BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

AUGMENTED REALITY IN COCKPIT

UPRAVENÁ REALITA V KOKPITU

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR **Bc. MICHAL TABÁŠEK**
AUTOR PRÁCE

SUPERVISOR **prof. Dr. Ing. PAVEL ZEMČÍK, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2020

Brno University of Technology
Faculty of Information Technology

Department of Computer Graphics and Multimedia (DCGM) Academic year 2019/2020

Master's Thesis Specification

Student: **Tabášek Michal, Bc.**
Programme: Information Technology Field of study: Computer and Embedded Systems
Title: **Augmented Reality in Cockpit**
Category: User Interfaces
Assignment:

1. Study the topic of augmented reality in aeroplane cockpit, relevant literature and solutions for visualization of checklists, and evaluation of their usability, position referencing, GUI on Head-mounted display, etc.
2. Explore and identify the suitable hardware for the task, keeping in mind the operator needs, operation time and standby time, context and lighting conditions, affordability, etc.
3. Explore, identify, and select a compatible rendering engine for the explored hardware, taking into account head position and attitude referencing.
4. Prepare a suitable example of e.g. checklist development and suggests a SW architecture.
5. Implement the augmented reality using the selected hardware and demonstrate its functionality.
6. Describe the achieved results and possibilities of continuing the work.

Recommended literature:

- Based on instructions from the supervisor.

Requirements for the semestral defence:

- Items 1 to 3.

Detailed formal requirements can be found at https://www.fit.vut.cz/study/theses/

Supervisor: **Zemčík Pavel, prof. Dr. Ing.**
Head of Department: Černocký Jan, doc. Dr. Ing.
Beginning of work: November 1, 2019
Submission deadline: June 3, 2020
Approval date: November 5, 2019

Abstract

This thesis deals with the usage of augmented reality in aviation. In this area, it focuses on the cockpit checklist used in each flight phase to ensure that all important tasks are performed. An experimental mobile application is developed to guide the pilot step by step through the cockpit checklist. Each step requires some action to be performed on a particular flight instrument located in the aircraft cockpit. The appropriate flight instrument, associated with the active task, is identified and tagged with a suitable visualization. The experimental application is implemented using the Unity engine and the AR Foundation framework. The implemented application has been tested during flight with positive feedback from participants.

Abstrakt

Tato diplomová práce se zabývá využitím rozšířené reality v letectví. V této oblasti se zaměřuje na kontrolní seznamy v kokpitu využívané ve všech fázích letu k zajištění vykonání všech důležitých úkonů. Pro tento účel je vyvinuta mobilní aplikace, jejímž cílem je provést pilota krok po kroku kontrolním seznamem. Každý krok vyžaduje provedení určitého úkonu na konkrétním letovém nástroji nacházejícím se v kokpitu letadla. Příslušný letový nástroj spojený s aktivním úkolem je identifikován a označen vhodnou vizualizací. Experimentální aplikace je implementována s využitím herního nástroje Unity a AR Foundation frameworku. Implementovaná aplikace byla testována během letu s pozitivní zpětnou vazbou od účastníků testování.

Keywords

Augmented reality, Unity, iPhone, iOS, AR Foundation, C#, aviation, cockpit checklist.

Klíčová slova

Rozšířená realita, Unity, iPhone, iOS, AR Foundation, C#, letectví, kontrolní seznam kokpitu.

Reference

TABÁŠEK, Michal. *Augmented reality in Cockpit.* Brno, 2020. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor prof. Dr. Ing. Pavel Zemčík, Ph.D.

Rozšířený abstrakt

Cílem této diplomové práce je prozkoumat možnosti využití rozšířené reality během provádění kontrolního seznamu úkonů. Kontrolní seznamy úkonů jsou v letectví běžně známé pod anglickým výrazem checklisty. Checklist je tvořen úkony takovými, aby jejich provedením bylo zajištěno dodržení standardních provozních postupů uvedených v provozní příručce dodavatele. Tato práce se zaměřuje na tzv. normal checklisty, které mají být členy posádky používány ve všech fázích letu za běžného provozu. Provozní příručka dodavatele však obsahuje i tzv. non-normal checklisty, které mají být členy posádky používány za nouzových a mimořádných situací.

První polovina práce poskytuje stručný přehled v oblasti rozšířené reality a metod využívaných pro vykreslení virtuální objektů do prostředí reálného světa. Do přehledu jsou zařazena vybraná zařízení pro zobrazení rozšířené reality a některé aktuálně dostupné knihovny a nástroje využívané vývojáři. Krátce jsou představeny již existující koncepty a aplikace rozšířené reality v oblasti letectví, i když ne všechna se nutně zaměřují na kontrolní seznamy úkonů. První polovinu práce uzávírá představení hlavních certifikačních a standardizačních autorit, jejichž nařízení je nutné v našem prostředí následovat. Je proveden stručný popis funkcí konstrolních seznamů implementovaných v leteckých systémech. Tuto část práce uzavírá představení vybraných aplikací cílených na zařízení s operačním systémem iOS, využívaných především piloty ve všeobecném letectví.

Druhá polovina práce popisuje aplikaci nabytých znalostí pro návrh a implementaci demonstační aplikace. Úvod je zaměřen na bližší specifikaci požadavků na aplikaci, které byly částečně upravovány během konzultací ve firmě Honeywell. Navazuje analýza stávajících řešení a identifikace jejich slabých míst, jejichž vliv by mohl být minimalizován na základě navrženého konceptu. Je popsána architektura systému, v jejímž rámci je představeno JSON schéma navržené pro účely aplikace, definující datovou strukturu kontrolního seznamu úkonů. Následuje představení navrženého uživatelského rozhraní, které si klade z cíl respektovat regulační standardy.

Navazující kapitola popisuje implementaci experimentální aplikace. Pro vývoj byl použit jazyk C#. Implementační část zabývající se rozšířenou realitou využívá rozhraní AR Foundation. Způsob mapování reálného světa je spolu s detekcí a sledováním předem známého vzoru popsán v následující sekci. Implementace ze zabývá vykreslením a spravným umístěním elementů rozšířené reality, které jsou získány na základě 3D modelu. Aplikace je složena z komponent, které implementují specifickou část funkcionality. K implementaci komponent, které netvoří jádro aplikace, byly využity již existující zdrojové soubory a balíčky. Jejich autoři jsou ve zdrojových souborech řádně uvedeni. V průběhu vývoje byla aplikace testována za jízdý v automobilu, což mělo za účel přiblížení se podmínkám vznikajícím za letu.

Vytvořená aplikace byla testována v letounu Tecnam P2002 Sierra. Experimentu se zúčastnili tři zájemci, z nichž jeden je pilot se zkušeností z produktového vývoje v oblasti letectví. Každý z účastníků obdržel evaluační dotazník a vyplňoval jej v průběhu experimentu. Experiment byl rozdělen do dvou fázi. V první fázi si každý účastník vyzkoušel aplikaci ve statickém prostředí na zemi. Všichni provedli kontrolní seznam související se spuštěním motoru a několik dalších, dle vlastního uvážení. V závislosti na první fázi byl upraven testovací scénář a byl vytvořen speciální kontrolní seznam pro druhou fázi experimentu. Poté proběhla krátká předletová příprava a přeslo se ke druhé fázi experimentu, která probíhala v dynamických podmínkách za letu.

V rámci experimentů byla zkoumána celková funcionalita a stabilita aplikace spolu s vnímáním uživatelského rozhraní uživatelem. Uživatelské rozhraní bylo hodnoceno velmi kladně a dosáhlo celkového hodnocení 4.76/5. Stabilita aplikace a čitelnosti symbolů rozšířené reality se lišily v závislosti na fázi experimentu. V první fázi experimentů byla aplikace stabilní s dobrou čitelností a viditelností jednotlivých elementů. V první fázi experimentů byla stabilita ohodnocena 4/5 a ve druhé fázi testování bylo hodnocení sníženo na 2.33/5. Problémy se stabilitou byly způsobeny výpadky v detekování značek během trubulencí. V závěru práce je naznačen možný budoucí vývoje aplikace.

Augmented reality in Cockpit

Declaration

I hereby declare that this Master's thesis was prepared as an original work by the author under the supervision of Mr. Prof. Dr. Ing. Pavel Zemčík. The supplementary information was provided by Mr. Ing. Jan Bílek and other Honeywell Aerospace employees. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

.....
Michal Tabášek
June 3, 2020

Acknowledgements

I would like to thank prof. Dr. Ing. Pavel Zemčík as well as Ing. Jan Bílek, and other professionals for their professional advice, helpfulness, and great patience. Also special thanks to Kamil Rozkopal, who made it possible to test the application in flight.

Contents

1 Introduction **7**

2 Augmented Reality **9**
2.1 Augmented Reality Concept 9
2.2 Methods and Algorithms Used in AR 10
2.3 Augmented Reality Devices 17
2.4 Frameworks and SDKs Overview 22
2.5 Rendering Engines Overview 26

3 Augmented Reality and Aviation **27**
3.1 Augmented Reality Solutions in Maintenance 27
3.2 Augmented Reality Solutions in Aviation 28
3.3 Cockpit Checklist 31

4 State of the Art Solutions of Electronic Checklist **36**
4.1 Regulatory Requirements 36
4.2 Checklist on Cockpit Displays 38
4.3 Checklist Applications for iOS 42

5 State of the Art Analysis and Application Requirements **45**
5.1 State of the Art Analysis and Proposed Solution 45
5.2 System Requirements 46
5.3 System Architecture 47
5.4 User Interface Proposal 50

6 Implementation **53**
6.1 Outline of the Application 53
6.2 Scene Understanding and World Tracking 53
6.3 Virtual World 54
6.4 Rendering Virtual and Real World 57
6.5 Graphical User Interface 58

7 Experiments and Reached Results **62**
7.1 Experiments Description 62
7.2 Experiments Realization 63
7.3 Reached Results 66

8 Conclusion and Future Directions **69**

Bibliography **70**

A Contents of the included storage media **73**

B User Experience Survey **74**

List of Figures

2.3 Point projecting through image plane[0]. 11
2.7 Left: Tracking along strong edges. Right: SIFT interest points.[0] 16
2.18 View-dependent spatial AR[0]. 22
2.19 AR Foundation API architecture 24

3.8 Checklist attached to yoke in Boeing 737 33
3.9 C-46 Commando aircraft scroll checklist 33
3.10 Mechanical checklist 34

4.1 Usual figures explaining a problem with blue color 38
4.2 Boeing 777 flight deck layout 39
4.3 ECL normal and non-normal procedure selection 39
4.4 Pre-flight checklist related to normal procedures 40
4.5 Left: Conditional statement with countdown timer (timer is marked yellow). Middle: False conditional statement. Right: Open-loop conditional statement 41
4.6 Multi-page checklist[0]. 41
4.8 ForeFlight Checklist application 43
4.9 Garmin Pilot EFB screenshots. 43
4.10 The smartCHECK application screenshots. 44

5.1 Proposed system architecture. 47
5.2 Checklist data structures. 48
5.3 Basic drawing illustrating all possible elements on on the screen. 50

6.1 Outline of the application. 53
6.2 Left: Visualized accumulated frames of mapped real work with found tracked feature points[1]. Right: Real world environment[1]. 54
6.3 Virtual world block schema. 54
6.4 Comparison of simulator's cockpit and virtual model. 55
6.5 Flow chart diagram illustrating state machines controlling the application runtime. 56
6.6 Left: Initial state. Right: When an aircraft makes a turn, the distance value has to be recalculated with respect to camera's rotation. 57
6.7 Left: Required state. Right: The system has lost tracking ability. 58
6.8 Left: Initial design. Right: Final design. 58
6.9 Top panel displays progress indicator, title, step back button, and drop down checklist menu. The checklist menu differs with respect to the displayed list. The reset button is displayed always. If a checklist is displayed on the task panel, the checklist menu contains defer and recall buttons. 59

6.10 After launching the application, the user is forced to detect all markers. . . 59
6.11 Left: Flight instrument marking in the default mode. Right: An active step can be deferred through Defer button. 60
6.12 Left: The Active+Next mode. Right: The Active+All mode. 60
6.13 Left: Message indicating completion of the checklist. Right: Active task was confirmed but flight instrument was not even in the field of view. 61
6.14 Setting panel allows to enable or disable implemented features and change highlighter mode. 61

7.1 Left: Tecnam P2002 Sierra. Right: Tecnam's flight deck. 63
7.2 Comparison of flight deck on the left side and virtual 3D model on the right side. Model elements marked blue represents an image markers. 64
7.3 Photos taken during on ground testing. 65
7.4 Photo compilation was taken from GoPro camera and iPhone screen recording during flight tests. 66

List of Tables

5.1 Comparison of aircraft systems and mobile applications. 45
5.2 AR devices price and battery life overview. 46
5.3 Left: Font style description. Right: Hexadecimal color codes. 51
5.4 Checklist menu operations with description. 51
5.5 Application features with description. 51
5.6 Application features with description. 52

7.1 General information. 63
7.2 Left: On ground tests scale rating. Right: In air tests scale rating. 67
7.3 Left: On ground tests scale rating. Right: In air tests scale rating. 67
7.4 Summarizing of application features usability. 68
7.5 Summarizing of overall GUI appearance. 68

Aviation Acronyms and Abbreviations

AC	Advisory Circular
AMC	Acceptable Means of Compliance
AOM	Aircraft Operating Manuals
ATC	Air Traffic Control
CAA	Civil Aviation Authority
CAS	Crew Alerting System
CCD	Cursor Control Device
EASA	The European Authority for Aviation Safety
ECL	Electronic Checklist
EFB	Electronic Flight Bag
EICAS	Engine Indicating and Crew Alerting System
FAA	Federal Aviation Administration
FCOM	Flight Crew Operating Manuals
HUD	Head-Up Display
ICAO	International Civil Aviation Organization
IFTB	In-Flight Turn Back
MFD	Multi-function Display
OBDS	On Board Data System
PFD	Primary Flight Display
QRH	Quick Reference Handbook
SARP	Standards and Recommended Practices
SOP	Standard Operational Procedure
TCAS	Traffic Collision Avoidance System

Chapter 1

Introduction

Only 117 years divide generation from the first flight performed by the Wright brothers in 1903 until the state of the art aircraft that fly above the heads of millions of people every day. From the desire to keep the aircraft in the air for only a few minutes, people are now in that time where airlines compete with each other in the length of the longest flight. The rapid development over the past years led to an increasing pilot workload to ensure the safety of air traffic. To compensate for the potential risk of failure and ensure completeness and consistency during carrying out the task, the checklist was created. This thesis focuses on improvements of electronic checklist application of modern flight deck with state-of-the-art technologies, resulting in the software application which takes advantage of augmented reality based assistance. The main improvement lays in a visualization that provides assistance to the pilot in order to go through the checklist step by step. For the active step, related flight instrument on the flight deck will be identified and tagged, using the proper visualization.

Statistically, a significant percentage can be attributed to the human factor. The pilot is responsible for many tasks during all flight phases, including pre-flight preparation, in order to ensure the flight efficiency and safety of all participants in the air traffic. Several factors such as mental or physical condition can influence human performance. In addition, memory limitations lead to insufficient knowledge and familiarity with the full set of checklists, especially less frequent ones.

This topic has been chosen because it combines a personal interest in aviation and in augmented reality. Generally, technology has improved safety in aviation in many ways, such as guiding the pilot for landing in low visibility or warnings about changing weather during the flight. Augmented reality is already spreading to industries as an effective assistance in performing a defined sequence of actions. The use of augmented reality could have a significantly positive effect on ensuring that all checklist actions are performed as required and in a timely manner.

The thesis itself is divided into 8 chapters. The second chapter introduces augmented reality field covering definition, methods and algorithm used in augmented reality. The chapter is concluded by available hardware devices overview and frameworks overview, required for development in this field. The third chapter provides an insight into the state of the art trends in augmented reality and is concluded by an brief overview of the checklist types used in aviation. The fourth chapter provides insight into regulation authorities, requirements, and state of the art electronic checklist solutions. The following, fifth chapter, describes state of the art analysis and proposed solution. Chapter sixth is focused on imple-

mentation based on selected technologies. The thesis is concluded by chapters describing system evaluation and outlook for potential future improvements.

Chapter 2

Augmented Reality

This chapter covers basic concepts of augmented reality such as a definition of augmented reality itself as well as description of mixed reality continuum. The following part of the chapter deals with a basic description of methods and algorithms used in the augmented reality field. It does not contain encyclopedic knowledge but only knowledge closely related to the topic of this thesis. The chapter is concluded by sections dedicated to required hardware equipment and software frameworks and software development kits overview necessary for engaging in AR applications development.

2.1 Augmented Reality Concept

Augmented reality (AR) and virtual reality (VR) are related terms sometimes mixed together [22]. Both terms along with reality and augmented virtuality can be placed under mixed reality. The mutual relationship between the mentioned terms is shown in Figure 2.1. Mixed reality represents the blend of physical and virtual worlds includes both real and virtually generated object mixed together to generate a realistic environment.

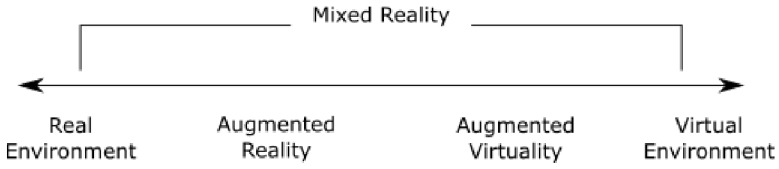

Figure 2.1: Mixed reality continuum[1].

The virtual world definition does not contain any restrictions regarding what a user can do or experience. The concept is based on the user's location in a completely computer-generated 3D computer model or simulation, which can be achieved using large fully and semi-immersive projection-based systems such as computer-assisted virtual environments (CAVEs) in domes or stereoscopic head-mounted displays (HMD). VR gaining more attention and popularity with the support of the game industry which introduced HMD gaming devices from which may be mentioned the Oculus Rift or HTC Vive.

The aim of the AR is embed digital information into the physical environment in the way it becomes a part of it from a user's perception. Specialized publications introduced a

[1]Taken from [22]

several definitions of AR but the most widely accepted one is ascribed to Ronald T. Azuma [22]. Based on Azuma's definition, a system could be called as AR if it meets following three characteristics:

- the system combines real and virtual information,
- the system is interactive in real time,
- the system is registered in 3D.

Azuma's definition of AR requires real-time control and spatial registration while a particular type of display technology is not specified.

2.2 Methods and Algorithms Used in AR

Each AR application's core is based on three important components known as calibration, tracking, and registration [22]. These components take care of measuring and properly aligning the virtual object into the real world. The following text is devoted to a brief explanation of these terms, description of camera calibration based on intrinsic and extrinsic parameters and distortions that arise during manufacturing, and is concluded by general tracking concepts.

2.2.1 Object measurement and alignment

Dynamic measuring of the AR system for the purpose of obtaining a relative three-dimensional position or the six-dimensional pose of real objects is known as tracking [22]. The pose represents a position and orientation of the AR display and needs to be continuously updated according the to real-time operation.

The goal of calibration is to determine parameters obtained from a reference device that will be used to calibrate the AR device. Especially important is calibrate AR device used for tracking. It is possible to replace the reference calibration device with a reference value or known coordinate system if the purpose of the calibration allows it. Calibration is performed at discrete times and it is referred to as offline adjustment of measurements. Calibration interval depends on the measurement system. Some devices can be calibrated only once during or after manufacturing while the others need to be calibrated concurrently within tracking or before the beginning of the process.

