den tredje testiterationen introducerades en ny testperson som ej hade tidigare kunskap om HoloLensen och Holocube. Testpersonen kritiserade otydligheten av vilken färg som var vald i färgpaletten som ledde till att den aktiva färgen markerades genom att rama in den som i figur 5.3. Testpersonen hade annars inga större hinder och kunde lösa rubiks kuben med hjälp av programvaran.

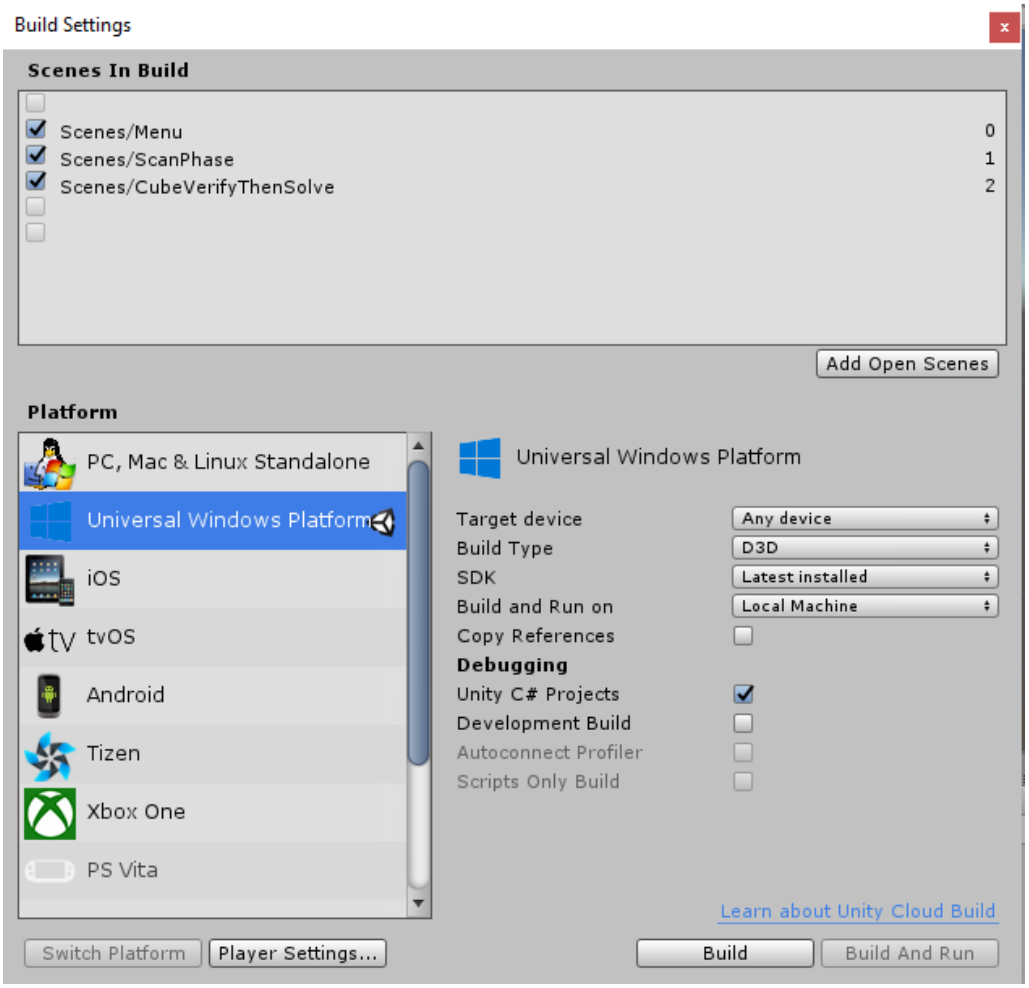
4.11 Unityinställningar för VR & HoloLensutveckling

För att AR-utveckling för HoloLens:en ska vara möjlig med hjälp av HoloToolKit så krävs det att rätt inställningar appliceras. Det underlättar att ta bort standardkameran i Unity och ersätta denna med HoloToolKits "HoloLensCamera".
Det behövs även en "CursorWithFeedback" för att användaren ska få feedback om vart fokuset ligger, som alltid är i mitten av användarens vy, och vilka objekt som går att klicka på och interagera med.
"InputManager" hanterar indata för att utföra själva klicket.



