MUHAMMAD FAHAD MINHAS
01-134131-054
MUHAMMAD SHAHBAZ
01-134131-061

Coloring Book With Augmented Reality

Bachelor of Science in Computer Science

Supervisor: Dr. Imran Siddiqi

Department of Computer Science
Bahria University, Islamabad

November 2016

Certificate

We accept the work contained in the report titled "Coloring Book with Augmented Reality", written by Mr. Muhammad Fahad Minhas and Mr. Muhammad Shahbaz as a confirmation to the required standard for the partial fulfillment of the degree of Bachelor of Science in Computer Science.

Approved by . . . :

Supervisor: Dr. Imran Siddiqi (Associate Professor)

Internal Examiner: Mr. Ghulam Ali Mirza (Assistant Professor)

External Examiner: Mr. Jawad Bashir (Assistant Professor)

Project Coordinator: Dr. Arif Ur Rahman (Assistant Professor)

Head of the Department: Dr. Faisal Bashir (Associate Professor)

November 21st, 2016

Abstract

Coloring books are one of the earliest opportunities for the children to enhance their creativity in the early growing stages. Unfortunately, the proliferation and popularity of the advanced digital devices has affected the real life activities like coloring the books and children are more attracted towards using and playing with the digital devices. However, given the advantages of digital devices, one cannot stop the children from using the digital devices rather giving the right directions to using these devices. Augmented reality is a relatively new technology which holds great potential to attract users due to its interactivity and can be used to bridge the gap between real-life activities and digital devices. In this project, we present the Coloring Book with Augmented Reality in which children color the provided sketches of different drawn characters and examine their drawings through the developed application. The drawing is detected in real-time and the corresponding 3D model is augmented in the video stream with the same colors provided by the children accompanied with related animations. This has become possible after researching and employing different novel techniques. 3D-model textures are mapped first according to the corresponding image targets which is termed as as UV-mapping. In UV-mapping, we define which part of the 3D-model's mesh would get the texture from which part of the image target i.e. defining 2D coordinates in 3D coordinates. Live image processing is also employed in which the video frames are processed to extract the image target area to generate the colors defined by the children and then apply that textured image through the mesh renderer to the 3D-model before augmenting it. Finally, the application experience is validated by a user study during the testing procedure and examined through different perspectives.

Acknowledgments

All praises to Almighty Allah who provided us with the opportunity to enhance our skills and also provided us with the courage, power and wisdom to work and plan with full devotion to our country by implementing the work being done internationally.

We pay gratitude and thanks to our supervisor **Dr. Imran Siddiqi** for guidance, support, patience and understanding throughout the research period. He is an inspiration for us and we are honored to be in his supervision. May Almighty Allah bless him with perfect health and potential to carry on with valuable work. Ameen!

We would also like to thank the Computer Science Department, Bahria University, Islamabad for providing us with the good facilities and learning environment so that we are able to do the work at this level.

There are no words that can acknowledge the sacrifices, love and moral support given to us by our beloved families. Their prayers had never left us alone in any good or worst circumstances. May they be included among all whom Allah loves. Ameen!

MUHAMMAD FAHAD MINHAS, MUHAMMAD SHAHBAZ
Bahria University Islamabad, Pakistan

November 2016

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

2D	Two Dimensional
3D	Three Dimensional
AR	Augmented Reality
API	Application Programming Interface
APK	Android Package Kit
BRISK	Binary Robust Invariant Scalable Key points
CB	Coloring Book
CBWAR	Coloring Book with Augmented Reality
CI	Communication Interface
CV	Computer Vision
EI	External Interface
GUI	Graphical User Interface
HI	Hardware Interface
HIT	Human Interface Technology
HSV	Hue Saturation Value Color Format
ICB	Interactive Coloring Book
IDE	Integrated Development Environment
JPEG	Joint Photographic Experts Group
RGB	Red Green Blue Color Format
SDK	Software Development Kit
SDLC	Software Development Life-Cycle
SIFT	Scale Invariant Feature Transform
SI	Software Interface
SLM	Star Life-Cycle Model
SQA	Software Quality Attributes
SRS	Software Requirement Specification
SURF	Speeded-up Robust Features
TC	Test Case
UC	Use Case
UI	User Interface
UT	Usability Testing