Registration can be understood as an ability to render virtual objects into the real world with proper alignment of coordinate systems. This process requires tracking of the user's head if the goal is to obtain a dynamic registration and calibration of a tracking system to create a common coordinate system between virtual and real objects for obtaining a static registration.

2.2.2 Camera model and basic projective geometry

A pinhole camera model is used as a standard camera abstraction of a common physical camera in computer graphics and vision [8]. The pinhole camera model describes a perspective projection of a 3D point in object space to a 2D point in image space. As it is shown in Figure 2.2, only light rays that intersect a particular point in space are released through a pinhole.

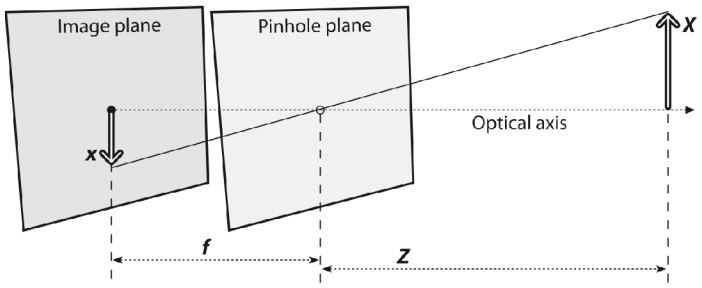

Figure 2.2: Pinhole camera model[2].

In the idealized case, the distance from the pinhole to the screen is precisely the focal length and it is expressed by equations (2.1), (2.2). The resulting image is formed using obtained rays onto image plane. The f is referred as the focal length of the camera, Z is the distance from the camera to the object, X is the length of the object, and x is the object's image in the imaging plane.

$$\frac{-x}{f} = \frac{X}{Z} \tag{2.1}$$

or:

$$-x = f\frac{X}{Z} \tag{2.2}$$

The pinhole camera model can be adjusted in accordance with the given equation to equivalent but mathematically easier form. As it is shown in Figure 2.3, the image plane and the pinhole are swapped which causes right-side up object appearance. The image plane performs the function of the projection screen, therefore a point Q = (X,Y,Z) is projected onto the image plane by the ray passing through the center of projection to the point q = (z,y,f) on the image.

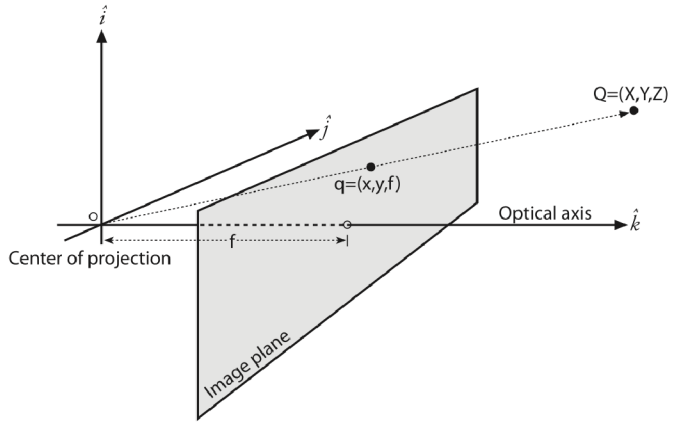

Figure 2.3: Point projecting through image plane[2].

[2]Taken from [8]

The point in the pinhole is reinterpreted as the center of projection and the point at the intersection of the image plane and the optical axis is referred to as the principal point [8]. New frontal image plane contains the image of a distant object with the same size as it was before rearrangement. The equation (2.3) express the similar relationship of the triangles that is more evident. The object image is no longer upside down so the negative sign is gone.

$$\frac{x}{f} = \frac{X}{Z} \tag{2.3}$$

The equations (2.4) describes the projection of point Q from the physical world onto a screen at some pixel. Parameters c_x and c_y for modeling a possible displacement of the center of coordinates on the projection screen due to the fact that the center of the chip is not commonly on the optical axis. The f_x and f_y indicate focal lengths necessary due to a typically low-cost rectangular imager instead of the squared one.

$$x = f_x(\frac{X}{Z}) + c_x, y = f_y(\frac{Y}{Z}) + c_y \tag{2.4}$$

Finally, equations (2.5), (2.6) describes calculation of the projection of the point in the physical world into the camera. Here, q is a point positions in homogeneous coordinates, Q is a point position in three-dimensional space, and M is a camera intrinsic matrix. The intrinsic matrix contains a parameters fixed for a specific camera.

$$q = MQ \tag{2.5}$$

or

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.6}$$

The ideal pinhole camera model is useful for some of the 3D geometry of vision but is not very suitable in practice.However, the pinhole is able to let only a small amount of light which leads to very slow imaging. To meet the requirement of a faster rate, much more light must be gathered over a wider area and bent to converge at the point of projection. This requirement is accomplished using a lens but this solution brings a negative effect known as distortion.

2.2.3 Lens Distortions

The distortions arise during manufacturing process and both of them can be mathematically removed with calibration [8]. The two main types of lens distortions are shown in Figure 2.4. The first of them is known as a radial distortion, which is a result of the shape of the lens. The second one is called tangential distortions arise from the assembly process of the camera.

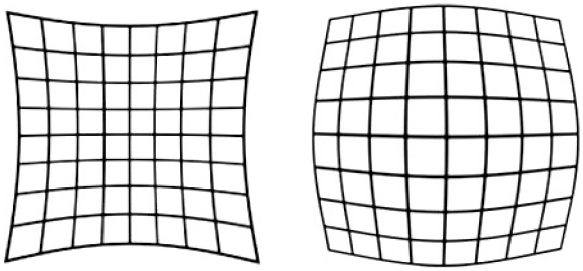

Figure 2.4: Left: Radial distortions. Right: tangential distortions[3].

The equations (2.7) describes rescaling point on the imager to resolve the radial distortion. Radial distortion occurs when rays farther from the center of the lens are bent more than those closer in and it is a source of the „barrel“ effect. The distortion is zero at the center of the imager but the closer to the periphery the more it increases.

$$\begin{aligned} x_{corrected} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\ y_{corrected} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \end{aligned} \tag{2.7}$$

Values k_1, k_2, and k_3 represents radial distortions. Distortion terms k1 and k2 are used for cheap web cameras but cameras such as fish-eye lenses create high distortions so a third radial distortions term k3 is used. Values (x, y) represents the original position of the distorted point on the imager, ($x_{corrected}$, $y_{corrected}$) is the new position after correction, and r is a distance from the optical center of imager.

In general, tangential distortions effect arise because the lens are not exactly parallel to the image plane. The equations (2.8) expressing rescaling a point with the tangential location on the images.

$$\begin{aligned} x_{corrected} &= x + [2p_1 y + p_2(r^2 + 2x^2)] \\ y_{corrected} &= y + [p_1(r^2 + 2y^2) + 2p_2 x] \end{aligned} \tag{2.8}$$

Values p_1 and p_2 represents tangential distortion. The meaning of other values does not change. Introduced equation used for correction of radial and tangential distortions required five coefficients k_1, k_2, k_3, p_1, and p_2 and all of them must be known. Many other kinds of distortions occurs in imaging system but they typically have lesser effects than the two introduced.

2.2.4 Camera calibration

Camera can be calibrated with use of any appropriately characterized object known as a calibration object. Some calibration methods rely on three-dimensional objects such as a box with markers, while the other one uses flat regular pattern such as a chessboard. According to some sources, it is practical to deal with chessboard patterns as it is easier than obtain, store and distribute precise three-dimensional calibration objects. The chosen pattern also ensures no bias toward one side or the other in measurement.

[3]Taken from [6]

Mapping a point on a two-dimensional planar surface can be expressed in therm of matrix multiplication (2.9). To this description, it is a necessary to express the viewed point Q and the point q on the imager with homogeneous coordinates. The process can be imaged as a mapping of a chessboard to the imager of our camera and it is a typical example of planar homography in computer vision.

$$\tilde{q} = sMW\tilde{Q} \tag{2.9}$$

Equations (2.10), (2.11) describes definition of W. Here, s is represents a scale factor, W describes extrinsic camera matrix, $\tilde{Q}$ 3D world coordinates of a point and $\tilde{q}$ represent the coordinates of the projected 3D point.

$$W = \begin{bmatrix} R & t \end{bmatrix} \tag{2.10}$$

or

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = s \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.11}$$

On the basis of the above, a camera can be calibrated for a camera intrinsics and extrinsics, which includes translation and rotation, and distortion parameters. The result of calibration without distortion is defined via equation (2.12). Both parameters can be solved separately because intrinsics together with extrinsic parameters are associated with three-dimensional geometry of the planar object and distortions are associated with the two-dimensional geometry. The distortion parameter can only be computed after the intrinsic camera parameters are known.

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = (1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \begin{bmatrix} x_d \\ y_d \end{bmatrix} + \begin{bmatrix} 2p_1 x_d y_d + p_2(r^2 + 2x_d^2) \\ p_1(r^2 + 2y_d^2) + 2p_2 x_d y_d \end{bmatrix} \tag{2.12}$$

Here, (x_p, y_p) represents point position in perfect pinhole camera and (x_d, y_d) represents a distorted position of the point.

2.2.5 Tracking

Spatial sensor arrangement is one of the tracking technology [22]. The technology is divided into two types known as outside-in and inside-out and both types are shown in Figure 2.5. The distinction between them is based on the position of used sources and sensors.

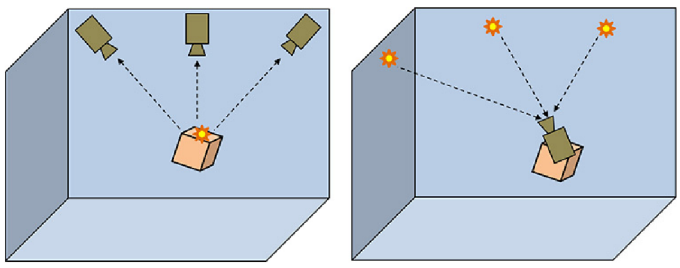

Figure 2.5: Left: Outside-in tracking. Right: Inside-out tracking[4].

.

The outside-in tracking refers to system where sensors are mounted stationary in the environment. On the opposite side, the inside-out system connects sensors with mobile or head-mounted device. Along with stationary tracking system and mobile sensors, the most promising one is optical tracking which comes from the fact that even inexpensive cameras provide rich measurement.

Optical tracking

Computer vision algorithms able to scale the results without improving the camera system relies on computing power [22]. To choose appropriate tracking techniques, varying circumstances need to be considered such as whether a digital reference model is created or if an appropriate place for artificial fiducials available.

If a digital reference model exists prior to starting the tracking process and is used for tracking, such a concept is known as a model-base tracking. On the other side is a model-free tracking. While it may seem that a model is not necessary for model-free tracking, the name is slightly misleading because these systems work with temporary models obtained on the fly in the course of the tracking.

Other terms associated with computer-based tracking are known as marker-based tracking and markerless tracking. The chosen technique determines the method of classifying the tracking target. In ideal cases, the real-world environment would not be instrumented in any way but with respect to the computational complexity, simpler and possibly more robust tracking algorithms requires artificial features for tracking, called a marker or fiducial. If no markers are used, the tracking target is based on natural features in the natural environment. This technique is known as markerless tracking.

Markers

Markers are defined as known patterns or trackable shapes attached to the target object [22]. The most popular markers design is shown in Figure 2.6. The design based on circular or square shapes of easily detectable black and white designs with different patterns or barcodes due to a good contrast.

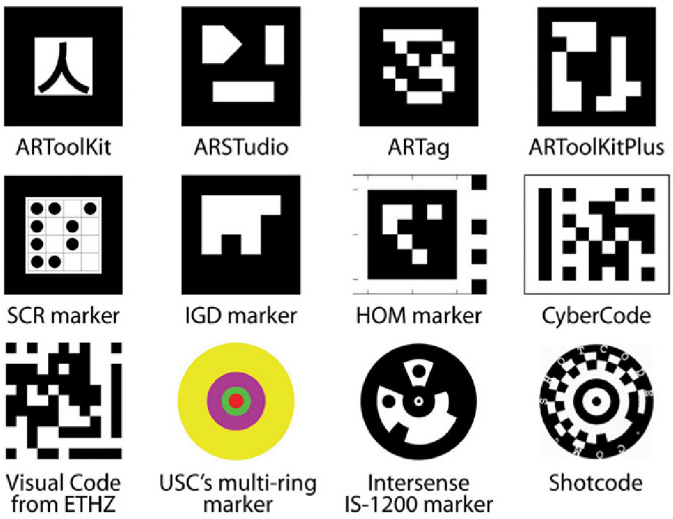

Figure 2.6: Square/circle markers are the most popular ones[5].

[4]Taken from [22]

Marker tracking solution exists for more than 15 years whereby the market offers even solid open source solutions. Especially tracking a square black-and-white fiducial markers is popular because is computationally inexpensive together with useful results even when working with a poor camera. Squares yield four corner points whereby theoretically minimum of three points are required to recover a full 6DOF (Degree of Freedom) pose and the fourth point is present only for practical implementations purposes to obtain a unique solution. it is crucial to properly identify all corners. In this context, the rotation-invariant pattern is added inside the marker shape to distinguish multiple markers and establish their orientation.

Natural features

Some environments or circumstances may not be ideal for placing markers [22]. In these cases, the tracking is based on naturally occurring features, typically requiring better image quality and more computational resources. The features are represented by interest points that should be easily found on a target object. To make the tracking stable, the irregular surface texture is practically required. Because this condition may not always be met, some tracking algorithms are based on edge features. Figure 2.7 shows naturally occurred features such as edges or interest points.

Natural features tracking pipeline consists of five stages known as the interest point detection, descriptor creation, descriptor matching, perspective-n-point camera pose determination, and robust pose estimation. The interest point detection is based on algorithms known as the Harris corner detector, the difference of Gaussian, and the FAST (features from accelerated segment test) detector. The most popular descriptor is knows as SIFT (Scale Invariant Feature Transform). Along SIFT, from other popular descriptors could be named SURF (Speeded Up Robust Features), and BRIEF (Binary Robust Independent Elementary Features).

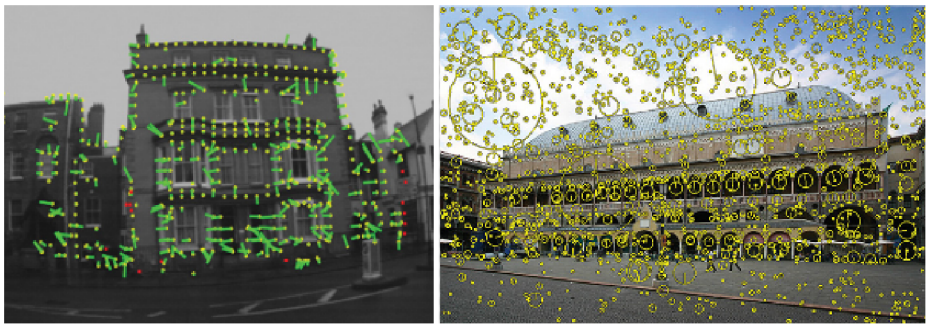

Figure 2.7: Left: Tracking along strong edges. Right: SIFT interest points.[5]

Both marker-based and natural features tracking can be used in connection with model-based tracking. The only differences are in the order of existence digital models such as printed marker and physical object designed for distinction and recognition purposes. In the case of a combination of a marker with model-based tracking, the digital model is created before the tracking phase and physical object is manufactured to match it. With the natural features approach, the order is reversed and the tracking process uses a scanner to obtain a digital model which is supposed to match already existed physical object.

[5]Taken from [22]

2.3 Augmented Reality Devices

AR devices can be categorized according to several factors such as the augmentation method into the optical see-through, the video see-through, and the spatial projection [22]. As it is shown in Figure 2.8, the selected categorization factor is based on distance from the eye to the display, .

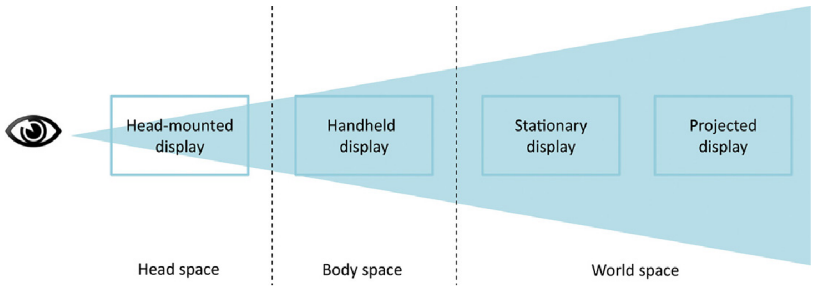

Figure 2.8: AR displays categorization[6].

The selected categorization creates four groups of display types designated as the head-mounted displays, the handheld displays, the stationary displays, and the projected displays.

2.3.1 Head-mounted Displays

Head-mounted displays (HMD), also known as head-worn augmented reality displays, can be divided into two main categories based on user views of the real world [22]. The main difference between these categories are shown in Figure 2.9.

Optical See-Through displays mediate AR experience by looking directly through monocular or binocular optical element to the real world. The optical elements contains holographic wave guider or other system elements that enables displaying graphical elements and overlay the real world by them.

Video See-Through displays concept presents AR experience by capturing real-world view on one or two video cameras mounted on the front of the display and combining the record with computer-generated imagery.

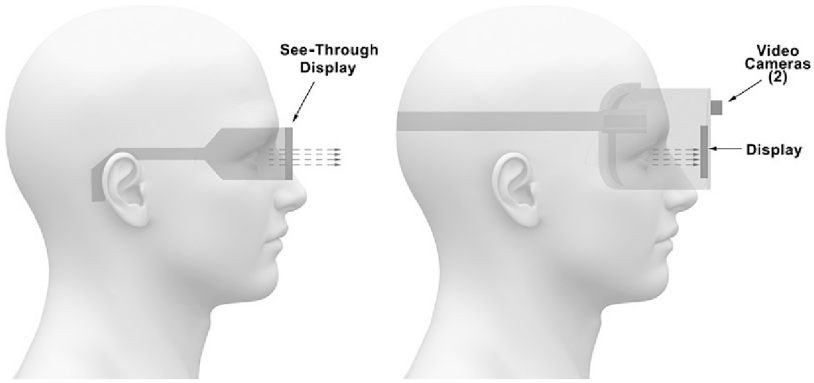

Figure 2.9: Difference between Optical See-Through and Video See-Through displays[7].

[6]Taken from [22]

HMD display design depends on device purpose and can be divided into two categories:

- Helmet-mounted displays,
- Smart glasses.

Helmet-mounted devices, commonly known as HMD, are designed for pilots, or firefighters whose work task already requires wearing a helmet.

Smart glasses are commonly designed as an ordinary glasses [20]. Some devices could use a clip-on design but this style is denoted by the so-called see-around effect. The display is out of the main field of view and the spatial arrangement is inappropriate for see-through and is much more suited for displaying text information. The approach requires careful design with respect to ergonomics, frame weight, adjusting to different head sizes, and ensuring sufficient amounts of airflow near the head.

F-35 Gen III

F-35 Gen III Helmet Mounted Display System (HMDS) is shown in Figure 2.10. It is a typical representative of HMD's in aviation [20]. The HMDS is provided by Rockwell Collins ESA Vision Systems, LLC, a joint venture between Elbit Systems Ltd. of Israel, through its U.S. subsidiary Elbit Systems of America, of Fort Worth, Texas, and Rockwell Collins.

Figure 2.10: F 35 Gen III HMDS[8].

The F-35 binocular 30-by-40-degree wide-field-of-view with 100 % overlap HMDS provides a real-time projected enhanced situational awareness for critical flight and mission information overlaid onto the view of the outside world. Other key benefits includes built-in nigh vision capability together, weapon targeting, and target verification.