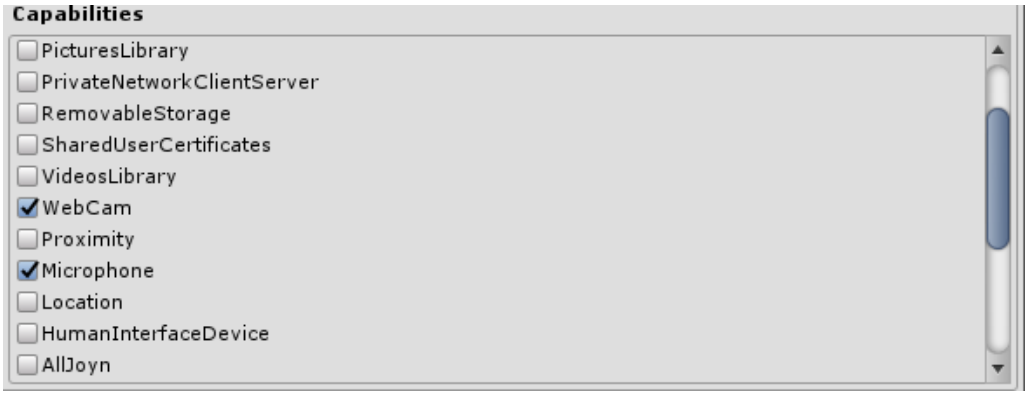
Figur 4.6: Objekt i scenen

I Unitys bygginställningar så måste "Unity C# Projects" vara markerad under "Debugging" och UWP (Universal Windows Platform) måste vara vald under "Platform".



Figur 4.7: Bygginställningar i Unity

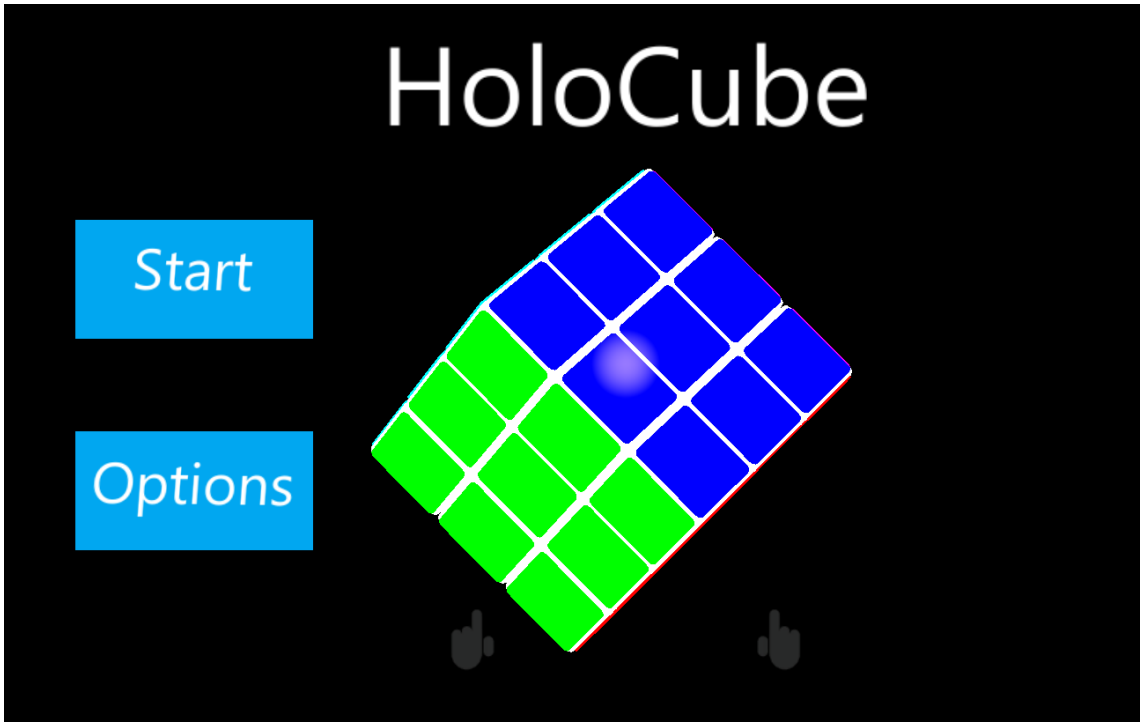
I “PlayerSettings” under “Other Settings” finns “Capabilities” och där måste “WebCam” och “Microphone” vara markerad för att mikrofonen och kameran ska kunna användas under körning för bildanalys och röstigenkänning.