Chapter 1

Introduction

All parents wish the very best for their children in every sphere of life from physical to mental health and early education to career goals. With the changing trends in our lifestyles, parents these days find it hard to spend the same amount of time with their children as it was the case a few decades back. A primary concern of young parents these days is to ensure that their children develop a due interest in the learning activities during early stages of their schooling. With the tremendous increase in the number of digital devices including computers, smartphones and tablets and the way they have become a part and parcel of our routine activities, the attraction of traditional activities has decreased. Consequently, the engagement of children in traditional learning activities has reduced as compared to the time they spend in interacting with these digital devices. This adds to the worries of parents as, in some cases, parental control on the activities of children using these devices is hard to ensure.

In an attempt to address the aforementioned issues, an interesting aspect is to reduce the gap between the traditional paper and pencil based learning activities and children's interactions with the digital devices. An attractive option to bridge this gap is the concept of Augmented Reality (AR) which holds great potential to develop interest and attract the attention of young minds. AR based learning solutions can provide children with the ability to link their imaginations in the traditional real-world activities side by side with the augmented world on the digital devices.

AR allows exploiting its interesting features and the attractiveness of digital devices to direct renewed emphasis on traditional learning activities. Among these, one of the most practiced activities in early days of school is coloring the characters, sceneries, landscapes etc. provided in their coloring books and, this activity, in fact, makes the subject of our study as well. We introduce the idea of an interactive coloring book (ICB) providing children with an opportunity to actually see their imagination in the augmented world. The key components of the proposed idea include a paper-based coloring book complemented with an augmented reality based application. The application will serve to bridge the colored drawings of children with animated cartoon characters and sceneries on the digital device. It is expected that with such an application, the children will find the traditional coloring activity much more interesting and attractive.

1.1 Objective

The key objective of the project is to develop an interactive augmented reality based application that enhances the learning experience of the children. The coloring activity has been chosen for this purpose. Allowing the possibility to render the 2D objects with their colors in the drawing book of children as 3D objects in the augmented environment to present an attractive learning application for the children.

1.2 Problem Description

As discussed earlier, with the increase in the availability and accessibility of smart digital devices equipped with a camera, the last few years have witnessed a shift of paradigm and many of the applications as well as frameworks have migrated to the mobile platform or are in process of migration. Computer vision (CV) and augmented reality (AR) [1] based solutions are the most notable of these. Inspired by the potential these solutions offer, we intend to develop an Android-based mobile application combining the powerful features of computer vision and the attractiveness of augmented reality to come up with an interesting coloring activity for children. A paper based coloring book with different characters and scenes are to be developed complemented with a mobile application.

The application is supposed to allow children to view the coloring page using the device camera and see the augmented objects on the device. Once colored, the application will render the colored augmented objects providing children with the opportunity to see how their colored objects look like in the real world.

1.3 Methodology

The project methodology relies on first designing a coloring book with characters, scenes and other objects of interest. An image of each page of the coloring book is stored as a template in the database of the system. For each image, a corresponding 3D model is also saved in the database. During an interactive session with the coloring book, the child is intended to focus the camera of the mobile device on the page. The received video frames are processed to detect key points or markers and match them with the stored templates. Matching can be carried out using any of the well-known computer vision descriptors including Scale Invariant Feature Transform (SIFT) [2], Speeded-up Robust Features (SURF) [3], Binary Robust Invariant Scalable Keypoints (BRISK) [4] or any other similar descriptor. In our case, using the matching functionality of Vuforia SDK for matching the camera frames with registered image targets makes this process quite simplified. Once the image on the drawing page is matched with one of the templates in the database, the corresponding 3D model is rendered on the screen. Once the child has completed coloring, it requires mapping the 2D regions of the colored image to the corresponding regions of the 3D object and finding the respective transformation matrices. This process also requires UV-mapping which is discussed in detail later in the document. An overview of the matching and rendering technique is presented in Figure 1.1.