Microsoft Hololens 2

Hololens 2 are shown in Figure 2.11. The headset offers HMD for AR experience through see-through holographic lenses with 2K resolution and display optimization for 3D eye position, developed by Microsoft [13]. The headset is equipped by several sensors including 4 visible light cameras, 2 IR cameras, 1-MP time-of-flight (ToF) depth sensor etc. that supports the head and eye tracking, dept, and inertial measurement unit (IMU). According to the manufacturer, the device can last for 2-3 hours of active use. Hololens 2 provides currently the best FOV on the market, 43○ horizontally and 29○ vertically.

[7]Taken from [6]

[8]https://www.redimec.com.ar/contenido/productos/pdf/1426684093$_1$*.pdf*

Figure 2.11: Microsoft Hololens 2[9].

Microsoft Hololens allows interaction with the virtual world in the form:

- selecting holograms leading the cursor by a head movement,
- selecting holograms, items, and apps opening using body gestures,
- apps navigation and controlling using users voice.

The device is not a mainstream product yet, it is targeted to professionals such as engineers, industrial designers, mechanics, and other professions.

Magic Leap One
Magic Leap One are shown in Figure 2.12. The hardware consists of two parts, the headset and a processing unit with processing circuitry for analyzing reality and 3D object rendering, called the Lightpack [13]. The headset feels bulkier than the Hololens 2. Furthermore, the appearance is another disadvantage.

Figure 2.12: Magic Leap One[10].

Magic Leap One also falls behind Hololens 2 in the FOV area with 40x30 degree field of view. According to some sources, the device is intended for application developer.

Focals by North
As it is shown in Figure 2.13, the Focals by North looks like ordinary glasses from the front [13]. The battery, the projectors, and the rest of the electronics are attached on the sides making them thicker. The device is designed to display all kind of notifications from the user's phone through a laser projector placed on the inside of the right temple.

[9]https://www.microsoft.com/en-us/hololens/hardware
[10]https://www.aniwaa.com/product/vr-ar/magic-leap-one/

Figure 2.13: Focals by North[11].

The FOV is 15 degrees diagonally which is sufficient for their purpose. The Focals by North can be synchronized with Android or iOS and can show information and notifications from a variety of applications like WhatsApp, Facebook Messanger, Google Maps, Calendar, Spotify, etc. The smart glasses can be controlled by voice through integrated Alexa but real control comes from a finger ring with a tiny four-direction joystick. It is designed for regular users.

Google Glass Enterprise Edition 2

Google Glass Enterprise Edition are shown in Figure 2.14. The concept is known as the representative of the clip-on design concentrated on user's workflow experiences support [13]. The hands-free concept is valuable during maintenance because it allows users to focus on the job in front of them while they have simultaneously access to a manual.

Figure 2.14: Google Glass Enterprise Edition 2[12].

The device's equipment include 820mAh battery with fast charge, Multi-touch gesture touchpad, and power-saving features containing on head detection sensor and Eye-on screen sensor. The device is similar to Epson Moverio so it does not require large FOV. In difference with Moverio device, Google Glass puts only tiny square of 2D visuals on the right side peripheral field of users right eye. Google Glass Enterprise Edition is for business use only.

Vuzix Blade

Vuzix Blade are shown in Figure 2.15 and falls into the same category as Google Glass and Moverio and its purpose is to overlay information [13]. Unlike the others, the smart glasses

[11]https://www.bynorth.com/focals
[12]https://www.google.com/glass/tech-specs/

have a lens that is a screen. The device provides a 19 degrees diagonal FOV with a 480 x 480 resolution, which is quite enough based on a limited projection of 2D information like a mini-HUD.

Figure 2.15: Vuzix Blade[13].

It connects to Android or iOS to obtain information to be displayed. The Vuzix can be controlled with head motions or touch and can also record videos and take photos through its 8MP camera.

2.3.2 Handheld displays

Handheld displays are represented by mobile AR devices such as smartphones or tablet computers, as it is shown in Figure 2.16. The devices use the onboard cameras to mediate AR experience to the user.

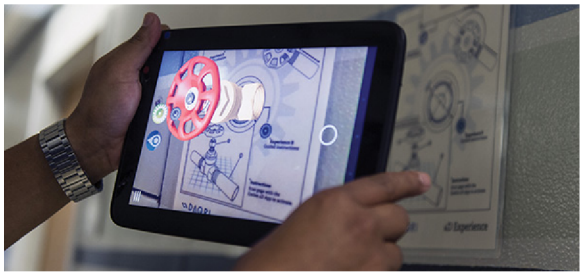

Figure 2.16: AR application used on tablet[14].

Based on the form of mediation, the systems can be classified as video-see through devices. The biggest advantage of a handheld AR device is battery life as well as availability.

2.3.3 Stationary displays

Stationary displays include desktop displays, mirror displays, display showcases, and window/portal displays [22]. The desktop display is shown in Figure 2.17. The concept of a virtual mirror is based on a front-facing camera. The picture of the user taken by the camera is reflected overt the vertical axis which creates an impression of looking into a mirror. The stationary configuration of the mirror display is called the virtual showcase. An observer is separated from the observed object by a semi-transparent mirror that combines a reflection of the observer with a computer-generated image and projects it on a mounted screen.

[13]https://www.vuzix.com
[14]Taken from [22]

Figure 2.17: HoloFlector by Microsoft Research[15].

Window and portal displays represent another similar set of component dependencies based on user tracking and positionally constant displaying.

2.3.4 Projected displays

Concept of projected display is shown in Figure 2.18. Projected displays are based on the spatial augmented reality concept [22]. View-independent spatial AR reflects projection directly from the surface of real objects that are changing their appearance.

Figure 2.18: View-dependent spatial AR[15].

View-dependent spatial AR requires the support of user tracking and active shutter glasses to be able to appear 3D virtual objects in space independent of the surface.

2.4 Frameworks and SDKs Overview

Several factor needs to be considered before choosing a suitable framework or SDK [21]. The list of factors may include a features that can be useful during development, support in the chosen development tool, platform support, license conditions or target device. Appropriate framework selection depends on consideration if the AR experience will be built using the native application or web application. The Web-based AR can provide an AR experience across all platforms, devices, and mobile OS without the need for additional installation in a native web browser environment.

[15]Taken from [22]

However, this approach encounters some challenges from which they can be named:

- Powerful computing capability is required for tracking and registration of AR system. The web-based solution provides limited computing capability.
- More appropriate use of cloud servers is providing more powerful computing causes network delay which is against requirement of real-time performance.
- Compatibility challenges due to the diversity of display platforms, operating systems, and data formats.

In view of the above and desired aim, the analysis will be focused on currently best-known frameworks and SDKs for AR native application environment.

ARKit
ARKit is a free framework for an augmented reality application development, released by Apple Inc. in 2017 [28]. The framework is designed to be used on iPhone and iPad hardware and requires A9 chip or later, utilizes VIO (Visual Inertial Odometry) and a minimum version of the operating system iOS 11. The VIO enables combination of Core Motion data with data obtained from camera sensor. This technology is able to track surrounding environment with significant accuracy. Application using this technology can detect a horizontal planes such as floors and tables as well as vertical planes such as walls.

The latest released version was introduced as ARKit 3. Compared to version 2, this version brings several new features from which could be named:

- front camera can be used simultaneously with back camera,
- iPhone X and higher use TrueDepth camera that allows to tracks up to three faces at one time,
- collaborative sessions between multiple users allow to build a collaborative world map,
- up to 100 images can be detected at one time.

ARCore
ARCore is a free framework introduced by Google for building augmented reality experiences. The framework is designed for Android phones running on Android 7.0 (Nougat) and later [17]. The platform uses different APIs, from which some of them are available across Android and iOS, though it essentially wraps ARKit on iOS. Integration of virtual content with the content of the real world is achieved using three key capabilities:

- motion tracking for allowing to understand and track phones position relative to the real world,
- light estimation for the current lighting condition,
- environmental understanding of the surface size and location detection. The detection works for all surface types in vertical, horizontal, and angled position.

The ARCore is based on identical concept as ARKit. Mobile device tracks position of some feature point and builds its internal understanding of the real world.

AR Foundation

AR Foundation is a free cross-platform AR framework that allows the development and deployment of Unity AR applications across smartphones and wearable AR devices [26]. According to the AR Foundation manual, the framework is based on several platform-agnostic interfaces for surfacing different types of information called subsystems. As it is shown in Figure 2.19, the functionality of each subsystem is implemented in other platform-specific AR packages. The packages are known as ARKit XR Plugin and ARCore XR Plugin and therefore, it is necessary to install at least one of them.

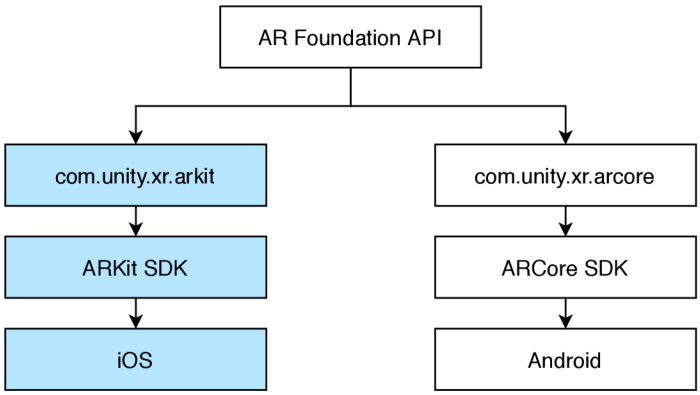

Figure 2.19: AR Foundation API architecture[16].

The framework is intended for cross-platform systems. Supported platforms are ARKit, ARCore, Magic Leap and HoloLens. Comparing ARKit and ARCore, not all features are available in both platforms. Currently, the framework large amount of cross-platform features, such as pass-through video, device tracking, raycast, plane tracking, face tracking, reference points, 2D image tracking, and point cloud detection. Table 2.1 provides a summarization of features that are platform-specific.

	ARKit	**ARCore**
3D object tracking	Yes	No
2D/3D body tracking	Yes	No
Human segmentation and occlusion	Yes	No
Collaborative participants	Yes	No

Table 2.1: Platform-specific features provided by AR Foundation [17].

To find correspondences between the real world the phone inhabits and the virtual 3D space, AR Foundation provides functionality to track images, planes, feature points, 3D objects, and human faces.

[16]https://www.viget.com/articles/cross-platform-ar-with-unity/
[17]https://unity.com/unity/features/arfoundation

Vuforia

Vuforia Engine is one of the best-known software platforms for AR development for smartphones, tablets, and smart glasses [29]. The platform supports AR application development for Andriod, iOS, and UWP (Universal Windows Platform) devices. The engine was integrated into Unity that supports a simple setup procedure for cross-platform development.

The core of the engine provides advanced computer vision functionality for object recognition and environment reconstruction. Objects recognition and tracking capabilities can be used on a several types of images and object among which may be appointed:

- model Targets for recognition based on pre-existing 3D models,
- image Targets for attaching content onto flat images,
- object Targets which are obtained by scanning an object with consistent shape and rich surface details,
- VuMarks which is a recognition feature allowing encode a range of data formats into customizable markers. VuMarks supports tracking and unique identification for AR applications,
- ground Plane for placing content on horizontal surfaces in the environment.

Table 2.2 provides a comparison of the frameworks and SDK described above in terms of the price, platform support and provided extensions and features.

	ARKit	**ARCore**	**Vuforia**
Type	Framework	Framework	SDK
Price	Free	Free	Free, Commercial
Platform support	iOS	Android, iOS	Android, iOS, UWP
Geo-location	No	No	No
Smart glasses support	Yes	Yes	Yes
Unity3D	Yes	Yes	Yes
SLAM	No	No	No
Cloud recognition	No	No	Yes
Tracking	Markerless	No	Marker-based
Image tracking	Yes	Yes	Yes
Planes tracking	Yes	Yes	Yes
Lighting	Yes	No	No
3D object tracking	Yes	No	Yes
Facial	Yes	No	No

Table 2.2: Comparison of ARKit, ARCore, and Vuforia AR SDK features [18].

Vuforia provides a free license key for development but deployment is conditional by purchasing one of the license plans.

[18] https://invisible.toys/best-augmented-reality-sdk/

2.5 Rendering Engines Overview

The leading game engines are known as Unity 3D and Unreal Engine [30]. Both engines offers the same sort of capability and functionality, such as cross-platform development of 2D and 3D video games or simulations, and possible AR development.

Unity 3D
The engine is provided by the Unity company [9]. The Unity 3D supports a scripting in C#, Boo, or JavaScript. Unity does produce high-quality visuals but unlike Unreal, it is more difficult to achieve this [30]. Unity was originally designed for low-end devices such as gaming consoles or smartphones. A powerful computing setup is not necessary even for complex projects. It is Unity is easier to learn and therefore may be a good choice for a single developer or a smaller team. Its asset store provides significantly more solutions.

Unreal Engine
This game engine is developed by Epic Games company [18]. The engine is written in C++ and supports developing using C++ or visual scripting language known as Blueprint. Unreal provides high-fidelity visuals and it is, therefore, appropriate to use it for projects requiring highly photorealistic assets [30]. Unreal provides a high processing power, which may not be suitable for lower-powered devices such as mobile phones. The engine is much more appropriate for massive teams of specialists that are dedicated to different parts of the process. Unreal is more appropriate for massive teams of specialists that are dedicated to different parts of the process.

Table 2.3 provides a summary of described specifications of Unity 3D and Unreal Engine.

	Unity 3D	**Unreal Engine**
Level of visuals	High-quality	As photorealistic as possible
Target devices	Game consoles, smartphones	Powerfull PC
Team size	Single developer, small team	Large team
Programming language	C#, Boo, JavaScript	C++, Blueprint

Table 2.3: Comparison of Unity 3D and Unreal Engine [19].

Unity 3D as well as Unreal Engine are free to use but both of them provide additional subscription plans.

[19] Taken from [30]

Chapter 3

Augmented Reality and Aviation

This chapter could be divided into two sections. The first section is focused on the augmented reality and highlights some of the ways in which AR applications are already used and future concepts. The latter section deals with basic introduction to the cockpit checklist as the goal of the thesis is to display it by an augmented reality application. This section covers basic knowledge about the cockpit checklist and deals with checklist concept and checklist types from the view of situation in which they are used.

3.1 Augmented Reality Solutions in Maintenance

This section will briefly introduce the usability of augmented reality in maintenance and training outside aviation [22]. This topic was briefly studied as a source of inspiration. Many professions spend an amount of time studying manuals and documentation in order to learn how to assemble, disassemble, or repair things. AR technology deployed on an appropriate headset is able to present instructions directly in the users field of view. This technology was initially created for maintenance workers.

Bentley Systems
Bentley System is a software development company providing software solutions to a wide range of professions such as architects, engineers, operators, and maintenance engineers [5]. As it is shown in Figure 3.1, the company presented a concept of using Microsoft HoloLens for a typical maintenance task in a plant environment. Using HoloLens leave the user's hands free for doing work. The solution contains the implementation of voice commands.

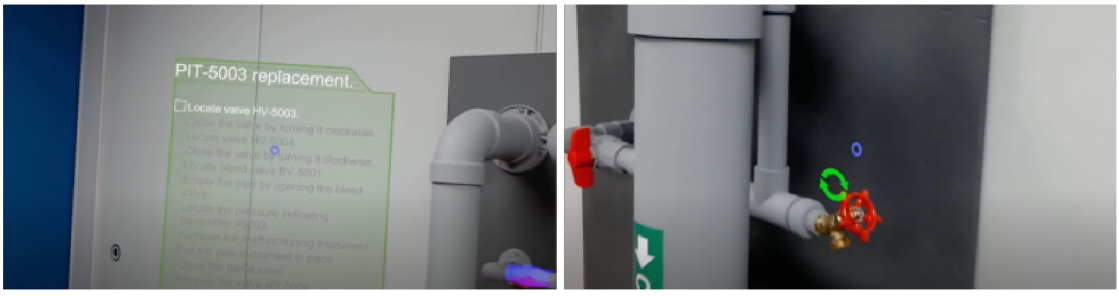

Figure 3.1: Left: A list of tasks is rendered directly in the user's view and can be moved to a different position. Right: Rendered AR elements have specific meaning[2].

The presented solution does not use markers to detect the valve position but the 3D model of the pipe setup, or a 3D mesh. The 3D model have to be aligned with the physical pipe setup. This process can be done only once as HoloLens is able to remember the mapped environment.

Hyundai
The different approach was proposed by Hyundai in 2015 [10]. The company presented an augmented reality owner's manual application called the Hyundai Virtual Guide. As it is shown in Figure 3.2, user can position the device camera over the part they want to learn about.

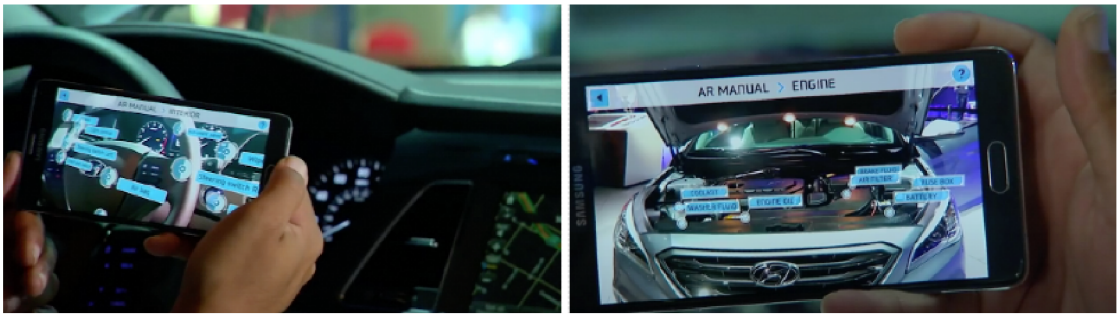

Figure 3.2: Hyundai Virtual Guide represents car owner's manual application[3].

The application was intended for smartphones or tablets with Android or iOS OS. According to the company, they are using 2D and 3D tracking technology. The application was able to recognize several major features of Sonata model and contains several videos as well as 3D overlay images. However, the application is currently not available in App Store.

3.2 Augmented Reality Solutions in Aviation

Applications for augmenting display technologies are widespread across different industries from which can be named gaming and entertainment, architecture and construction, science and engineering, health and medicine, education, aviation, telerobotics, and defense which is attributed to a solid impact of the augmented reality systems on human performance and cost-efficiency. Although the boom of AR technologies is evident mainly in recent years, published exploration can be traced back as far as the 1990s.

Airbus
Airbus has focuses on expanding assembling components area. For its purposes, Airbus developers also worked on application of simplifying the process of cabin seats assembling for Vuzix AR Smart Glasses [4]. Another project is related to the Airbus military division that developed a MOON (asseMbly Oriented authOring augmeNted reality) project in 2010 [23]. Both projects are shown in Figure 3.3.