Figur 4.8: Inställningar för tillstånd

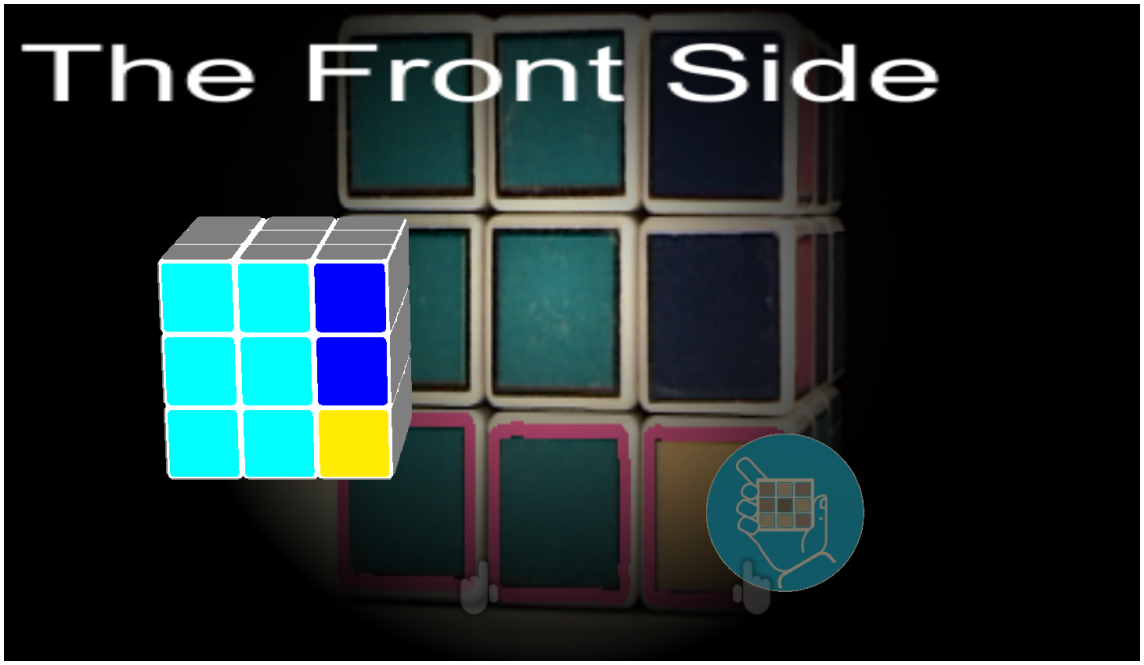
5
Resultat

Efter att programvaran gått igenom flera körningar av testgruppen så blev HoloCube godtyckligt användbar med ett tydligt grafiskt användergränssnitt. Samtliga i testgruppen kan nu navigera sig igenom programmet för att få hjälp med att lösa en rubiks kub inom den uppsatta tidsramen. Kontrollpersonen som introducerades i slutet för att eliminera potentiell förkunskap kunde utan större hinder gå från olöst till löst kub inom tidsramen.
Startscenen är där för att välkomna användaren till programmet med hjälp av en TTS-bot (Text-To-Speech) från Adobe Captivate. Kuben i scenen roterar 360° runt sin axel med hjälp av animeringsfunktionen i Unity. När användaren klickar "Start" så börjar nästa scen som är skanningsfasen.