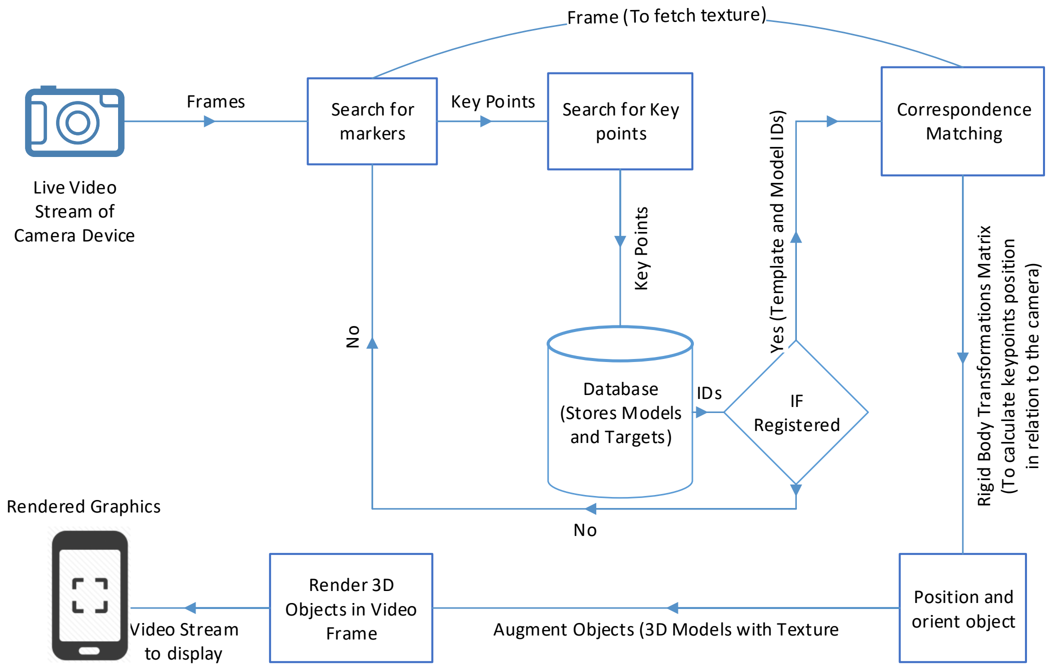

Frame (To fetch texture)
Frames
Live Video Stream of Camera Device
Search for markers
Key Points
Search for Key points
Correspondence Matching
Key Points
Yes (Template and Model IDs)
No
Database (Stores Models and Targets)
IDs
IF Registered
Rigid Body Transformations Matrix (To calculate keypoints position in relation to the camera)
No
Rendered Graphics
Video Stream to display
Render 3D Objects in Video Frame
Augment Objects (3D Models with Texture
Position and orient object

Figure 1.1: Application Overview

To support the image classification task, two methods have been primarily investigated for AR books. These include fiducial registration [5] and the natural feature registration. Fiducial registration [5] is related to providing a specially designed symbol to the target so that they are easily detectable [5, 6]. These symbols are used to improve the accuracy and reduce the complexity of registration, but they are affected by occlusions and modifications and do not offer a generalized framework. Natural feature registration is the dynamic features searching methodology [2], and describing them in such a way that ensures they are detectable, identifiable and unique regardless of changes in viewpoint, scale, orientation, and other transformations [2, 3]. Once these features have been matched from both the marker and camera image, the transformation is calculated [7]. The aforementioned descriptors SIFT, SURF and BRISK can be employed for natural feature registration.