[2]https://youtu.be/QTuKcm8s4QQ
[3]https://youtu.be/MDtxOmtVZGs

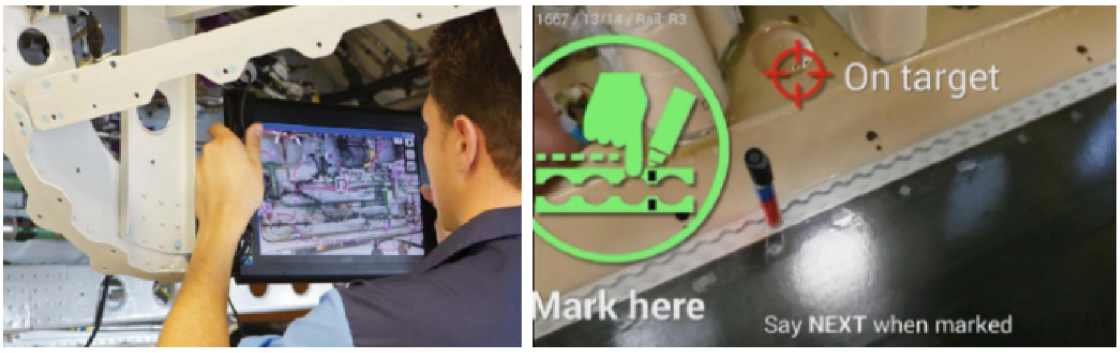

Figure 3.3: Left: MOON project[4]. Right: Cabin seats assembling[5].

MOON uses 3D information obtained from industrial Digital Mock-Up (iDMU) to assembly instructions generation with the support of AR technology. The technology was demonstrated on the electrical harness routing in the frame 36 of the Airbus A400M.

Atheer AR platform

Atheer is a California-based enterprise AR company founded in 2012 [15]. The company creates augmented reality training and safety solution for manufacturing, aviation and other sectors. The company has created Atheer AiR^{TM} Enterprise platform, shown in Figure 3.4.

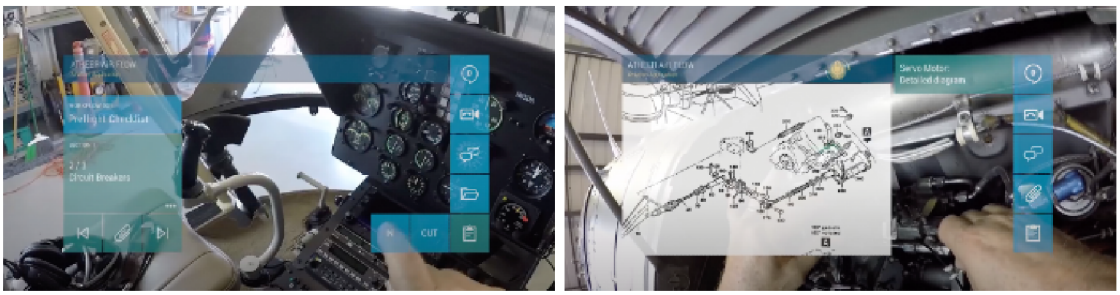

Figure 3.4: Atheer's AiR Enterprise is used to inspect a helicopter[6].

The platform is aimed at improving security, taskflow reporting and expanding the range of devices customers can use. The platform provides several features such as instant access to remote expert through videoconferencing, step by step guidance while performing task, full encryption, taskflow related features such as history, synchronization, reporting and auditing.

Aero Glass

Aero Glass, Inc. is oriented on the development that is supposed to display data such as altimeter readings, fuel pressure, heading, and oil temperature, and more [7]. These data are obtained from ADS-B and other instruments through the augmenting head-mounted display such as the Epson Moverio and Osterhut Design Group (ODG) R-7. In October 2016, Airbus Bizlab provided support for the company to transform its technology to become a business proposition.

[5]https://www.accenture.com/gb-en/success-airbus-wearable-technology
[6]https://youtu.be/CDLEgsxAKnM

As it is shown in Figure 3.5, raw data would be normally presented as 2D information on a multifunction display or presented statically on the printed chart.

Figure 3.5: Aero Glass presented concept of AR in cockpit[7].

The purpose of the software is to take the data as static and time-varying 3D phenomena and display them in a manner that depicts the actual spatial characteristics of the data. If possible, with respect to their precise position. The company was also successful in the European Union's Horizon 2020 research and innovation program when its head-worn display concept received funding.

Air New Zealand

Companies Air New Zealand and IT service-provider Dimension Data work together on beta-testing possibility of use Microsoft Hololens for cabin crew [7]. Figure 3.6 shows proposed concept of AR helper intended for a cabin crew. The concept is based on face recognition.

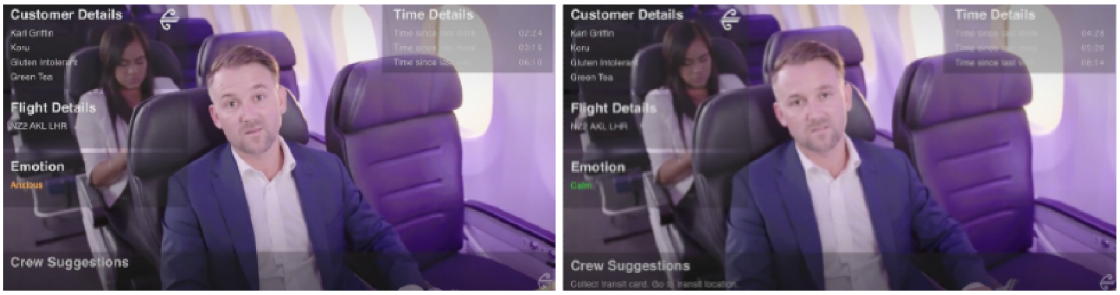

Figure 3.6: Air New Zealand AR helper for cabin crew[8].

The company's idea is based on a vision that flight attendants can take advantage of wearing a Hololens headset to display information about passengers such as flight details, time since last served and their emotional state.

[7]https://glass.aero

[8]https://www.aviationtoday.com/2017/08/ 24/9-companies-using-augmented-virtual-reality-aviation/

3.3 Cockpit Checklist

The primary objective is to ensure safety in air traffic. However, human errors may lead to incidents and accidents [24]. Generally, omission of required action or performance of inappropriate action are included in common factors that cause accidents. The data shows that these factors cause:

- 45 % of fatal approach and landing accidents,
- 70 % of all approach and landing accidents.

Based on statistics, the occurrence of accidents and occurrence of fatal accidents is different relative to the flight phase. As it is shown in Figure 3.7, the most devastating accidents occur during the final approach and landing phases. From the view of the frequency of fatal accidents, the sensitive phase of the flight is a climb. During this phase, the aircraft is taking place off the ground. If some fault occurred when the aircraft was at the gate and the fault was not detected, it may become apparent during climbing. Depending on the fault severity, the crew may perform an In-Flight Turn Back (IFTB) maneuver. The aircraft altitude could be to low, which makes the maneuver difficult.

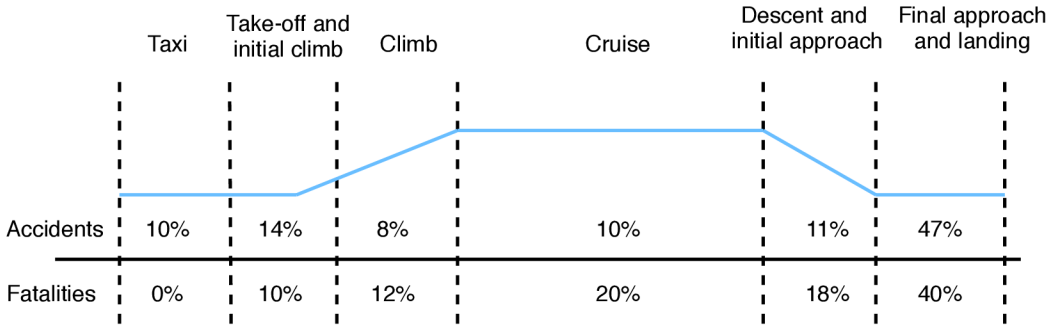

Figure 3.7: Differences between flight phases and the occurrence of accidents[9].

The incidents and accidents can be prevented by strict adherence to standard operating procedures (SOPs) which includes checklist. The flight deck checklist constitute tools that support crew airmanship and ensures proper airplane configuration for any given flight phase with special focus on critical phase. Almost every phase of the flight is assigned to a sub-divided part of the complete checklist containing phase-specific tasks. The complete checklist represents the basis of procedural standardization in the cockpit. As suggested in the chapter description, checklists are categorized depending on the situation of use and depending on the method of execution.

3.3.1 Checklist concepts

Two types of documents used in referencing of checklist term are known as follows [24]:

- Challenge and response checklist,
- Read and do list.

[9] https://www.1001crash.com/index-page-statistique-lg-2-numpage-3.html?fbclid=IwAR3Rh-5PfEp1G4fKGq$jiotJg7PWeeq40PSUwvYEa38CrxfdGo3vDPfRRzk$

In reference to the situation of use, the checklist procedures are recognized as follows:

- Normal procedures,
- Non-normal procedures including abnormal and emergency procedures.

Both normal and non-normal checklists are generally bundled in an easy-to-use Quick Reference Handbook (QRH).

Normal checklist
The normal checklist depicts a set of tasks ensuring that the flight crew will properly configure the safety critical aspects of the systems and aircraft configuration in each and every flight and maintain the required level of quality throughout the flight [12]. The checklist is based on the challenge and response concept, the tasks are performed from memory and follow a cockpit flow pattern. After completion, the pilot passes the entire checklist and verify that all required steps were performed. Key elements for ensuring safety are based on timely and effective completion. The methods of conducting the checklist varies from airlines to airlines, nevertheless most checklist formats follow the same strategy:

- action that have to be done is read or heard,
- correct setting or executing is verified,
- it is responded to the outcome of performed action.

If the cockpit is equipped by electronic checklists, the system may automatically detect if the action was performed, which leads to erasing the action from the list or changing the color of the action.

Non-normal checklist
Abnormal and emergency checklists are usually handled by the read-and-do concept as the challenge and response concept is not suitable [24]. These items are related to actions that must be accomplished immediately when a critical situation occurs. The situations are carried out from the memory and includes engine failure, loss of pressurization, or fire. Once the situation is stabilized, action taken is confirmed by reference to EAC (Emergency or Abnormal Checklist).

3.3.2 Checklist devices

Checklist variants have developed all over the years and technological advances had also an impact on it. In the following text, each checklist type will be briefly discussed from perspective of its advantages and disadvantages.

Paper checklist
Paper checklist is commonly used device [12]. Particular set of tasks can be printed on paper card and held by pilot or glued to the instrument panel. The checklist can be written on a placard and attached to the yoke, as it is shown in Figure 3.8. Hand held checklist brings several disadvantages. Missing pointer that distinguishes between accomplished and non-accomplished tasks may be included among them.

Figure 3.8: Checklist attached to yoke in Boeing 737[10].

The other ones are the lack of a memory system to store unaccomplished tasks, and occupying one hand when completing a set of tasks, which lowers pilots mobility. Some of the disadvantages can be eliminated by attaching checklist to the yoke. Construction design includes yellow pointer while both hands are free and the checklist is in a suitable visual position.

Scroll checklist

As it is shown in Figure 3.9, construction of the scroll checklist is based on two reels and a strip of paper inserted between them [12]. This construction is embedded after the window with a lubber line. The process of passage consists of steps when pilot reads the current task, completes it and rotates the reels to position of the next task on the lubber line. This system is commonly used in the United States Air Force (USAF) transport aircraft.

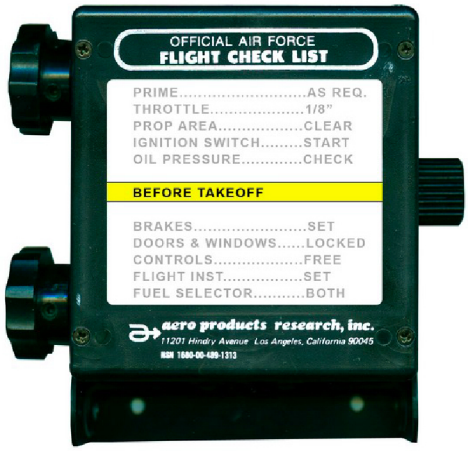

Figure 3.9: C-46 Commando aircraft scroll checklist[11].

[10] http://flightassurance.net/blog/wp-content/uploads/2015/09/EFB-Meets-The-Electronic-Checklist.pdf

[11] http://www.usmilitariaforum.com/forums/index.php?/topic/303074-c-46-commando-aircraft-scroll-checklist/

Implicitly included pointer is the main advantage of the scroll checklist. As mentioned in the case of paper checklist, the lack of memory persists. An added disadvantage is a relatively small size and orientation of the scroll checklist that makes it harder to read it for the pilot as the construction is mounted on copilot's side.

Vocal checklist

Vocal checklist is constructed as an audible generated checklist calls unit which can be pre-programmed by manufacturer or the user [12]. The device is equipped with rotary switch and proceed and acknowledged by push button mounted on the yoke. Rotary switch is used for selecting different task from normal and abnormal checklist. The acknowledge button is used for confirmation of task accomplishment. The proceed button can be used in two different use cases. First use case describes situation when the proceed button is pressed right after pressing the acknowledge button. In this case, it triggered the generation of next task. The second one describes situation when the acknowledge button was not pressed and pilot wants to heard the current task once again. Vocal checklist also allows to skip the task and safely move it to the bottom of the list to recall it later. The main disadvantage lies in the ability to mask audio checklist and blend it into cockpit communications, and vice versa.

Mechanical and electromechanical checklist

Concepts of a mechanical and electromechanical checklist are similar [12]. Construction of mechanical checklist is based on several plastic slides, as it is illustrated in Figure 3.10. They are moved to cover the accomplished tasks nomenclature so this way, the system highlights the non-accomplished tasks. An electromechanical checklist is constructed by small panel consisted of an internally highlighted list of tasks and toggle switch mounted alongside of the tasks.

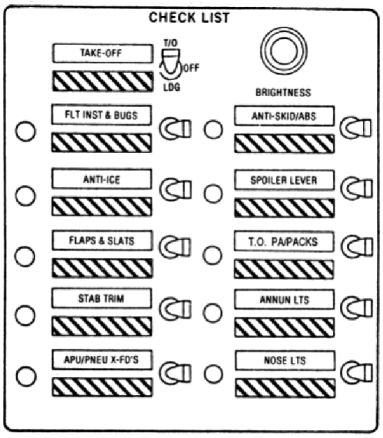

Figure 3.10: Mechanical checklist[12].

The switch is turned off when the task is completed. This principle provides quick visual feedback of accomplished and non-accomplished tasks. Only one major U.S. carrier uses this device for only BEFORE TAKEOFF and LANDING checklists purposes. The rest of tasks are performed from a printed paper card.

[12]http://hdl.handle.net/2060/19910017830

Computer-aided checklist

The computer-aided checklist appeared with the introduction of alphanumeric and graphic displays in the cockpit [12]. According to the FAA (Federal Aviation Administration), the electronic checklist is better than the paper one. Benefits are therefore undeniable, however, these checklists are associated with several disadvantages [16] :

- limited display and font size,
- early MFD CRTs have lower alphanumeric quality,
- distance between CRT and pilot's eyes is non-adjustable,
- expensive updating the the of the normal checklist,
- valuable CRT display real estate is consumed.

In response to the disadvantages of a computer-aided checklist displayed on MFD, the On Board Data System (OBDS) Electronic Checklist (ECL) emulator was developed. Both ECL types are shown in Figure 3.11. The OBDS contains a well-designed normal procedures checklist. OBDS solves the first three objections from the above list by employing it on an Electronic Flight Bag (EFB) or on an Apple iPad, and also conflicts that appears by consuming the Primary Nav Display (PND) with other ways of use such as TCAS, Weather Display, or Terrain while retaining the OEM support for Abnormal and Emergency procedures.

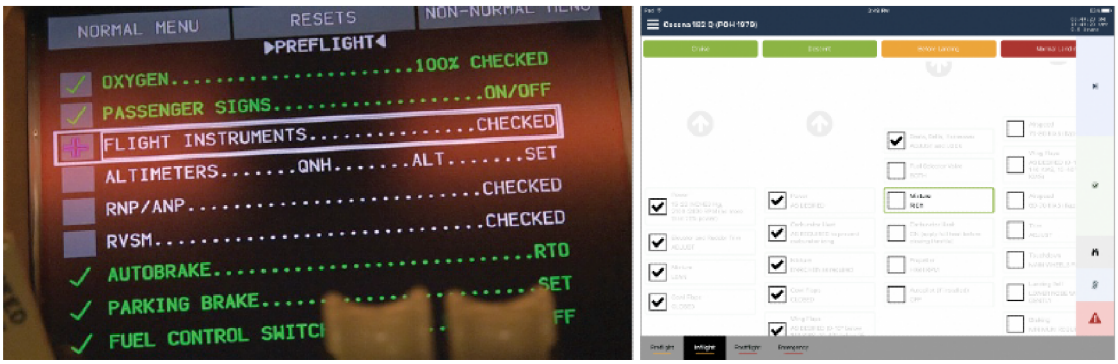

Figure 3.11: Left: Boeing 777 electronic checklist[13]. Right: Application on iPad[14].

[14]https://clairetoland.com/over-flowing-dont-undermine-your-checklist/
[14]https://ipadpilotnews.com/2017/08/miracheck-brings-audio-checklists-ipad/

Chapter 4

State of the Art Solutions of Electronic Checklist

This chapter's intention is to briefly overview currently used electronic checklist solutions. The presented applications can be divided depending on the environment they are used in. The first set is dedicated to the transport category aircraft and the second one consists of applications intended for mobile devices. Although electronic checklists have been around for decades, this feature was traditionally found in transport category airplanes with high-end avionics. Each available feature is tested in detail and certified. The whole process is subject to regulations and standards of certification authorities. The rapid development of technologies is changing the way of how modern avionics systems provides information to the pilots. Pilots from general aviation can fly with a tablet and take advantage of electronic checklists.

4.1 Regulatory Requirements

An aircraft is ready to fly after obtaining a Certificate of Airworthiness [11]. Aviation standards are defined at high level by specialized agency of United Nations, known as ICAO. ICAO is abbreviated from International Civil Aviation Organization. The organization covers 191 member states and several global aviation organizations. Together, they cooperate on developing international Standards and Recommended Practices (SARPs).

Each nation or commonwealth of nations has the local Civil Aviation Authority (CAA). National civil aviation regulations defined by CAA and refers to SARPs. Related CAA in our environment is known as the FAA of the USA, and the EASA (European Aviation Safety Agency) of the EU countries. The organizations actively promote mutual cooperation so the FAA Advisory Circular (FAA AC) and EASA Acceptable Means of Compliance (EASA AMC) guidance are similar.

In the interest of safety, the certification ensures that aircraft is designed and constructed in compliance with the appropriate airworthiness requirements of the State of Register of the aircraft. Generally, in aircraft systems, standards application is a task for a group of specialists. Table 4.1 contain an overview of selected regulatory documents that are studied and followed by engineers in the aviation industry and that could be applicable for the developed application.

Document number	Document description
DOT-VNTSC-FAA-00-22	Human Factors Considerations in the Design and Evaluation of Electronic Flight Bags (EFBs)
AC 25.1302-1	Installed Systems and Equipment for use by the Flight Crew
AC 25.1322-14	Flight Crew Alerting
AC 25.11B	Electronic Flight Deck Displays
AC 120-64	Operational Use and Modification of Electronic Checklists
CS/AMC 25.1302	Installed Systems and Equipment for Use by the Flight Crew
CS/AMC 25.1322	Flight Crew Alerting

Table 4.1: Regulatory documents related to design of aviation application[1].

Regulatory documents contain regulations, guidance material, and other recommendations of each aspect of avionics systems as aviation is one of the areas where consequences of poor color design can be life-threatening [1]. The general objective is to ensure that the displayed information will be easily and clearly distinguishable with enough visual contrast for the flight crew to see and interpret it. Recommendation relate to electronic display information elements and features such as labels, text, symbols, indications, and color-coding.