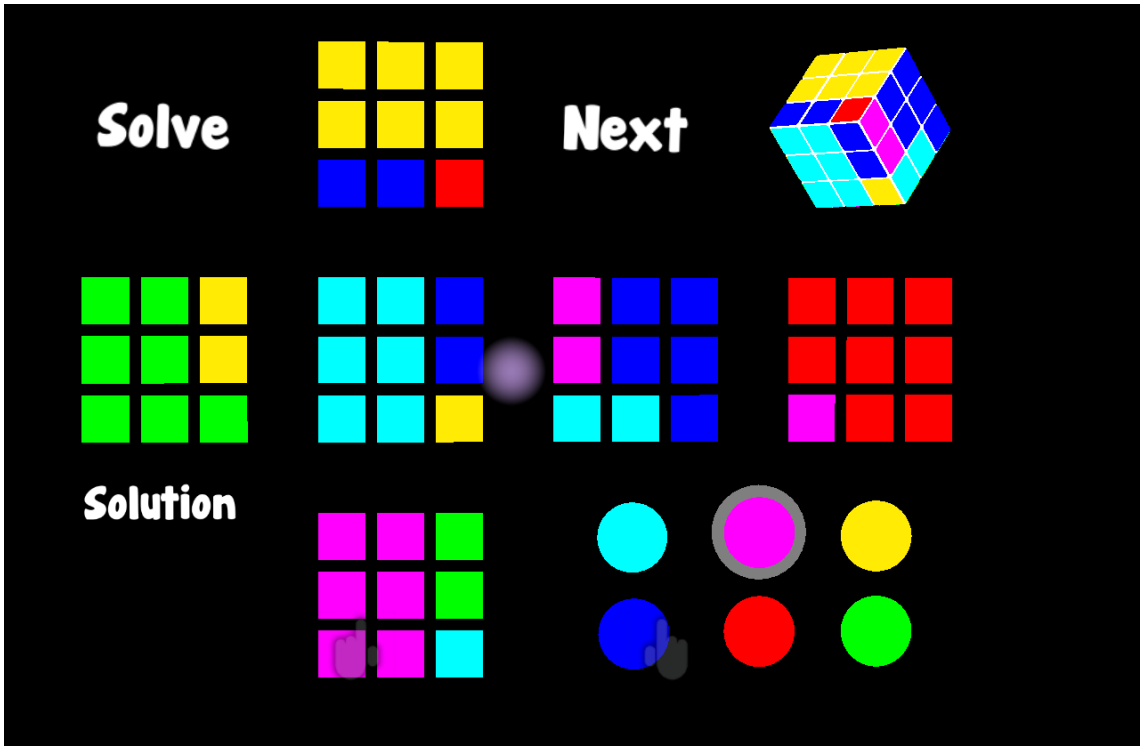
Figur 5.1: Startscenen

Skanscenen vägleder användaren genom skanningen av rubiks kuben. TTS-boten gav även hjälp genom att förklara i en mening att det är dags att skanna in en valfri sida och sedan hur den ska roteras efteråt. Ikonen som finns med i nedre högra hörnet föreslår hur användaren ska hålla i kuben framför kameran för att hela sidan ska kunna upptas och att inga fingrar är i vägen eftersom att kuben bara skannas när nio stycken kubbitar hittas. I denna scen nyttjas en animerad kub för att ge feedback till användaren om vilka färger som uppfattats och vilken relation till varandra de har. När användaren bekräftat en sida med ett nyp i luften roterar den animerade kuben till nästa sida. Kuben rör sig marginellt fram och tillbaka för att underlätta orientering av kuben för användaren. Efter att bekräftat den sista av de sex sidorna så skickas användaren till nästa scen.



Figur 5.2: Skanfasen

Scenen för verifiering och lösning har en del funktioner. Ett av de primära målen med scenen är att användaren ska rätta små misstag som har hänt under skanningen. Detta görs genom att den uppfattade permutationen ritas upp i en 2D-karta med en kompletterande animerad 3D kub som roterar runt sin egen axel för att visa hela kuben i relation till sig själv. Scenen förser sedan användaren med en färgpalett i nedre högra hörnet där användaren väljer en färg och rättar misstag i 2D-kartan. För att scenens funktion skulle vara tydligare så kommunicerade TTS-boten att denna scen är gjord för att rätta eventuella misstag med hjälp av färgpaletten.
När användaren tror sig ha en färdig representation av kuben kan denne trycka på "Solve". Programmet bekräftar då att uppställningen är korrekt och skickar sedan in permutationen till backend för att lösas. Användaren får instruktionerna i nedre vänstra hörnet och följer varje instruktion genom att klicka "Next" tills kuben är löst.



Figur 5.3: Verifiering och lösning

6
Diskussion

HoloLens:en idag är relativt dyr i förhållande till andra AR-enheters möjligheter och det kan vara svårt att rättfärdiga behovet av en HoloLens [17]. Vi bakom HoloCube ser inte någon större potential för HoloCube att ta en omfattande del av marknaden för assisterande inlärning. Detta kommer med stor sannolikhet förändras i framtiden när hårdvaran blir billigare. Vi ser däremot mycket större potential i att föra över temporär kunskap till användaren vid till exempel lättare arbeten så som mindre reperationer eller assemblering av möbler.