Vuforia SDK [8] is not an open-source SDK. While we are using the detection functionality provided by the Vuforia itself, it is assumed by most developers that Vuforia uses a bunch of natural features registration algorithms. That is the combination of two best algorithms together. *SIFT* and *FERNS* are most probably deployed by the Vuforia which is discussed in the paper [9]. Furthermore, general modification and the combination of both the algorithms can be seen in the Figure 1.2.

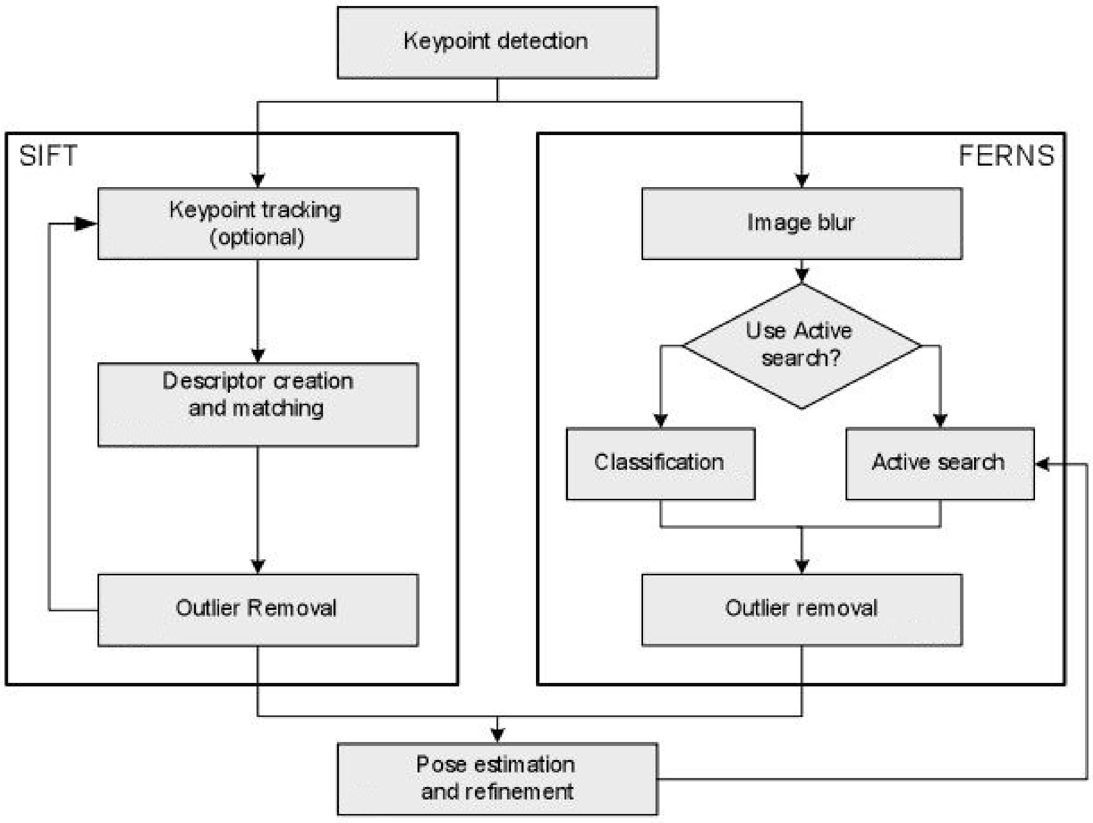


Figure 1.2: Overview of the SIFT and Ferns pipelines

1.4 Project Scope

The application is designed to be as robust as possible and easy to use. However, due to the nature of the application, we need to impose certain constraints to allow better recognition of templates and subsequent rendering of objects. We consider black and white lines art pages on the drawing book and the images of these pages will be stored in the database. The application will read and recognize pre-defined sketches/drawing which will be provided in its database. Naturally, if an image is presented which is not in the database, the application will not be able to render its model.