Text

Related general guidelines says that abbreviations and acronyms used in the text should be consistent with the established standards. It is acceptable to use only upper case letters for text labels. Contractions such as „can't" instead of „cannot" should be avoided. Regarding to the choice of font, the font should be compatible with the display technology. Although serif font may become distorted on some low pixel resolution displays, sans serif font such as Futura or Helvetica are recommended for displays viewed under extreme lighting conditions.

Colors

Table 4.2 describes a recommended colors for certain functions and can be found in the FAA AC 25-11B, which is dedicated to Electronic Flight Displays. Similar description can be found in EASA AMC 25-11 for Electronic Flight Deck Displays.

RED	Warnings, Invalid Flight Instruments
AMBER	Caution Conditions/Messages, Invalid Data, Miscompare Annunciation
MAGENTA	Flight Director Cue/Coupled Data
CYAN	Advisory Data/Messages, Selected Data
GREEN	Engine Modes, Normal Conditions, Current Data
WHITE	Scales, Dials, Tapes, Labels
GRAY	Units or Symbols, Inactive button labels, Legends

Table 4.2: Recommended colors for certain feature[2].

[1] https://www.faa.gov/regulations$_p olicies/$

[1] https://www.easa.europa.eu/document-library/acceptable-means-of-compliance-and-guidance-materials

[2] https://www.faa.gov/documentlibrary/media/advisory$_c ircular/ac_2$5 − 11*b.pdf*

The guidelines contains of the recommendation of color combinations that need to be avoided, such as:

- blue, or red and black,
- yellow or green and white,
- saturated red, blue, or yellow and green,
- saturated red and blue,
- yellow on purple,
- magenta on black.

The list is based on human factors research. The last item could be acceptable for lower critical items. The guideline deals also with proper labeling, menu, and symbols as well as font size and other graphics. An example shown in Figure 4.1 is often used as an explanation of usual problem with blue color.

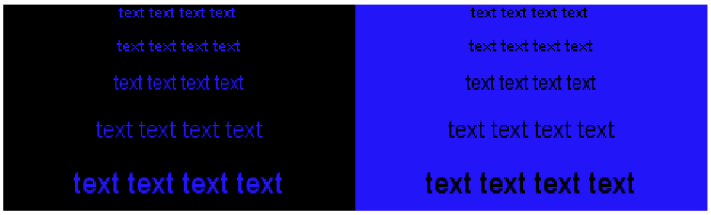

Figure 4.1: Usual figures explaining a problem with blue color[3].

Due to its low luminance, combination of blue and black leads to reduced text readability independently on font size [1].

Labels
Labels may include text and icons and should be consistent across all occurrences. Their orientation should ensure readability. If the text will be replaced using icons, its use should not cause confusion to the flight crew. Icons should be designed so that only brief exposure to the icon should be needed in order for the flight crew to determine the function.

4.2 Checklist on Cockpit Displays

The process of studying and developing an ECL can be traced back to the 1980s [19]. The research started in Boeing in response to documented checklist errors. It took almost 16 years until the ECL was certified and introduced in the Boeing 777. This section describes an ECL solution on Boeing 777, which represent a transport category large aircraft. For comparison, following subsection briefly describes solution used on Pilatus PC-12 business jet with smaller cockpit.

[3] https://colorusage.arc.nasa.gov/guidelines$_0$.*php*

4.2.1 Electronic Checklist Solution on Boeing 777

As it is shown in Figure 4.2, the checklist can be displayed on any of three multifunction displays (MFDs) marked green and can be operated by pilot as well as copilot through cursor control device (CCD) marked blue [19]. The remaining display marked yellow is known as EICAS. The term is abbreviated from Engine Indicating and Crew Alerting System. An EICAS system displays engine parameters and possibly other information such as cabin pressure and landing gear. The system alerts the crew about aircraft configuration issues with a Master Warning red light or Master Caution amber light and aural alert.

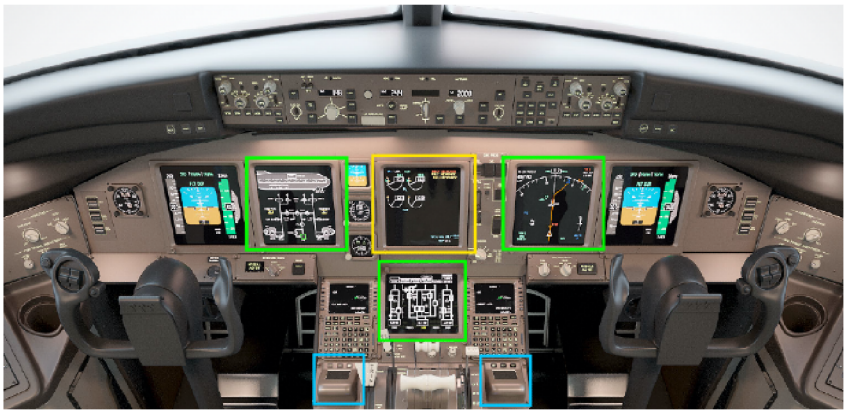

Figure 4.2: Boeing 777 flight deck layout[4].

As it is shown in Figure 4.3, the ECL can display all normal and non-normal checklists defined in Operational Manual. Normal procedures are presented sequentially and non-normal procedures are organized by subject area. Non-normal checklists takes priority over normal checklists. Some non-normal checklists are indicated by EICAS message. In case of time critical non-normal procedure, all necessary tasks are performed and checklist is displayed and confirmed after.

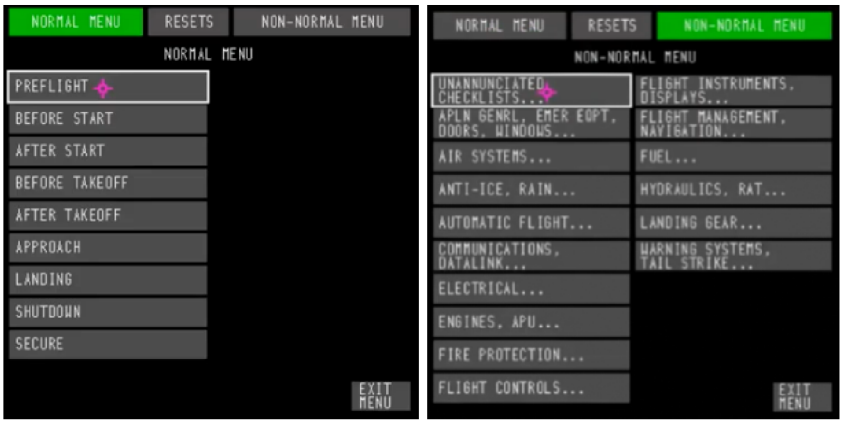

Figure 4.3: ECL normal and non-normal procedure selection[5].

[3]https://www.youtube.com/playlist?list=PL$_4$$ljWrVvM91FL_Q7wv2HGHU_M8MUR6Tu$

Typical normal checklist is shown in Figure 4.4. All incomplete steps are displayed by white color [19]. When the cursor is placed over some step, selection box appears. The system is able to sense if the switches on flight deck are set to the proper position. These steps are confirmed by the system without manual interaction. They are known as closed loop steps and shown as last three steps. Steps that have to be manually performed are called as open-loop steps and appear with a gray box next to them. If the current step is performed, the following one is highlighted automatically. Steps with green mark in are completed.

Checklist title is placed at the top. Normal checklist title is white and non-normal checklist title appears in the same color as EICAS message. In case of non-normal situation, it may happen that the condition no longer exists. The message is removed but the checklist is still displayed. Still, the checklist needs to be confirmed as completed. Some non-normal conditions does not have associated EICAS messages but do have a checklist. These are generally known as unannunciated checklists and have to be selected manually.

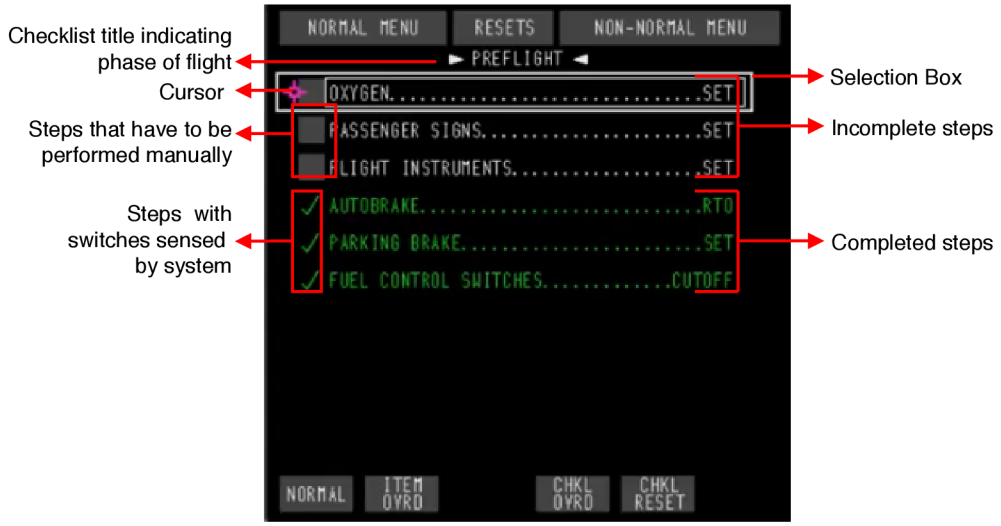

Figure 4.4: Pre-flight checklist related to normal procedures[6].

As it is shown in Figure 4.5, some tasks may contain conditional statement. The conditional statement usually begins with the word „if“. The ECL supports two types of conditional statements. The first of them is called an open-loop conditional statement. Yes or no must be selected and it is possible to change the selection anytime. The second type of conditional statement depends on a time delay. This conditional statement is associated with countdown timer positioned near the top right corner of selection box and starts automatically.

The countdown timer continues to run on the background even if the checklist is left. The step is active as long as the timer is running. When the timer reaches zero, the step become amber. Irrespectively of the type condition applies that if the conditional statement is true it turns green and the steps bellow will become active. If the conditional statement is false, the condition together with the steps bellow turn cyan.

Figure 4.5: Left: Conditional statement with countdown timer (timer is marked yellow). Middle: False conditional statement. Right: Open-loop conditional statement[8].

Multi-page checklist is shown in Figure 4.6. Page indicator is displayed on the right side of the display [19]. It can be seen that the checklist on the right contains three pages. The currently displayed page is visually indicated by a white slider around the page number. Page numbers are displayed in white and once they are completed, the page number turns green.

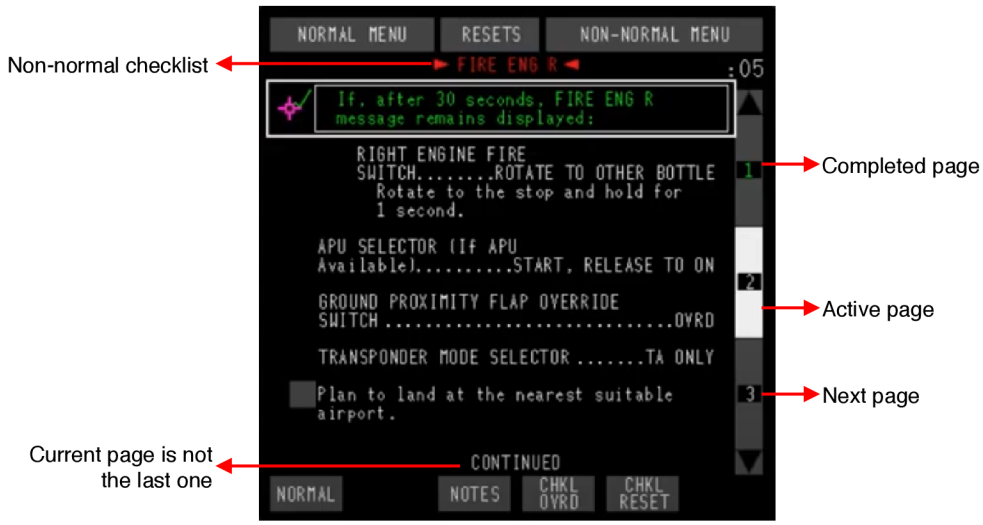

Figure 4.6: Multi-page checklist[8].

Checklists can be reset individually, in bulk within a procedure group, or all of them at once. Some checklists are reset automatically. Reset of normal checklist happens after a go-around. Go-around may happen when flight crew decides not to continue an approach or landing. The related checklists are automatically reset in this case. Another automatic reset is associated with non-normal checklists when they are completed and causing condition no longer exists. This way, the checklist is ready when the condition occurs again. The ECL allows the flight crew to override step or an entire checklist. If the step is not performed by choice, it turns to cyan color. Checklists are considered to be completed if all steps are green or cyan.

4.2.2 Electronic Checklist Solution on Pilatus PC-12

Figure 4.7 shows flight deck of the Pilatus PC-12 NG cockpit. The Pilatus PC-12 is equipped by Primus Apex system designed by Honeywell and is intended for single-pilot turboprop

[8]https://youtu.be/xEpUHEGuHlY

aircraft and very light jets [3]. These aircrafts are equipped by four displays, which leads to different layout in difference from Boeing displays. The ECL together with CAS messages can be displayed on both MFDs marked green. The default location is on the lower MFD, known as system MFD. The whole system including display swap can be operated through the Cursor Control Device (CCD) marked blue.

System MFD is divided to six windows with different approach. The CAS message window is marked yellow and is located above the ECL window marked green. CAS is abbreviated from Crew Alerting System and its functionality is similar to EICAS. Generally, the ECL functionality on Apex system is similar to functionality on Boeing system. There are some differences related to the way of how the features are displayed or named, however, both systems are based on the same regulatory requirements.

Figure 4.7: Left: PC-12 cockpit flight deck[9]. Right: System MFD display layout[10].

4.3 Checklist Applications for iOS

The most used solution includes ECL as part of the Electronic Flight Bag application [25]. Generally used abbreviation for Electronic Flight Bag is EFB. The term covers any portable electronic device intended primarily for pilot or flight crew use. Along with checklist, its functionality includes document storage, applications software enabling an independent performance of calculations needed for the operation of the aircraft, display of aeronautical chars, and more. EFB can be tracked to the past when many of these functionalities were accomplished with paper reference carried out in the aviation bag.

ForeFlight Checklist
One of the most used iOS aircraft checklist application is developed by ForeFlight [2]. ForeFlight Checklist is a feature available with a ForeFlight Mobile EFB Plus subscription plan. The application requires iOS 12.0 or later and is compatible with iPhone, iPad, and iPod touch devices. Application screenshots are shown in Figure 4.8. ForeFlight provides a list of built-in templates for variety of fixed-wing aircraft models and several rotorcraft models. Each template is customizable and can be shared with other pilots. The application includes normal procedures as well as abnormal and emergency procedures. Normal procedures are organized into pre-flight, in-flight, and post-flight groups. All procedure buttons are easily

[10]https://aerospace.honeywell.com/en/learn/products/cockpit-systems-and-displays/primus-apex-for-pilatus-pc-12ng
[10]https://www.youtube.com/watch?v=X6o8NNJ9GDA

accessible, which can save time in case of some time-critical situation in comparison with a paper checklist.

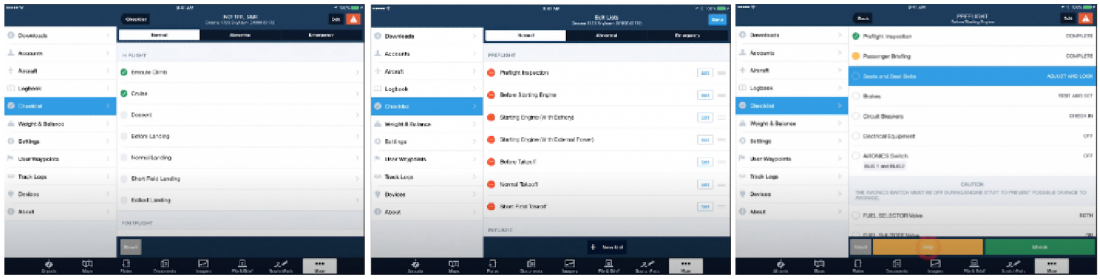

Figure 4.8: ForeFlight Checklist application[11].

Emphasis is also placed on visual feedback by color-coding. The application is equipped by Checklist Speak that is automatically calling out each item of the checklist. The speak mode can be adapted by moving faster or slower, paused or exited.

Garmin Pilot

Garmin Pilot is another EFB application containing checklists as one of many features [14]. Application screenshots are shown in Figure 4.9. It is a multiplatform application and besides iOS is also available for Android. iOS has to be in version 13.0 or later. Garmin provides a 30 days free trial on initial download. Garmin Pilot provides similar functionality as ForeFlight Mobile regarding the checklists feature.

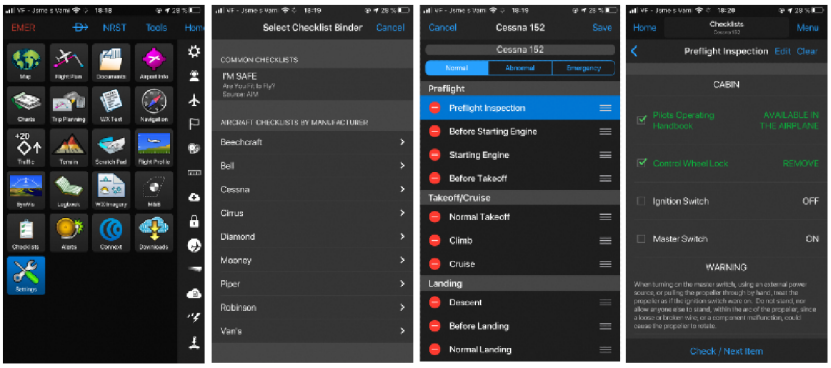

Figure 4.9: Garmin Pilot EFB screenshots.

The application provides checklist templates that can be edited. Checklists can be categorized based on procedure type to normal, abnormal and emergency and all of them are easily accessible through buttons on the top of the screen. Checklists can be sorted according to the flight phase, such as preflight, before take-off, and before landing. The checklist task turns green when it is selected. Completion confirmation is indicated visually by a green checkmark. Compared to ForeFlight, the application is missing Speak mode.

[11] https://foreflight.com/products/checklist/

smartCHECK

The smartCHECK application was developed by airWORK aviation media UG [27]. Figure 4.10 shows screenshots from the application. In contrast to the previously described applications, smartCHECK provides only checklists features. The application is available for iOS only and requires iOS version 10.0 or later.

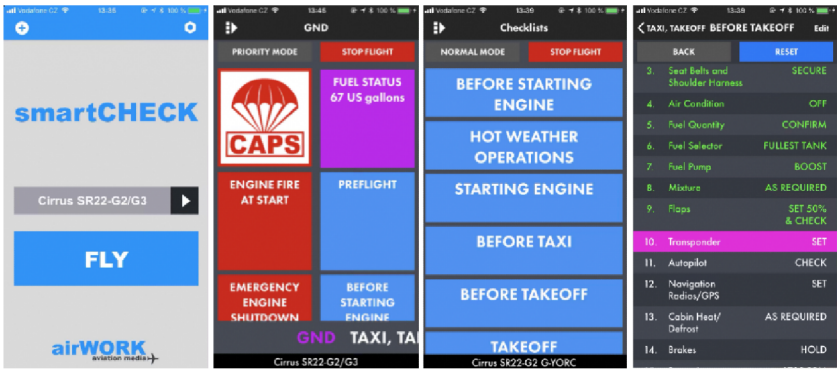

Figure 4.10: The smartCHECK application screenshots.