Det är en bra idé att ändra på den fysiska världen när det jobbas med bildigenkänning i tidigt stadie med tanke på att större konstraster ger en bättre igenkänning. De originella rubiks kuberna använder sig bland annat av färgerna röd, orange och gul som i tidigt utvecklingsstadie gav oss större svårigheter i att urskilja. Även den blåa färgen och de svarta kanterna visade sig vara svåra att differentiera i sämre ljusförhållanden. Det blev svårt att specifiera och identifiera exakta värden för de sex originella färgerna i ett RGB förhållande där tre parametrar av färg ska analyseras och sättas i relation.
I detta projekt så byttes den fysiska kuben ut för att underlätta för programmet att identifiera kuber istället för att fortsätta jobba på koden som bara gav marginella förbättringar.

Problem uppstod med git eftersom att Unity-projekten inte hade rätt inställning. Unity skapar en del .meta-filer som håller reda på t.ex objektreferenser. Att bara öppna spelmotorn skapar eller ändrar tiotals meta-filer. Det är en bra idé att inte lägga upp dessa filer på git eftersom att flera användare kommer att ha olika meta-filer och det skapar bara problem genom att scenen inte laddar in ordentligt [18].

Ett projekt bör ha en planering i form av ett GANTT-schema och medlemmarna bör föra en dagbok för att lättare kunna blicka bakåt, över svårigheter och hur de löstes, när det är dags för rapportskrivning. GANTT-schemat kan bli svårt att följa, som i det här projektet, då vissa delmål kan ta längre tid än förväntat; därför bör medlemmarna alltid räkna med att vissa moment kommer dröja längre i produktion än planerat.

7
Kritisk Diskussion

Det är viktigt att delegera tid i början av ett projekt till att överväga andra möjliga teknologier. I detta projekt valdes det att direkt använda de påbörjade teknologierna, som kanske inte var det mest optimala. AR fungerar t.ex även på mobilen och bör ha utforskats som en möjlighet eftersom att fördelarna med att använda glasögonen är små men glasögonen är mycket dyrare och en större del av människor har redan en AR-kapabel telefon. AR-Core kanske är mer praktiskt för utvecklare och har ett bättre underliggande system för bildigenkänning-applikationer. En annan möjlighet var att programmera i C++ med hjälp av Unreal som är en annan spelmotor. Denna spelmotor kanske är bättre optimerad för AR-programmering för glasögonen eller att C++ p.g.a sin manuella minneshantering kan visa sig vara ett effektivare språk än C#. för optimering.

Innan projektet påbörjades förekom det många diskussioner om vad projektets *mål* skulle vara och vad vi ville åstadkomma förutom att enbart utveckla något halvt användbart. Vi började med väldigt ambitiösa målsättningar så som att lära ut nya kunskaper och faktiskt konkurrera med liknande hjälpmedel. Det visade sig med projektets gång att omfattningen kring sådana mål var större än vad våra förutsättningar kunde förse oss med. Trots AR-teknologins långa historia finns det relativt få experter jämfört med andra ingenjörsfält och expertis är svår att hitta när vi avgränsat oss tidigt mot HoloLensen [19].

Det blev uppenbart i sent stadie av projektet att syftet blev mer och mer komplicerat att kombinera med målet av att utveckla en applikation för att lösa rubiks kuber och presentera lösningar. Fokus lades disproportionerligt på att identifiera kuben, som inte är en komponent av AR, istället för att förstärka verkligheten. Kortfattat hade det varit bättre att antingen fokusera på att samla in information på ett så smidigt sätt som möjligt, eller på att presentera information i AR och inte försöka kombinera dessa.

HoloLensens och AR-teknologins styrkor ligger i att känna igen omgivningen som inte alls var användbart för att uppnå våra mål. Hade vi enbart fokuserat på att bildanalysera en rubiks kub och spotta ut lösningar skulle vi kunna utnyttja kameror folk redan har tillgängliga, så som webbkameror och mobilkameror. Om vi däremot gått åt andra hållet och försökt göra ett AR-hjälpmedel är det nog smartare att luta sig åt en annan vinkel än att samla in nödvändig information med hjälp av en HoloLens kamera. Det hade t.ex. varit underlättande att utveckla ett hjälpmedel som ej kräver utomstående information om denna informationen inte råkar vara dimensionerna av rummet och dess innehåll samt deras fysiska relation till varandra. Ett bra exempel på något som utnyttjar AR så som det var tänkt är IKEA Place [20] där man kan placera virtuella möbler i sitt rum med hjälp av AR.

8

Slutsats och Vidareutveckling

För en vidareutveckling av HoloLens föreslås följande åtgärder för en mer komplett och användbar upplevelse för användaren.