We mainly consider producing the augmented object for the uncolored and later the colored drawing. An interesting extension to this could be runtime texturing where the child can color parts of objects and see the augmented object during the coloring process. This run-time rendering, however, will be considered as an extension to the current application. This will require implementation of features like deformable surface tracking [10], texture generation from 2D images for 3D meshes [11], lookup-maps and real-time processing of videos for matching the templates and identifying the regions that have been colored [12].

1.5 Solution Application Areas

The proposed system has the potential to be of great interest to the children which will not only enhance their learning experience but will also provide them the ability to see their colored drawings as augmented objects, hence mapping their imagination to real world objects. This attractive application will also ensure a productive and educational usage of the digital devices, a part and parcel of today's life.

1.6 Organization of Report

This document is organized as follows. A review of the existing similar and related applications is presented in Chapter 2. Chapter 3 presents the requirements specifications followed by the system design in Chapter 4. System implementation and the realized results are discussed in Chapter 5 and Chapter 6 respectively. Chapter 7 concludes the report with a discussion on various possible extensions of this project.

Chapter 2

Literature Review

This chapter presents an overview of augmented reality and related image processing techniques as well as a discussion on similar and related applications.

2.1 Augmented Reality Literature

Augmented Reality is an uprising computer vision field from the last few decades. With time, it has enormously evolved and has attracted the world due to its versatility. Significant research and development efforts have been carried out exploring the capabilities of augmented reality.

Authors in [13] present a survey of AR techniques where 3D objects are integrated into a 3D real environment in real time. It describes the medical, visualization, manufacturing, path planning, military, and entertainment applications that have been explored. This paper portrays the qualities of increased AR frameworks, including a discussion about the exchange of the tradeoffs amongst optical and video mixing approaches. Registration and detecting mistakes are two of the most concerning issues in building successfully enlarged reality frameworks, so this paper outlines current endeavors to beat these issues. Future directions and areas requiring further research are discussed. This survey provides a starting point for anyone interested in researching or using augmented reality.

Furthermore, Augmented Reality on Android smartphones has been discussed by the researchers in [14]. Authors concluded that today's Android smartphones can successfully be used for AR applications. Although the authors employed an open source AR SDK ARToolkit [6], other SDK's can also be used for the application development.

Limited resources in smartphones pose a great challenge but certain research efforts have been made in which developers have used computationally complex algorithms on smartphones. In [15], authors have shown a number of tasks including high processing being implemented on the smartphones.

2.2 Image Processing

Coloring book with augmented reality contains very critical to time tasks. The camera frames need to be fetched and processed, colored regions are required to be extracted and colors are to be applied to the 3D model before the augmentation of the corresponding model. Consequently, fast image processing operations are required to be carried out on the fetched camera frames to ensure real-time rendering.

Authors in [16], present an application where automatic color extraction from the images with the predefined color sets is presented. The algorithm searches a large number of color sets, a color is index based on the database and it is created in a similar fashion to that of file inversion. This allows very fast indexing of the images by the color contents. Furthermore, details about different regions such as the color set, size, and location, enable a rich variety of queries that specify both color content and spatial relationships of regions. Researchers present the single color extraction, indexing method and they have contrasted it to other available color approaches. They have examined multiple and single color extraction and image query on a database of 3000 colored images.

As the children do not always color correctly, different portions of the image or sketch are left blank therefore we may also need to do $in-painting$. Inpainting is related to addressing the task of filling a portion of an image from other parts of the image. The methods for $inpainting$ can be split further into two categories: diffusion-based and patch-based [17].

Region extraction is carried out through boundaries of the objects but children mostly do the rough coloring and may not always respect the object boundaries. Therefore, certain image processing techniques can be employed for the recovery of the image. In paper [18], certain techniques are discussed for this purpose. Researchers have proposed a methodology in which they transform the input image from RGB to HSV color space, in which only the luminance channel is used because it mostly captures the information about original black line draws. Lines are more visible in the HSV luminance channel than the other formats like the grayscale image. By applying adaptive thresholding to the luminance image, we get a binary image with lines representing the object. Small noisy connected components can be removed later and Gaussian smoothing can be employed to remove the staircase effect of binarization.