The application is equipped with several interesting features from which can be named automatic and GPS driven checklist activation, annunciation of altitude above ground after takeoff, and speech technology that reads written checklists to pilots. The application runs in normal and priority mode. The priority mode allows a user to change the step order. Text to speech technology features can be enabled or disabled globally for the entire application or for each checklist. The application allows pilot to choose between automatic read mode or read and confirm mode.

Chapter 5

State of the Art Analysis and Application Requirements

The aim of this thesis was to create an experimental mobile application for aircraft checklist execution with the use of augmented reality technology. The personal goal was to get acquainted with Unity 3D and object-oriented programming in C#. The task was solved with the use of consultation in the Aerospace division of Honeywell Technology Solutions. System requirements, as well as suggestions for improvement, was consulted with a consultant and several other employees whose work partially covers the subject in some ways. Those meetings were also an opportunity to slightly adapt the system requirements based on discovered facts.

5.1 State of the Art Analysis and Proposed Solution

Analysis of the existing solutions found that only one similar solution is currently available on the market. The solution is owned by the Aero Glass and is currently still in progress. The company has released several demonstration videos, proving a basic functionality of the concept. The analysis included standardized checklist solutions provided by leading manufacturers in aviation as well as available mobile applications intended for general aviation. Table 5.1 provides a summarization of key points.

	Strengths	**Weaknesses**
Aircraft systems	standardization (colors, font style, etc.)	unavailability in some aircrafts
	testing and validation before deployment	look at the checklist distracts from the flight deck
Mobile applications	availability	non-compliance with official standardization
	portability	look at the checklist distracts from the flight deck

Table 5.1: Comparison of aircraft systems and mobile applications.

The non-compliance with official standardization may be an issue for commercial pilots who are well aware of the standards. Taking a closer look at ForeFlight application,

deferred steps are marked amber and the active step is marked blue. Pilots aware of colors standard meaning could be confused. Amber is used primarily for caution conditions or messages. Blue can be confused with cyan, which is used for advisory messages. The biggest disadvantage of both solutions is the need to divert attention from the flight deck to look to the checklist display.

Concerning a suitable device, Table 5.2 provides a comparison of studied AR smart glasses. Listed devices are available on the market now or in the near future [13]. Battery life data has been taken from the manufacturer and the data appears to be promising.

	Price [$]	**Battery life [hours]**
Microsoft Hololens 2	3500	2-3
Magic Leap One	2295	up to 3
Focals by North	starting at 600	up to 18
Google Glass Enterprise Edition 2	999	up to 8
Vuzix Blade	1000	up to 8

Table 5.2: AR devices price and battery life overview.

However, when a user will use Alexa actively for 6 minutes the battery will be drained by 6 percent in a case of Vuzix Blade. Using Accuweather demo drain battery by 14 percent in 5 minutes. The information in Table 5.2 is based on the fact that the devices will turn into the sleep mode when a user does not use it for some time. A similar situation applies to any other smart glasses. It has to be taken into account that the mentioned smart glasses did not perform computationally intensive operations. The market provides another option, known as the Mira Prism headset. The headset is built for mobile devices such as the iPhone 8 and Mira introduced lately a solution with a built-in camera. With the headset, the battery life depends on battery life provided by smartphone. Comparing to the other solutions, the battery life will increase to several hours with greater computing power.

The proposed solution is based on implementing an experimental AR checklist application. The Mira Prism headset with an integrated camera seems to be suitable hardware. However, the headset is not available yet. Thus the application will be developed for the iPhone 8 device with iOS operating system and deployment on the headset will take place outside this thesis.

5.2 System Requirements

System requirements for the experimental application have been consulted and are based on the analysis of existing solutions. The implementation does not depends on fully implemented ECL as is known and used in aviation, but the application shall maintain following functionality.

- Display normal checklist procedures,
- prepare data structure parsing for abnormal and emergency checklist procedures,
- implement reset, defer, and recall operations related to checklists,
- recognize if a pilot announce flight instrument check and not even look at the direction,
- implement guidance support during a checklist performing.

The format and structure of checklist configuration files were not specified. Used colors should be in compliance with the standardized meaning. There were discussions about how to deal with choosing an appropriate programming language, possible frameworks that can be used, and rendering engine. The concerns occurred as eventual product development is strictly controlled with the number of procedures and specifications focusing also on used technologies. It was decided that this does not need to be considered as it is an experimental application.

5.3 System Architecture

The system architecture shall contains four input components providing data to application core, as it is shown in Figure 5.1.

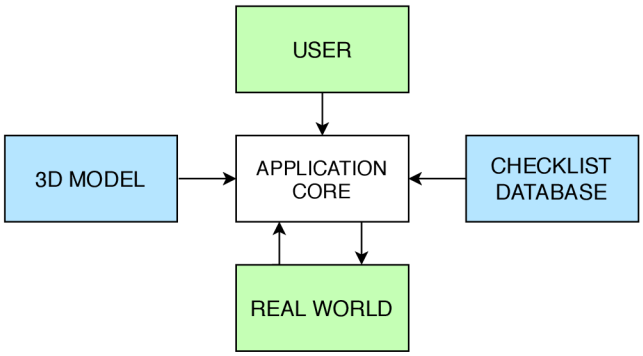

Figure 5.1: Proposed system architecture.

Input components can be divided into two groups. The first group is marked blue and includes a data components providing necessary information ensuring application functionality. Input data are processed only once immediately after the application starts. The second group is marked green and is formed by components representing a constantly processed inputs from user and data feed from device camera.

5.3.1 Input components

Real world component represent environment the system will be used in. The environment differs based on aircraft type but it always will be well known in advance. The concept of application functionality will be based on known markers placed in the real world environment.

3D model is a key element as it can provide accurate information about model element dimensions and their relationship in 3D space. The system architecture will benefit from this advantage. The 3D model will be adjusted with markers to determine a user's spatial position and orientation. In model and the real world, markers position have to be in consistent. One marker has to be related to a group of model objects otherwise, it could result in an untraceable number of markers.

User component represents operations performed by the user. The user should be able to view, browse, and execute checklists, perform operations related to the checklist, and enable or disable available features.

Checklist database is composed of checklist configuration files that will be designed for specific aircraft. The data organization reflects a grouping or logical sequence. Normal procedures usually follow a sequence to prepare the aircraft for flight, flying, and shutting down after fa light. Emergency and abnormal procedures include tasks grouped by major aircraft subsystems. As it is shown in Figure 5.2, data for normal and emergency checklists have identical structures, but the abnormal checklist follows a slightly different pattern.

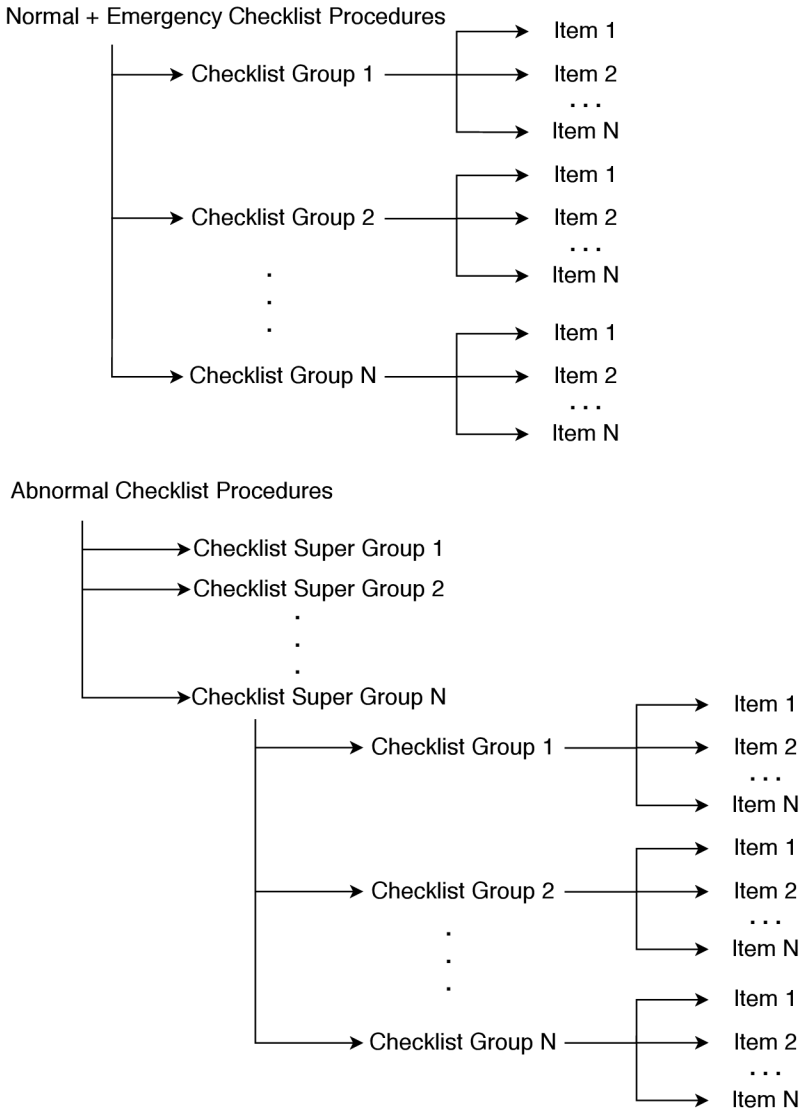

Figure 5.2: Checklist data structures.

The dataset used for application purposes should be obtained from Aircraft Operating Manuals/Flight Crew Operating Manuals (AOM/FCOM). AOM/FCOM are considered as a reference for safety operation of an aircraft. Checklist data shall be stored in JSON data format. JSON is abbreviated from JavaScript Object Notation and is a widely used language-independent open standard file format and data interchange format. Scheme for normal and emergency procedures is shown in Listing 5.1. Listing 5.2 shows schema for

abnormal procedure. Schema begins with a header specifying aircraft type the checklist is intended and checklist procedure type.

```
{
"aircraft_type": string,
"checklist_type": string,
"groups": [
    {
    "group_ID": string,
    "tasks": [
        ...
     ]
    }
  ]
}
```

Listing 5.1: Scheme for normal and emergency procedures.

```
{
"aircraft_type": string,
"checklist_type": string,
"super_groups": [
  {
    "super_group_ID": string,
    "groups": [
      {
        "group_ID": string,
        "tasks": [
            ...
        ]
       }
     ]
   }
 ]
}
```

Listing 5.2: Schema for abnormal procedure.

As it is shown in Listing 5.3, tasks contains array of objects. Each object consists of five elements.

```
"tasks":
[
    {
        "condition": bool,
        "task": string,
        "check1": string,
        "check2": string,
        "flightInstrument": string
    }
]
```

Listing 5.3: Schema for array of tasks

The element named as *condition* determines whether the step contains a conditional statement. If the value is true, the element named *task* contains a text of the conditional statement. In the case of a positive result, the element named as *check1* contains next step. Otherwise, the following action is listed under the element named as *check2*. Element named as *flightInstrument* contains an empty string.

If the value related to condition statement is false, the *task* describes the area of interest in the current step. The *check1* guides the pilot what to do with the area of interest. The *check2* contains an empty string. The area of interest is described in *flightInstrument*. This content needs to be identical with the flight instrument name in the virtual model. A dash indicates that the step is not associated with a flight instrument.

5.3.2 Application core

When it will be possible, the application core should use working solutions in order to not get distracted by reinventing the wheel. The application core itself shall implement following functionality:

- checklist functionality,
- model processing,
- AR functionality.

Checklist functionality requires checklist database on the input. Model processing requires a suitable 3D model on the input. Both JSON data and 3D model data shall be obtained from input components. Based on the system requirements, the application should be prepared for abnormal and emergency procedures processing. These procedures have higher priority and may cause interruption of normal procedures.

5.4 User Interface Proposal

GUI requirements description is based on a manner that is known and used for requirements for aviation systems. Application design shall be as minimal as possible. It must not overwhelm the screen as the pilot still needs to see especially a cockpit and surroundings. Figure 5.3 shows the final design created with respect to the mentioned requirements. The design settled on three statically displayed elements marked blue.

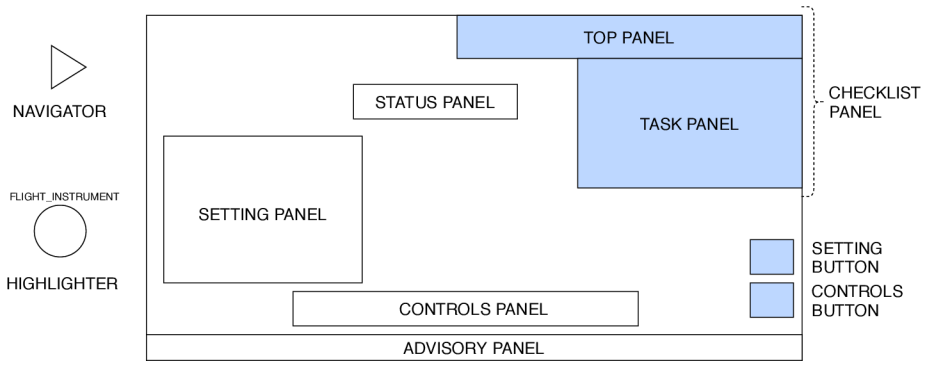

Figure 5.3: Basic drawing illustrating all possible elements on on the screen.

The initial inspiration for application design was obtained during the state of the art research. The main influence came from Aero Glass and its concept of AR checklist. Aero Glass placed checklist information to the top right corner. This position is probably the best possible and it was selected with respect to the area that will be overlaid. Its position in any corner on the left side would overlap flight deck directly in the user's view. Similar situation occurs if the panel would be placed to the bottom right corner as it may overlap another important flight instruments.

Application GUI style shall be implemented as defined in Table 5.3. Primary font size shall be applied to elements such as procedures, group, and step name and status messages.

Secondary font size shall be applied to elements such as procedures, super group, and group status message as well as step confirmation text.

Text	Definition
Font	Futura medium
Title font size	33 px
Primary text font size	30 px
Secondary text font size	25 px
Minor text font size	16-20 px

Color	Hexadecimal code
White	#FFFFFF
Blue	#00CCFF
Green	#00FF00
Gray	#292929
Red	#FF0000

Table 5.3: Left: Font style description. Right: Hexadecimal color codes.

The top panel shall consist checklist menu button, step back button, checklist title and progress indicator. The checklist menu shall contain operations related to the checklist tasks and shall be designed as drop-down menu. Table 5.4 summarizes checklist operations included in drop down menu.

Option	Associated operation
Reset all	resets all checklists
Reset	resets currently displayed checklist
Re-call	pilot access deferred item of the current checklist
Defer	pilot intends to complete the task later

Table 5.4: Checklist menu operations with description.

The application shall implement features described in Table 5.5. User shall be allowed to enable or disable any of the feature. The navigator feature is a key point of the application and shall be enabled automatically.

Feature	Description
Navigator	pilot's view is guided to the right part of the cockpit
Voice control	pilot can confirm a task performance by voice command
Audio read	individual tasks are read to the pilot, which simulates a copilot

Table 5.5: Application features with description.

The Navigator shall be designed as arrow and shall be shown if currently tracked marker does not have associated flight instrument related to active step. The application should display a warning message if the pilot announces the confirmation of active step even in this state. The concept can be applied as a basic check if pilot at least looked at the right place in the cockpit.

Group of GUI elements shall include the most important one, named as a highlighter. The highlighter shall be shown on a position of flight instrument related to active step if proper marker will be in the user's field of view. The highlighter shall be marked green. The color use is based on inspiration by Head-up displays GUI design due to its sensitivity in the retina of the human eye. The displayed style of the highlighter shall depend on the selected mode as described in Table 5.5.

Mode	Description
Active	Only active flight instrument shall be marked
Active + Next	Additionally, instrument related to the following step shall be marked
Active + All	Additionally, instruments related to the tracked marker shall be marked

Table 5.6: Application features with description.

The second mode was included after suggestion from general aviation pilot. Only active highlighter shall be marked green. Any other shall be marked gray.

Chapter 6

Implementation

The application was developed using C# language, AR Foundation framework, and Unity 3D. AR Foundation framework provides AR functionality. The application is intended for iOS devices, thus ARKit XR plugin is used.

6.1 Outline of the Application

Figure 6.1 shows the outline of the application. The application core consists of several blocks, which are responsible for capturing the image of the real world, processing it, rendering virtual objects in the image, and displaying the result to the user.

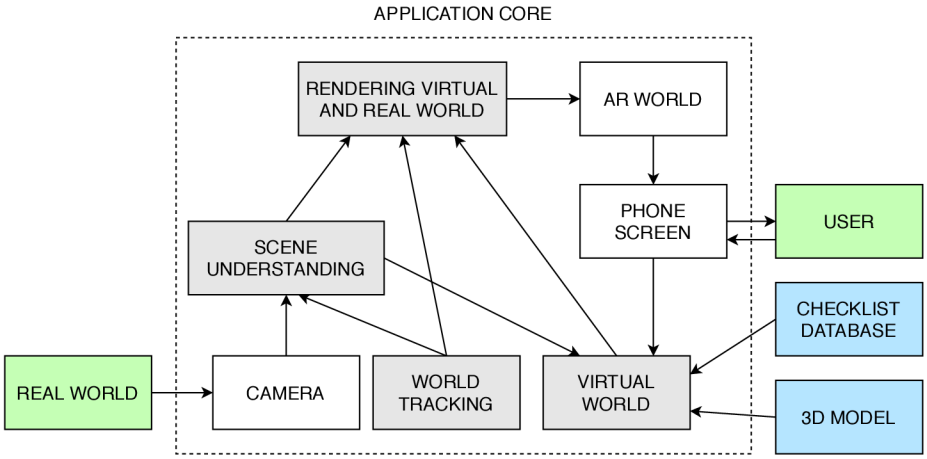

Figure 6.1: Outline of the application.

The user is able to control the application behavior through the phone screen or by voice commands. The blocks marked gray are described in following sections.

6.2 Scene Understanding and World Tracking

Tracker maps the real world to know what it looks like in digital form. A new map is initialized every time when the application is launched. Once the mapping is finished and 3D map is established, the user moves withing the map. The spatial orientation is done

using a known markers in the real world. The Tracker requires a reference image library and a number of images in the library. The reference image library collects an image with its name, width, and height. Only images included into the reference image library will be tracked. The dimensions are used to determine distance from camera to the image. The distance is key value for the application functionality, thus the dimensions are accurate as possible. When an image is tracked, it is recorded in the 3D map, as it is shown on the example in Figure 6.2.

Figure 6.2: Left: Visualized accumulated frames of mapped real work with found tracked feature points[2]. Right: Real world environment[2].

The 3D map contains trackable point that are in the camera's field of view and extends as the user moves. To not overload the memory with tracked points, the framework uses a sliding window. The sliding window manages only a variable amount of recent points based on time and distance.

6.3 Virtual World

Figure 6.3 shows a block schema of the virtual world and where the input components are processed at a lower level.

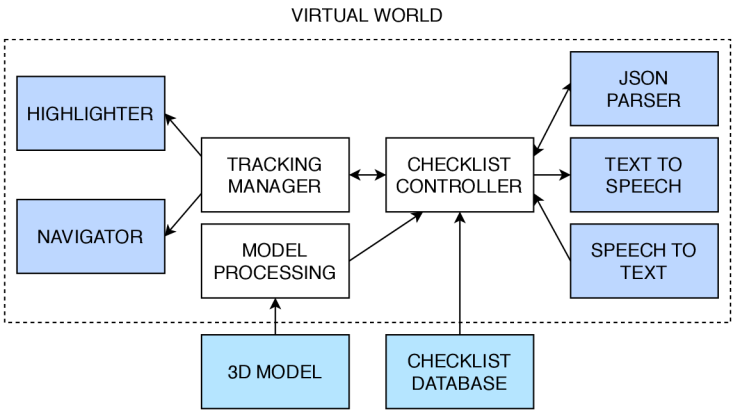

Figure 6.3: Virtual world block schema.