- Ett adaptivt ljussystem som tar i hänsyn till kamerans uppfattning av omgivningens ljusförhållanden. I och med ett sådant system kan man i stor omfattning uppnå en mycket lägre felmarginal när det kommer till programmets färguppfattning. Utöver ett tryggare konfidensintervall för avgörandet av färg samt mindre tid spenderad i verifikationsfasen erhåller man dessutom tidsbesparingar i själva skanningsfasen i och med att man nu potentiellt skulle kunna förse OpenCV metoderna med gynnsammare parametervärden. Därmed kan användaren undgå ett antal försök i att hitta korrekt vinkel för kuben.

- En instruktionscen med fokus på att överlämna kunskap till användaren och inte bara lösa uppgiften för tillfället. Förslagsvis kan man implementera tillvägagångssättet att inte ge de mest optimerade stegen för att lösa kuben utan uppmanar användaren att ta sig till olika uppsättningar av kuben själv. Ett allmänt bekant sätt att lära sig lösa sin första Rubiks kub är nybörjarguiden som ber användaren sätta upp ett kors av vita kuber för att sedan fylla i kanterna till en full vit sida och arbeta sig därifrån.

- Hos en av testpersonerna fick vi veta att denna person hade svårt att hänga med under instruktionsscenen samt verifikationsfasen då det krävs att man ska kunna orientera sin verkliga kub mot den virtuella kuben. Detta problem har många fantasifulla lösningar och man skulle exempelvis kunna implementera nya värden för att tolka andra uppställningar färger, och måla en ny verklig kub. Man skulle även kunna introducera symboler eller text som tillvägagångssätt att orientera sig igenom programmet.

- Programmet i dagsläget har egentligen ingen förståelse för vad en Rubiks kub är och hur förhållandet mellan färgerna, sidorna och kuberna är. Detta medför att skanningsfasen förblir väldigt enformig och kan vara alltför beroende av användarens noggrannhet för att få till en lyckad skan. En vidareutveckling där man implementerar en uppfattning av hur en Rubiks cube ska se ut hade kunnat innebära en skanningsfas där enskilda gap automatiskt fylls i.

Referenser

[1] R. Silva, J. Oliveira, and G. Giraldi, "Introduction to augmented reality," 01 2003.

[2] [Online]. Available: http://augmentedreality.se

[3] [Online]. Available: https://www.microsoft.com/en-us/hololens/hardware

[4] [Online]. Available: https://www.youtube.com/watch?v=kwn9Lh0E_vU

[5] [Online]. Available: https://en.wikipedia.org/wiki/Pokémon_Go

[6] [Online]. Available: https://microsoft.github.io/MixedRealityToolkit-Unity/Documentation/Input/Gestures.html

[7] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[8] [Online]. Available: https://ruwix.com/the-rubiks-cube/mathematics-of-the-rubiks-cube-permutation-group/

[9] [Online]. Available: https://docs.microsoft.com/en-us/windows/mixed-reality/using-the-hololens-emulator

[10] J. Preece, Y. Rogers, and H. Sharp, *Interaction Design: Beyond Human-Computer Interaction*, 4th ed. Hoboken, NJ: Wiley, 2015.

[11] [Online]. Available: https://docs.microsoft.com/en-us/windows/mixed-reality/locatable-camera-in-unity

[12] [Online]. Available: https://github.com/VulcanTechnologies/HoloLensCameraStream

[13] Rashmi and R. Saxena, "Algorithm and technique on various edge detection : A survey," *Signal Image Processing : An International Journal*, vol. 4, pp. 65–75, 06 2013.

[14] H. Kociemba. [Online]. Available: http://kociemba.org/cube.htm

[15] S. Haffar, F. Bazerbachi, and M. H. Murad, "Peer review bias: A critical review," *Mayo Clinic Proceedings*, vol. 94, no. 4, pp. 670–676, Apr 2019. [Online]. Available: https://doi.org/10.1016/j.mayocp.2018.09.004

[16] V. Jupp, *The SAGE Dictionary of Social Research Methods*. SAGE Publications, 2006. [Online]. Available: https://books.google.se/books?id=RyiL-Hi0wFcC

[17] [Online]. Available: https://en.wikipedia.org/wiki/Comparison_of_virtual_reality_headsets#Standalone_(mobile_device_mounts_and_all-in-ones)

[18] [Online]. Available: https://thoughtbot.com/blog/how-to-git-with-unity

[19] [Online]. Available: https://theappsolutions.com/blog/development/augmented-reality-challenges/)

[20] [Online]. Available: https://apps.apple.com/se/app/ikea-place/id1279244498