2.3 Related Applications

A number of studies have been carried out on the development of open source augmented reality frameworks [19, 20]. Among notable frameworks, authors in [21] present a novel framework to create marker based augmented reality applications for mobile devices. The framework comprises five main subsystems. These include a) Presentation: with a game engine as the core component, b) Tracking: which implements different computer vision algorithms, c) Interaction: composed of a custom visual editor which processes all the inputs to the framework, d) World Model: used to store and provide access to the digital representation of the world and e) Context: used to provide status information of the entire system. Another significant work on the problem of fetching the color information from the live video stream and applying it to generated 3D augmented models is discussed by the researchers of HIT Lab New Zealand [22]. This work, in fact, is the major source of inspiration for our study.

It is also reported that due to the attractiveness of Augmented Reality, it can be ranged from advertising to edutainment, education to engineering, medicine to industrial manufacturing [23]. In basic applications, like in advertisement or games, users only see the actions or interact with part of the screen designed to initiate some actions. In order to make users have realistic experiences, the interaction between virtual objects in Augmented Reality must be restricted to the laws of Physics and Virtual objects can have their own dimensions, volumes or weights. In [23], Unity 3D game engine is used with Vuforia platform [8]. To test the integration and concept of Unity engine with AR library, 8 pieces of virtual 3D puzzle modules were created using 8 markers. Each virtual module was assigned to physical properties such as dimensions, shapes, and positions. When the puzzle is assembled, each piece of the marker must be able to move around so that the virtual modules can fit each other. By lifting and rotating the markers, the virtual module will snap with the other proper virtual part, forming virtual 3D puzzle. With this experiment, users had a realistic feeling on assembling the virtual model. It was concluded that this integration concept can be implemented for experiments that are expensive to setup. Experiments related to interaction between objects such as physics and chemistry experiments, engineering and medical training are the main targets for using this kind of technology.

Furthermore, real-time environment Unity3D related augmented reality games have also been developed. In one of such games [24], the player finds and defuses virtual calory bombs in a real world environment. It is also observed by the authors that working with Unity3D for augmented reality based solutions was far more simplified than the previous version of the game which was developed without using any third party game engine. It was also observed that mixed-method usability evaluation by different personnel indicated that interaction with AR content and user interface clarity were improved in the Unity 3D version. The authors conclude and expect that game engines such as the Unity 3D will become essential for AR game development in the future.

After having presented an overview of augmented reality and few of its applications, we present the requirements specifications in the next chapter.

Chapter 3

Requirement Specifications

This chapter presents the details on the requirement specificiations of our system.

3.1 Existing Systems

As discussed earlier (Section 2.3) the key inspiration of our work is the research of HIT Lab New Zealand [22]. The researchers at HIT Lab New Zealand have carried out a significant work related to the Interactive Augmented Reality Coloring Books. They have discussed challenges regarding every aspect for the completion of this application but in an abstract way. They have discussed the processing of video frames like the removal of color, which is basically helpful for the good detection of image target and then it's different sub-parts, which are then applied separately to the corresponding 3D model. They also discussed the Image Registration and suggested to take consecutive five frames to make sure we get the frame without any motion-blur defects. After successfully performing first two procedures, the texture extraction is trivial.

3.2 Proposed System

Inspired by the idea proposed by the HIT Lab New Zealand Researchers and in one of the research papers published by Disney [18, 25], we intended to create such an interactive application which would create new directions in the field of Computer Vision and explore the ability and attractiveness of Augmented Reality.

3.3 Software Requirement Specification

3.3.1 Introduction

Following section is the detailed version of the Software Requirement Specification (SRS) of the proposed system.

3.3.1.1 Purpose of document

The purpose of the document is to describe the details of Coloring Book with Augmented reality. This document is to explain the working, specifications, functionality and constraints under which the system will operate and how it will behave with the external environment. This document describes all functional and non-functional requirements of the system.