[2]https://github.com/ittybittyapps/ARVisualizer

The schema reflects a Unity concept, where all functional parts are implemented in smaller functional units called *Components*. The *Components* marked white represents an application basis. The color light purple is used for additional *Components*, whose use depends on the application basis.

Model Processing

Position of Highlighters rendered to the world space is calculated based on distances from markers to flight instruments in 3D flight deck model. The concept functionality was tested in cockpit simulator, as it is shown in Figure 6.4. The simulator have to be adjusted with markers. In model, the markers have to be modeled with respect to their position in the simulator. Each model element is represented by independent *GameObject* with its transforms and other properties.

Figure 6.4: Comparison of simulator's cockpit and virtual model.

GameObject can be accessed by script through its name or tag. The tag is intended to identify a group of *GameObject* and meet the needs of the concept. *GameObject* were divided into two groups named flightInstrumentTag and markerTag depending on their purpose. In the model, each marker is assigned a group of flight instruments based on the shortest distance. Obtained distance is calculated in the Unity coordinate system. Both Unity and AR Foundation uses a left-handed coordinate system so the distance does not have to be inverted.

Tracking Manager and Checklist Controller

These two components are summarized in a block diagram because their functionality is closely related. Rendering is controlled based on their status. Tracker provides a detected image data. The data are processed and compared with data obtained from the JSON configuration file and the 3D model. The result of this process determines if Highlighter or Navigator would be rendered. Figure 6.5 shows flow diagram of described concept, which is implemented in the ChecklistController class and TrackingManager class.

ChecklistController class works with checklist data and implements preparing data for rendering, and maintaining the internal state of the application. TrackingManager class implements AR functionality and calculation of Highlighter position or Navigator direction. Each class implements its internal state machine, which controls the application runtime. Both states are set in ChecklistController.

ChecklistController takes a currently tracked marker name and returns the proper state to the TrackingManager. If the tracked marker has assigned a required flight instrument, a match is found. Then, TrackingManager creates an instance of Highlighter and renders it.

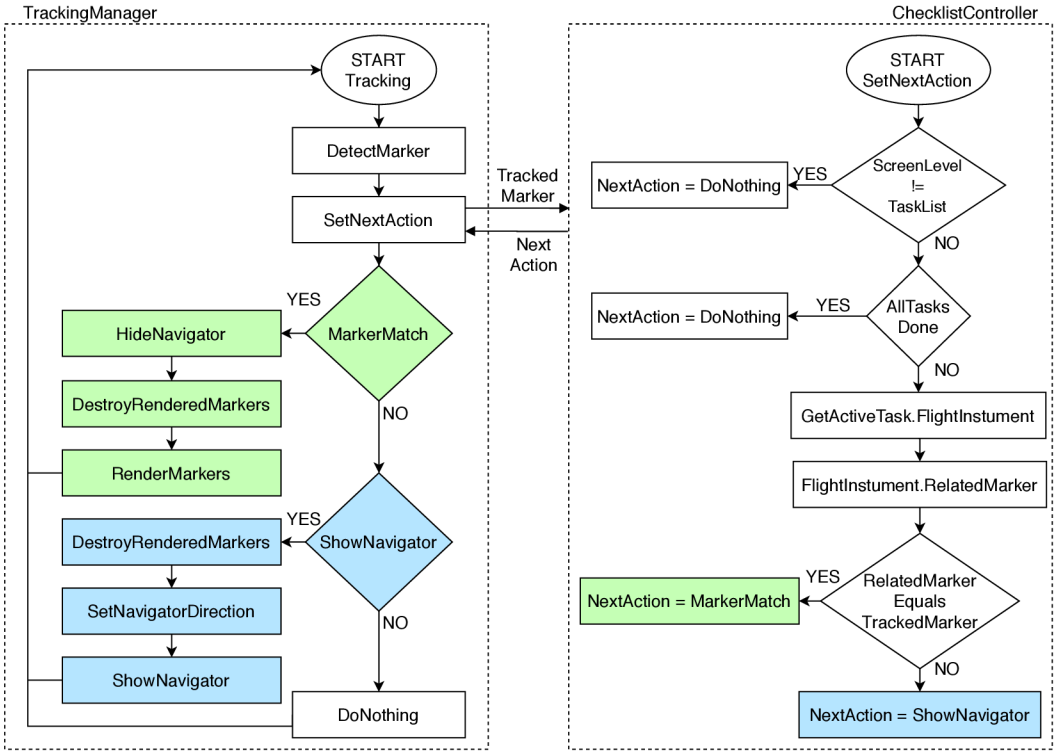

Figure 6.5: Flow chart diagram illustrating state machines controlling the application runtime.

Since the application will be used while moving, the Highlighter has to be destroyed and instantiated in the same frame, as its position is constantly changed. If the match is not found, TrackingManager determines the position of marker that should be tracked instead and the Navigator points to this position.

Highlighter
Highlighter represents a key component. This component is constructed as *Prefab*. AR Foundation provides a plane detection provided by AR Plane Manager. This functionality was implemented and tested in the cockpit simulator. The idea was that the Highlighter would be positioned parallel to the detected plane. However, the current implementation provides support only for horizontal and vertical plane detection based on edges. This resulted in that the plane was not recognized at all, or it took non-trivial time to recognize it. The approach was changed and the Highlighter is position facing the camera.

Navigator
The Navigator itself is symbolized by arrow pointer. After the application is launched, the user is guided to detect all markers to store their original position. The direction vector is calculated from the original position of the tracked marker to the original position of the required marker. The concept suffers from the same issue as the Highlighter and the direction vector needs to be recalculated with respect to the rotation of the camera.

JSON Parser

Checklist configuration files are loaded as Unity objects from Unity's Resources directory. File content is based on the schema described in Section 5.3. Unity provides methods for converting Unity objects to and from the JSON format. The JSON was deserialized into classes using the appropriate method.

Speech to Text

Apple provides a native framework to recognize spoken words. As the framework was created for Swift, which is known as a programming language for Apple devices, it is not possible to use it directly in Unity. This obstacle has been resolved by buying a C# wrapper of the Speech framework available in the Unity Asset Store. The package is named Mobile Speech Recognizer. It is a cross-platform wrapper for a native iOS and Android speech to text frameworks supporting all main languages.

This component is implemented only for confirmation that an active step was performed. The first tested word for the confirmation command was „done" but it was often recognized as „them" and other similar words. Speech recognition has been proved as successful for the „check" command. After the initial success, a state machine allowing full voice was implemented. Most of the commands included at least two words, such as show normal, show menu, etc. This set did not returned so accurate results and the commands had to be repeated. Due to the higher recognition inaccuracies, the list of commands was limited to the „check" command only.

Text to Speech

This component uses Google Cloud Text To Speech wrapper. This platform is interesting as it provides many useful APIs and services, such as Maps API, Cloud Vision API, and more. The Google Cloud Text to Speech provides methods for synthesizing text, choosing a variety of voices and languages, and setting speaking rate tuning and pitch tuning. It is necessary to have API Key to use the functionality. The implementation was already divided into the logical and rendering parts, which allowed Text To Speech integration.

6.4 Rendering Virtual and Real World

Initial mapping is relative to the camera pose. Figure 6.6 illustrates situation when the aircraft will make turn. As the camera rotates withing the 3D map, flight instrument distance obtained from model has to be recalculated.

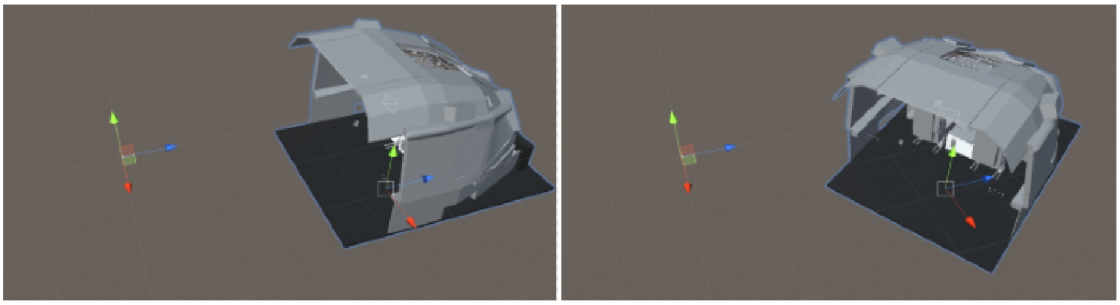

Figure 6.6: Left: Initial state. Right: When an aircraft makes a turn, the distance value has to be recalculated with respect to camera's rotation.

The flight instrument position is recalculated based on current camera rotation described by Euler's angles. To get at least a little closer to the real conditions, the application was tested in a moving car. As it is shown in Figure 6.7, the testing revealed issues with tracking stability, which is a known issue of many frameworks.

Figure 6.7: Left: Required state. Right: The system has lost tracking ability.

Several studies successfully used Moving Average filter, FIR, or Kalman filter. The Moving Average filter was used because of its simplicity. The ineffectiveness of this filter was evaluated by ordinary observation. It turned out that the filter proves better results for applications in a relatively stable environment.

One other different approach has been tried. It was intended to calculate The position of the flight instrument was calculated from the last preserved position and distance from the camera. It worked pretty well during slow movement but failed otherwise. The reason may be found in the way of creating a 3D map. If the movement in the real world is too fast, the orientation is completely lost and Tracker may initialize a new map.

6.5 Graphical User Interface

Application design was changed several times at the beginning of the development. Initial long-used variant appeared to be ordinary after several months. The appearance has been redesigned to make it more modern. Both variants are shown in Figure 6.8.

Figure 6.8: Left: Initial design. Right: Final design.

The application was consulted with a general aviation pilot, who is a member of the Brno-Slatina aeroclub. The final design of the application was introduced to the pilot

and was evaluated positively. However, he pointed out that checklists fall into the highest safety critical category. Each system status in this category shall be indicated by more than a color, commonly by additional symbology. This comment was taken into account. Figure 6.9 shows all panels that can be displayed at the same time.

Figure 6.9: Top panel displays progress indicator, title, step back button, and drop down checklist menu. The checklist menu differs with respect to the displayed list. The reset button is displayed always. If a checklist is displayed on the task panel, the checklist menu contains defer and recall buttons.

The task panel is constructed from buttons and displays checklist related information. The list is automatically scrolled, to relieve the load of the pilot. Active step is marked by green box with green arrow. Each button is divided into two parts. If the panel shows a list of groups or procedures, the top part contains a group, or procedure name. The bottom part is empty until all child items are completed. The completion state is indicated by the word „COMPLETED“ displayed green. If the panel shows list checklist steps, the top part specifies what should be done and the bottom one specifies how it should be done. Text on the bottom part appears green if the step is completed. Each state is additionally visualized by a progress indicator. If list is completed, fully filled progress indicator is supported by an additional green circle in its center. Figure 6.10 shows screen that appears after launching application.

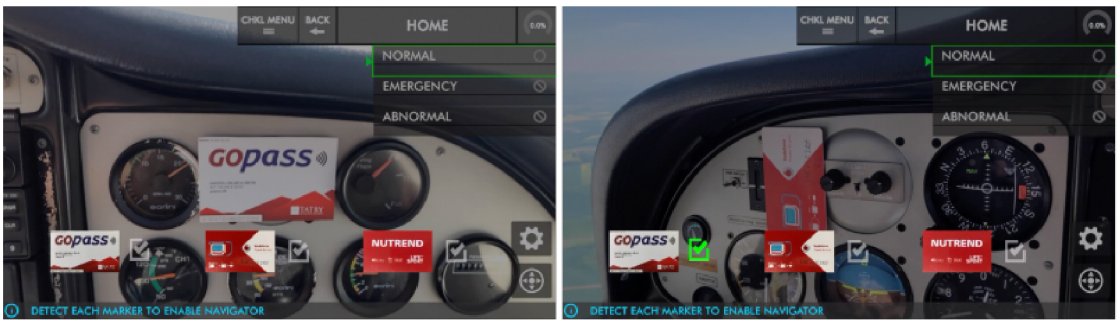

Figure 6.10: After launching the application, the user is forced to detect all markers.

The user is not allowed to do anything until all markers are detected. The screen automatically disappears when it is done. Detected markers are distinguished from the undetected ones by the color of the check icon.

Figure 6.11 shows the highlighter's default mode and GUI appearance in case of deferred step. The default mode is marked as Active-Only. As the name suggests, only one active Highlighter is shown at the same time. The Highlighter is designed as proposed in the previous chapter. It consists of a green circle and the name of the flight instrument.

Figure 6.11: Left: Flight instrument marking in the default mode. Right: An active step can be deferred through Defer button.

As it was proposed, several operations are implemented for the checklist. An active step can be deferred through the button of the same name. User can possibly perform deferred steps again through the Re-call button. After pressing the Re-call button, the first of the deferred steps is activated. Deferred steps are counted equally as the completed ones, so it is not required to make a re-call.

Figure 6.12 shows appearance of the remaining modes, which are marked as Active+next and Active+All.

Figure 6.12: Left: The Active+Next mode. Right: The Active+All mode.

The application can display two types of status messages, both shown in Figure 6.13. First of the message notes that the checklist is completed. This message will remain until the state changes. The user is offered the option of displaying the next checklist if there is one. The second message occurs when a user confirms that an active step was performed but the proper flight instrument is not in a pilot's field of view. This message will be displayed for five seconds.

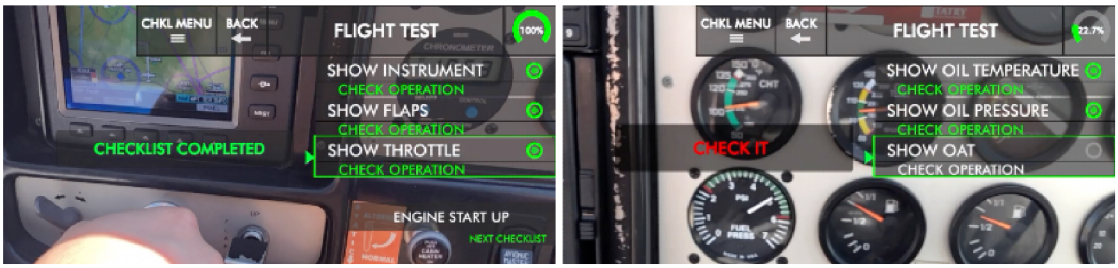

Figure 6.13: Left: Message indicating completion of the checklist. Right: Active task was confirmed but flight instrument was not even in the field of view.

The setting panel is shown in Figure 6.14. The highlighter mode can be changed and the options are mutually exclusive.

Figure 6.14: Setting panel allows to enable or disable implemented features and change highlighter mode.

Implemented features can be enabled or disabled at any time. Active features are marked green and supplemented by a green box.

Chapter 7

Experiments and Reached Results

The goal of this thesis was to propose and implement an AR checklist application. Four general test sets have been created to evaluate overall application impression as well as GUI perception. Partial generality was necessary because the application had been tested on the ground and during the flight.

7.1 Experiments Description

Four test sets were prepared, in which the participants used the application and tried to perform a checklist procedure. The evaluation metrics were based on the subjective perception of the individual participants filled in the questionnaire and based on the interaction with the participant. The questionnaire is attached to Appendix A and contains open questions as well as closed questions. The questionnaire also contains questions where the user evaluated the specified parts of the application by using a scale.

Test Set 1 - Highlighter
This tests set has been designed to evaluate size, visibility, readability, and stability of the Highlighter GUI element. The test also includes whether the element indicated the correct flight instruments. These questions were evaluated using a scale. Within the open questions, the user's preferences when choosing the display mode and possible ideas for improvement were interviewed. For completeness of results, the element was monitored during the test on the ground and during the flight.

Test Set 2 - Navigator
The second test set has been oriented to the Navigator and its position, size, visibility, stability, and helpfulness from the user's perception. These parts were evaluated using a scale. The open question was designed to obtain a user's suggestion, possible issues he was facing, and any ideas for improvement. For completeness of results, the element was monitored during the test on the ground and during the flight.

Test Set 3 - Voice and Speaker
This test set has been designed to evaluate the usability of implemented features. The Text to speech feature is named as Speaker and the Speech to text feature is named as Voice. These features were evaluated by using interviews and scales in the questionnaire.

Test Set 4 - Overall GUI Impression
The fourth test set has been created to evaluate the overall effect of the GUI on the user from the point of view of intuitiveness, attractiveness, visibility of all GUI elements such as buttons and panels, application response times, and the overall organization and clarity of the GUI. This test set evaluated by using scales in the questionnaire.

The group consisted of three participants designated as A, B, and C. Each participant was given a prepared questionnaire that is attached to Appendix B. The group was diverse even in this number. General information about each participant is summarized in Table 7.1.

Participant	**A**	**B**	**C**
Pilot license owner	Yes	No	No
Used checklist form	Paper	-	-
ECL ever used	No	-	-
Familiar with checklist concept	-	No	Yes
Familiar with AR concept	Yes	Yes	Yes
AR apps ever tried	No	Yes	Yes
AR checklist app idea rate	4	3	4

Table 7.1: General information.

Participant A is well aware of flight systems, owns a Private Pilot Licence (PPL), and has experience with software development in aviation. On the other side, aviation systems knowledge of participants B and C was more or less limited. Participants B was in an aircraft for the first time ever.

7.2 Experiments Realization

The first phase was initially supposed to take a place in the HF lab in Honeywell. This approach was changed because the laboratory environment could affect the results of the GUI perception by the user. The light conditions are stable and much more suitable and some visibility defects could remain hidden. The experiments took a place in Vyškov airport, and aircraft known as Tecnam P2002 Sierra was used for the purposes of the experiments. Tecnam P2002 Sierra is shown in Figure 7.1.

Figure 7.1: Left: Tecnam P2002 Sierra. Right: Tecnam's flight deck.

The aircraft is a twin seat, single engine aircraft with tapered, low wing, fixed main landing gear and steerable nose wheel[1]. The aircraft has capacity for one passenger and its the cruise speed is 120 knots, thus 222 km/h.

The experiments required a prepared 3D model of the aircraft's flight deck. The flight deck is noticeably easier in comparison with transport category aircrafts such as the Airbus A320 flight deck. Figure 7.2 shows a simple model of the Tecnam's flight deck created in Unity.

Figure 7.2: Comparison of flight deck on the left side and virtual 3D model on the right side. Model elements marked blue represents an image markers.

The proportions of the model corresponds with reality 1:1. The checklist database contained normal procedures and was based on Tecnam P2002 Sierra QRH.

7.2.1 On Ground Tests

Participants B and C were introduce into the aircraft checklist concept. All of them were introduce into the application concept and functionality. After the initial introduction, each participant was asked to perform the Engine Startup checklist. The test case was set as follows, where participants were asked to:

- launch the application and detect all markers,
- find the Engine Startup checklist,
- perform the checklist,
- display the following checklist and perform few steps,
- reset checklists state,
- enable and disable all available features in different combinations,
- try every highlighter mode,
- perform all available checklist operations under checklist menu.

[0]https://www.tecnam.com/aircraft/p2002-sierra-mkii/

Each participant had a separate space for testing without any time pressure in order to become familiar with the application. Figure 7.3 shows collage taken during on ground testing.

Figure 7.3: Photos taken during on ground testing.

The participants were encouraged to talk about what they are doing what is on their mind.