3.3.1.2 Document Conventions

The document is built using Latex with the following conventions.
$MainSections, SubSections, Othertextexplanation$

- Document class = 11pt, a4paper, twoside, openright.

3.3.1.3 Intended Audience and Reading Suggestions

1. **Non-Technical**: The non-technical users of the Coloring Book with Augmented Reality will get a clear idea of the application and hardware requirements which are to be there.

2. **Students**: The project laid down various development directions in the fields of Computer Vision and Augmented Reality offering a rich perspective of development for the students.

3. **Developers**: Project developers have an advantage to further explore what can be achieved after this successful exploration of interesting Computer Vision field.

3.3.1.4 Product Scope

The application is designed to be as robust as possible and easy to use. However, due to the nature of the application, we need to impose certain constraints to allow better recognition of templates and subsequent rendering of objects. We consider black and white line art pages on the drawing book and the images of these pages will be stored in the database. The application will read and recognize pre-defined sketches/drawing which will be provided in its database. Naturally, if an image is presented which is not in the database, the application will not be able to render its model.

We have mainly considered producing the augmented object for the uncolored and later the colored drawing, once the object is augmented, a user can play with the animations provided.

3.3.2 Overall Description

3.3.2.1 Product Perspective

The product is primarily designed for Android platform implemented using Unity3D cross platform functionality.

It is furthermore implemented using Augmented Reality Software Development Kit Vuforia for Unity3D and Computer Vision library i.e. OpenCV for Unity.

3D models in this application are developed, animated and UV-mapped using Autodesk Maya Modeling Tools.

3.3.2.2 Product Features

Following are the main features of the Coloring Book with Augmented Reality.

- Interactive user interface.
- A user-friendly environment to see a 2D object in a 3D environment.
- Multiple sketches for children to color.
- Complex 3D models with their UV-maps which provides a real feeling of transformation from 2D to 3D environment.
- Once colored, availability to map children's/user imaginations in a real environment.
- Coloring and mapping with the 3D environment can be done concurrently.

3.3.2.3 User Characteristics

Target users for this application are children of the age where they learn the difference between colors. With the help of this application, they can not only learn the difference between the colors but also map their colored character imagination in the real environment and also see the particular character's movement with provided animations and interactivity.

3.3.2.4 Operating Environment

The application can be operated anywhere conforming to certain constraints. Since we target the Android platform, the application can be used in different Android camera devices like tablets, mobiles.

The application needs the coloring book templates to be present on the device while using the application as it recognizes different target images provided in its database and augments the corresponding model. Furthermore, for the sake of recognition, target image need to be visible in the camera frame i.e. good lighting.

3.3.2.5 Design and Implementation Constraints

Other than GUI menus, the application is all about camera feature, therefore, during the real-time interaction with the augmentation, we have to be restricted providing UI objects on the camera frames for a good usability.

Furthermore, the SURF and SIFT feature extractors are non-free, therefore, we have to use the API provided by Vuforia itself and the third-party wrapper of OpenCV for Unity which is not supported by OpenCV developers themselves, therefore it lacks some functionality which is native in C++/Java. As the application is being developed in Unity platform, therefore solutions provided in languages other than C#/UnityScript produce uncertain behavior at times, therefore proper conversion of code to UnityScript becomes a challenge.

3.3.2.6 User Documentation

The product has been implemented and user documentation will be provided with the completion of all the evaluation phases.

3.3.2.7 Assumptions and Dependencies

For the application to provide its functionality, there are certain assumptions for it to run smoothly. One core requirement is that the mobile must be running on Android platform and it must have a camera. Secondly, it must have enough performance to handle real-time processing as it requires a good processor to do real-time camera frames processing to extract and apply textures to the 3D models otherwise the application may lag on certain points.

It is also assumed that the user has a good mental model of Android native applications i.e. how to use basic applications and the camera device. The experience required related to using the camera device simply involves knowledge of how to hover a camera over a target image.