7.2.2 Flight Tests

The checklist configuration file was extended to contain a special checklist just for the flight test. There were some concerns related to marker tracking during flight, due to the turbulence and light conditions. It would not be pleasant for the pilot if the passenger would waving the smartphone in front of his eyes for a long time. The additional checklist was named as Flight Test and includes only a flight instruments on the left side on the flight deck.

Participants B and C completed a 20-minute flight each. Although only one pilot participated in the experiments, it was managed that also he was able to test the application for a shorter time. As a first step, I took a preliminary flight to make sure that the weather and ling conditions are appropriate. In the case that the application would not work, the experimental flight would become a sightseeing flight for other participants. The baseline was the Flight Test checklist execution. The flight time was not so long, which led to the creation of two test sets with different tasks.

The first test set was assigned to participant B and contained the following steps:

- defer some steps,
- re-call deferred steps,
- restart whole application a perform all task again,
- try to perform before landing checklist.

The second test set was oriented to the Navigator functionality and participant C was instructed to:

- complete part of the flight with enabled Navigator,
- complete part of the flight with disabled Navigator,
- try to confirm a step while tracking an inappropriate marker,
- restart whole application and perform all tasks again.

Figure 7.4 and shows collage taken during the flight testing. The flight testing was recorded by GoPro camera and iPhone screen recording.

Figure 7.4: Photo compilation was taken from GoPro camera and iPhone screen recording during flight tests.

The GoPro camera was located on the participant's chest. The location was not ideal and the records are not as good as they could have been. A technical issue occurred during the last flight and the screen record from this flight was not taken.

7.3 Reached Results

The questionnaires were evaluated and their results are described in this section. Listed tables summarize questions based on a scale rating.

Test Set 1 - Highlighter
Table 7.2 provides a summary of Highlighter rating. Comparing both experimentation phases, it was reveals that Highlighter stability dropped the most. The tracking instability led to the subsequent issues with rendering position, visibility, and readability. Almost all cases proved that the Highlighter indicated correct flight instruments. The incorrect

designation was proved in one case. The debugging discovered a minor bug and two blocks were exchanged in the model.

	A	B	C	Average
Position	5	3	4	4
Size	5	3	5	4.33
Visibility	5	5	5	5
Readability	5	5	5	5
Stability	4	3	5	4

	A	B	C	Average
Position	4	2	2	2.67
Size	5	4	4	4.33
Visibility	3	5	4	4
Readability	5	5	4	4.67
Stability	3	2	2	2.33

Table 7.2: Left: On ground tests scale rating. Right: In air tests scale rating.

All participants preferred Active-Only highlighter mode, but participant A also liked Active+Next mode. Active+All mode was specific. Flight instruments are usually quite close on the flight deck, so their names were not very legible from a greater distance. But unacquainted participants looked at a flight instrument from a shorter distance and find out its meaning. Interviews during experiments revealed an idea for a new highlighter mode. Its concept is based on modified Active+All mode. Instead of highlighting all flight instruments related to tracked marker, there could be highlighted only instruments related to the active checklist.

Test Set 2 - Navigator

Table 7.3 provides a summary Navigator rating. The overall rating of the navigator AR element was slightly worse then it was expected. Its functionality was more stable during development. The decrease in the rating is partly caused by a change in appearance just a few hours before experiments. The original 2D symbol was replaced by a 3D symbol. The size and stability of the 3D symbol were not verified enough.

	A	B	C	Average
Position	3	3	3	3
Size	3	2	4	3
Visibility	3	5	4	4
Stability	3	3	2	2.67
Helpfulness	4	3	3	3.33

	A	B	C	Average
Position	3	2	1	2
Size	3	1	4	2.67
Visibility	3	5	4	4
Stability	3	2	1	2
Helpfulness	4	3	2	3

Table 7.3: Left: On ground tests scale rating. Right: In air tests scale rating.

Based on the opened questions and observation, the Navigator was more useful for participants that were in the cockpit for the first time. It was obvious that the Navigator reduces the time needed to find a required flight instrument. One participant even stated that this functionality is mandatory for such an application. On the other side, some participants were sometimes bothered by this functionality. This feeling was rather evoked by appearance and then by instability. However, the implementation could take into account several different positions of the Navigator on the screen.

Test Set 3 - Voice and Speaker

Table 7.4 provides a summary of text to speech and speech to text features. The application has to be connected to the intercom to test those features in flight. Both features were only included in on-ground testing, which naturally lowered their ratings. Obtained results show

that text to speech feature obtained ratings from opposite sites of the scale. Participant A expressed a concern about distracting pilots during communication with ATC (Air Traffic Control).

	A	**B**	**C**	**Average**
Speech to Text	3	4	3	3.33
Text to Speech	1	4	5	3.33

Table 7.4: Summarizing of application features usability.

Users B and C no chance to take into account this possible issue due to the lack of experience. The Speech To Text feature obtained a much consistent result. Some participants suggested expanding the feature with more commands.

Test Set 4 - Overall GUI Impression
Table 7.5 shows application partial rating as well as an overall evaluation calculated from the arithmetic average. The individual evaluation criteria of application achieved a high rating. Application control was clear and easy to use. Used colors were appropriately chosen.

	A	**B**	**C**	**Average**
Intuitive	5	4	5	4.67
Easy to learn	5	5	5	5
Clear	4	4	5	4.33
Attractive	5	5	5	5
Reaction time	5	3	5	4.33
Element visibility	5	5	5	5
Organized	5	5	5	5

Table 7.5: Summarizing of overall GUI appearance.

All participants agreed that the application would be much more valuable using smart glasses. Weather conditions may have been reflected in application rating during flight tests. The tracking was sometimes lost during turbulence, which led to stability issues with Highlighter and Navigator. The worst situation was observed after intense maneuvering during turbulences. The system was not able to stabilize itself even after landing. The Highlighter was intensively shaking and the text was not readable at all. This issue occurred only once within three flight tests.

Performed experiments confirmed sufficient tracking stability of the used framework during a smooth flight. Based on the evaluation, the implemented application obtained positive feedback from participants.

Chapter 8

Conclusion and Future Directions

The goal of this Master's thesis was to design and implement an experimental application using augmented reality when performing an aircraft checklist. The goal has been fulfilled.

The topic of augmented reality in an aircraft cockpit was studied and is described in chapter three. All possible hardware options for the application were explored and are summarized in the second chapter. As a result of the exploration in section 2, the hardware for the application purposes was chosen as described in the fifth chapter. Rendering engines were explored and the overview is described in the second chapter. The selected engine is listed in chapter five. The software architecture is proposed in the chapter five.

The application marks a flight instrument related to the active step if it is in the user's field of view. Otherwise, a navigation arrow directing to the required cockpit part is shown. The application was tested in Tecnam P2002 Sierra in both, on the ground and in flight. A suitable checklist was prepared for the flight test purposes but testing on the ground was performed on normal checklist procedures defined in the QRH. The feedback was positive from the participants in the experiments.

Concerning future work, it is necessary to improve tracking during unstable conditions. The application needs to be deployed on a suitable headset to make it usable in real conditions. Another possibility includes an implementation of support for abnormal and emergency checklists execution. The application could process data obtained from flight instruments and verify that the pilot has actually inspected the flight instrument. Furthermore, it could be interesting to investigate and implement another option of spatial orientation to avoid using special markers in the cockpit.

Bibliography

[1] *25-11B - Electronic Flight Displays* [online]. Washington, D.C., USA: U.S. Department of Transportation Federal Aviation Administration, 2014 [cit. 2020-10-5]. Available at: https://www.faa.gov/documentlibrary/media/advisory_circular/ac_25-11b.pdf.

[2] *ForeFlight Checklist*. Houston, Texas, USA: ForeFlight, 2020 [cit. 2020-10-5]. Available at: https://foreflight.com/products/checklist/.

[3] *Primus Apex* [online]. Charlotte, North Carolina, USA: Honeywell Aerospace, 2020 [cit. 2020-10-5]. Available at: https://aerospace.honeywell.com/en/learn/products/cockpit-systems-and-displays/primus-apex.

[4] *Airbus soars with wearables.* [online]. Dublin, Ireland: Accenture, c2020 [cit. 2020-01-02]. Available at: https://www.accenture.com/gb-en/success-airbus-wearable-technology.

[5] *Using the HoloLens to facilitate plant maintenance*. Exton, Pennsylvania, USA: Bentley Systems, c2020 [cit. 2020-3-6]. Available at: https://communities.bentley.com/other/old_site_member_blogs/bentley_employees/b/stephanecotes_blog/posts/using-the-hololens-to-facilitate-plant-maintenance.

[6] AUKSTAKALNIS, S. *Practical Augmented Reality: A Guide to the Technologies, Applications, and Human Factors for AR and VR*. 1st ed. Boston, Massachusetts, USA: Addison-Wesley, 2016. ISBN 978-0134094236.

[7] BELLAMY, W. *9 Companies Using Augmented and Virtual Reality in Aviation.* [online]. Miami, Florida, USA: Avionics International, 2017 [cit. 2020-01-02]. Available at: https://www.aviationtoday.com/2017/08/24/9-companies-using-augmented-virtual-reality-aviation/.

[8] BRADSKI, G. R. *Learning OpenCV*. Sebastopol, California, USA: O'Reilly, 2008. Computer Programming. Robotics. ISBN 978-0-596-51613-0.

[9] BUTTFIELD ADDISON, P., MANNING, J. and NUGENT, T. *Unity Game Development Cookbook: Essentials for Every Game*. Sebastopol, California, USA: O'Reilly, 2019 [cit. 2020-10-5]. ISBN 9781491999127. Available at: https://books.google.cz/books?id=q_2MDwAAQBAJ.

[10] COMPANY, H. M. *Hyundai Virtual Guide Introduces Augmented Reality to the Owner's Manual* [online]. Seoul, South Korea: Hyundai Motor America, c2020 [cit. 2020-10-5]. Available at: https://www.hyundainews.com/en-us/releases/2080.

[11] De Florio, F. *Airworthiness : an introduction to aircraft certification : a guide to understanding JAA, EASA, and FAA standards*. 1st ed.th ed. Oxford, UK: Butterworth-Heinemann, 2006. ISBN 0-7506-6948-9.

[12] Degani, A. and Wiener, L. *Human Factors of Flight-deck Checklists: The Normal Checklist*. Mountain View, California, USA: Ames Research Center, 1990 [cit. 2020-01-02]. NASA contractor report. Available at: https://books.google.cz/books?id=bZlBAQAAMAAJ.

[13] DIAZ, J. *From Apple Glasses to Hololens 2: AR glasses you can buy now (and soon)*. [online]. New York City, New York, USA: Tom's Guide, 2019 [cit. 2020-01-02]. Available at: https://www.tomsguide.com/reference/ar-glasses.

[14] Garmin. *Announcement: New Garmin Pilot Tools for Pre-Flight Planning In-Flight Operations* [online]. Olathe, Kansas, USA: Garmin, c1996-2020 [cit. 2020-10-5]. Available at: https://www.garmin.com/en-US/blog/aviation/announcement-new-garmin-pilot-tools-for-pre-flight-planning-in-flight-operations/.

[15] Geroimenko, V. *Augmented reality art: from an emerging technology to a novel creative medium*. New York City, New York, USA: Springer, 2014. Springer series on cultural computing. ISBN 9783319062020.

[16] Huffman, B. *The EFB & iPAD'S Enhance the electronic checklist (ECL)*. [online]. Tucson, Arizona, USA: Flight Assurance, 2015 [cit. 2020-01-02]. Available at: http://flightassurance.net/blog/wp-content/uploads/2015/09/EFB-Meets-The-Electronic-Checklist.pdf.

[17] Lanham, M. *Learn ARCore - Fundamentals of Google ARCore*. Packt Publishing, 2018. ISBN 9781788830409.

[18] Lee, J. *Creating Augmented and Virtual Realities*. Birmingham, UK: Packt Publishing, 2016. ISBN 9781784398156.

[19] McKenzie, W. A. and Hartel, M. C. Design of the Electronic Checklist on the Boeing 777 Flight Deck. In: *SAE Technical Paper*. Pittsburgh, Pennsylvania, USA: SAE International, September 1995 [cit. 2020-10-5]. DOI: 10.4271/951986. Available at: https://doi.org/10.4271/951986.

[20] Peddie, J. *Augmented Reality: Where We Will All Live*. New York City, New York, USA: Springer, 2017. ISBN 9783319545028.

[21] QIAO, P. R. S. D. L. L. H. M. a. J. C. *Web AR: A Promising Future for Mobile Augmented Reality-State of the Art, Challenges, and Insights*. [online]. Piscataway, New Jersey, USA: IEEE, 2019 [cit. 2019-10-02]. ISSN 0018-9219. Available at: https://ieeexplore.ieee.org/document/8643424.

[22] Schmalstieg, D. *Augmented reality: principles and practice*. Boston, Massachusetts, USA: Addison Wesley, 2016. ISBN 978-0-321-88357-5.

[23] Serván, J., Mas, F., Menéndez, J. and Ríos, J. Using augmented reality in AIRBUS A400M shop floor assembly work instructions. In:. College Park, Maryland, USA: American Institute of Physics, September 2011, vol. 1431, p. 633–640. DOI: 10.1063/1.4707618.

[24] SKYBRARY. *Normal Checklists and Crew Coordination* [online]. San Francisco, California, USA: SKYbrary, 2018 [cit. 2020-01-02]. Available at: https://www.skybrary.aero/index.php/Normal_Checklists_and_Crew_Coordination_(OGHFA_BN).

[25] SKYBRARY. *Electronic Flight Bag (EFB)* [online]. San Francisco, California, USA: SKYbrary, 2020 [cit. 2020-10-5]. Available at: https://www.skybrary.aero/index.php/Electronic_Flight_Bag_(EFB).

[26] STEPHANIDIS, C. *HCI International 2019 - Posters: 21st International Conference, HCII 2019, Orlando, FL, USA, July 26–31, 2019, Proceedings, Part II*. New York City, New York, USA: Springer, 2019 [cit. 2020-10-5]. Communications in Computer and Information Science. ISBN 9783030235284. Available at: https://books.google.cz/books?id=4KKhDwAAQBAJ.

[27] UG airWORK aviation media. *SmartCHECK V5.0 - Manual* [online]. Munich, Germany: airWORK aviation media UG, 2019 [cit. 2020-10-5]. Available at: http://www.airwork.biz/wp-content/uploads/2015/11/smartCHECK-V5.0-Manual.pdf.

[28] WANG, W. *Beginning ARKit for iPhone and iPad: Augmented Reality App Development for iOS*. New York City, New York, USA: Apress, 2018. ISBN 9781484241028.

[29] XIAO, C. and LIFENG, Z. Implementation of mobile augmented reality based on Vuforia and Rawajali. In: *2014 IEEE 5th International Conference on Software Engineering and Service Science*. Piscataway, New Jersey, USA: IEEE, 2014, p. 912–915. ISBN 9781479932788.

[30] YASUFUKU, K., KATOU, G. and SHOMAN, S. Game engine (Unity, Unreal Engine). *Kyokai Joho Imeji Zasshi/Journal of the Institute of Image Information and Television Engineers*. Tokyo, Japan: Inst. of Image Information and Television Engineers. 2017, vol. 71, no. 5, p. 353–357. ISSN 13426907.

Appendix A

Contents of the included storage media

- dp.pdf
- Demonstration videos - demoVideo.mp4, flightDemo1.mp4, flightDemo2.mp4
- ARChecklistApp.unitypackage
- Scripts
- Readme

Appendix B

User Experience Survey

AUGMENTED REALITY IN COCKPIT - USER EXPERIENCE SURVEY

GENERAL INFORMATION

Level of experience in aviation

- ☐ Pilot license owner
- ☐ In a training
- ☐ Just a fan

[Pilot license owner] What checklist form do you mostly use?

- ☐ Paper
- ☐ Electronic

[Pilot license owner] Did you ever used an electronic checklist?

- ◯ YES
- ◯ NO

[Just a fan] Are you familiar with the concept of aircraft checklist?

- ◯ YES
- ◯ NO

Are you familiar with the concept of augmented reality?

- ◯ YES
- ◯ NO

Did you ever try some AR applications?

- ◯ YES
- ◯ NO

How would you rate the AR checklist app idea?

	1	2	3	4	5	
Worst	◯	◯	◯	◯	◯	Best

STATIC TEST - CHECKLIST EXECUTION ON GROUND

Performed normal checklist procedure(s)

Did the Highlighters indicate correct flight instruments? Was everything clear?

- ◯ YES
- ◯ NO

If NO, specify the issue

Highlighter position

	1	2	3	4	5	
Worst	◯	◯	◯	◯	◯	Best

Highlighter size

	1	2	3	4	5	
Worst	◯	◯	◯	◯	◯	Best

Highlighter visibility

	1	2	3	4	5	
Worst	◯	◯	◯	◯	◯	Best

Highlighter text readability

	1	2	3	4	5	
Worst	◯	◯	◯	◯	◯	Best

Highlighter stability

	1	2	3	4	5	
Worst	◯	◯	◯	◯	◯	Best

Which Highlighter mode do you prefer?

- ☐ Active Only
- ☐ Active + Next
- ☐ Active + All

Any suggestions, issues, improvement ideas related to Highlighter?

Navigator position

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

Navigator size

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

Navigator visibility

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

Navigator stability

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

How much was Navigator helpful?

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

Any suggestions, issues, improvement ideas related to Navigator?

Voice commands usability

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

Would you use this feature?

○ YES
○ NO

Speaker commands usability

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

Would you use this feature?

○ YES
○ NO

Caused some element a confusion?

○ YES
○ NO

If YES, which one

DYNAMIC TEST - CHECKLIST EXECUTION DURING FLIGHT

Performed normal checklist procedure(s)

Did the Highlighters indicate correct flight instruments? Was everything clear?

- ◯ YES
- ◯ NO

If NO, specify the issue

Highlighter position

	1	2	3	4	5	
Worst	◯	◯	◯	◯	◯	Best

Highlighter size

	1	2	3	4	5	
Worst	◯	◯	◯	◯	◯	Best

Highlighter visibility

	1	2	3	4	5	
Worst	◯	◯	◯	◯	◯	Best

Highlighter text readability

	1	2	3	4	5	
Worst	◯	◯	◯	◯	◯	Best

Highlighter stability

	1	2	3	4	5	
Worst	◯	◯	◯	◯	◯	Best

Which Highlighter mode do you prefer?

- ☐ Active Only
- ☐ Active + Next
- ☐ Active + All

Any suggestions, issues, improvement ideas related to Highlighter?

Navigator position

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

Navigator size

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

Navigator visibility

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

Navigator stability

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

How much was Navigator helpful?

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

Any suggestions, issues, improvement ideas related to Navigator?

Voice commands usability

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

Would you use this feature?

○ YES
○ NO

Speaker commands usability

	1	2	3	4	5	
Worst	○	○	○	○	○	Best

Would you use this feature?

○ YES
○ NO

Caused some element a confusion?

○ YES
○ NO

If YES, which one

GUI

Please rate overall impression from GUI (1 - Not Satisfied, 5 - Very Satisfied)

	1	2	3	4	5
Intuitive	○	○	○	○	○
Easy to learn	○	○	○	○	○
Clear	○	○	○	○	○
Attractive	○	○	○	○	○
Reaction time	○	○	○	○	○
Elements visibility	○	○	○	○	○
Organized	○	○	○	○	○

Is it always clear, which elements are active?

○YES
○NO

If NO, which element raised a confusion?

Is used text font sufficiently readable?

○YES
○NO

Is used text size sufficient?

○YES
○NO

If NO, which text element would you change and how?

Any suggestions, issues, improvement ideas related to GUI, or application functionality at all?