3.3.3 External Interface (EI) Requirements

This section provides the details of all the inputs to the system and outputs from it. It also describes the hardware, software and communication interfaces and provides the basic theoretical description of the user interface.

3.3.3.1 User Interface (UI)

As far as user interface is concerned, we would like to provide as less control to the user as we can. Due to the nature of application and its target users, total functionality of the application is performed by itself.

There are two phases of the user interaction with the application. First, the splash and menu screen. We would like to provide a good interface at the application's start-up and for the sake of children's attraction. There would be a number of interactive buttons i.e. Start, Pause, Gifts, Settings and Exit with attractive animations and background. Our main focus would be on the start button right now, once the user clicks the *Start* button, another scene is to be opened. It should be noted that these interfaces would be managed by Unity Scenes. The other scene would be the augmentation scene. The mobile camera would be opened by tapping on the *Start* button it would start looking for the Image Targets. In this scene, there would be most likely only one *Back* button which would take us back to the menu screen. As discussed earlier in the section 3.3.2.5, we need to be restricted providing UI components on the camera scene for a good user experience.

3.3.3.2 Hardware Interface (HI)

Since the mobile application does not have any designated hardware, it does not have any direct hardware interface. The camera functionality is managed by the application itself and the hardware connection to the camera, screen and user interaction interfaces like hard buttons are managed by the underlying operating system on the mobile phone.

3.3.3.3 Software Interface (SI)

The application integrates different tools/libraries together in Unity3D i.e. Vuforia SDK, Autodesk Maya 3D models, OpenCV programming libraries. Unity provides the interface to interact with all the tools in a simplified way. Image targets are handled by Vuforia and the augmentation as well. Autodesk Maya provides the functionality to directly export the projects into Unity project or it can just generate Maya Binary files and Unity supports this *.mb* file Maya generates and it is then integrated with Unity. OpenCV package for Unity is easily loaded into Unity and its libraries can be then used in any UnityScript by just adding the line 'using OpenCVforUnity'.

3.3.3.4 Communications Interface (CI)

The communication between the different modules of the application like 3D-Models, Image Targets, and image processing part is important since they depend on each other. Communication between these modules is discussed in discussed later in Section 5.4.

3.3.4 Other Nonfunctional Requirements

3.3.4.1 Performance Requirements

The important aspect of the Coloring Book software is the time constraint. As our software system is real time and hence should do the processing in minimum time and the colored augmentation is a core feature and this could only be assured if the system is working in full capability. So uninterrupted battery power supply, good camera, and processor are needed.

3.3.4.2 Safety Requirements

As the application is not going to be open-source, therefore certain measures would be taken for taking care of it i.e. development methods will be discussed but its source code won't be disclosed. *apk* file for the installation will be available publicly.

3.3.4.3 Security Requirements

As this application is not administered using any user accounts i.e. any user of any age can use this application because this application does not store or provide any personal information about anyone to anyone respectively, therefore there are not any designated security requirements.

3.3.4.4 Software Quality Attributes

- **Availability**: The application will be available publicly and also available on Google Play Store.
- **Flexibility** : The application will be flexible enough for some later requirements change or features enhancement and further expansion of models and features.
- **Usability** : The interface and GUI design of the application will be user-friendly with no training required to use it, except just one tutorial.

3.3.5 Other Requirements

Some of the other particular non-functional attributes required by the system are listed below.

3.3.5.1 Accuracy

The application needs to be accurate enough that at first it should detect the right Image Target and select the corresponding template. Next, it should calculate the right area of the Image Target from the environment so to be able to generate colors and to apply them correctly on a 3D model.

3.3.5.2 Maintainability

The maintenance of the application will be carried out time to time.

3.3.5.3 Portability

As this is the Android mobile application, therefore it would be portable provided there are Image Targets which it detects.

3.4 Use-Cases

3.4.1 Use-Case 1

Use Case in Figure 3.1 describes the processes from starting the application to starting augmentation, till exiting from application while more details are presented in Table 3.1.

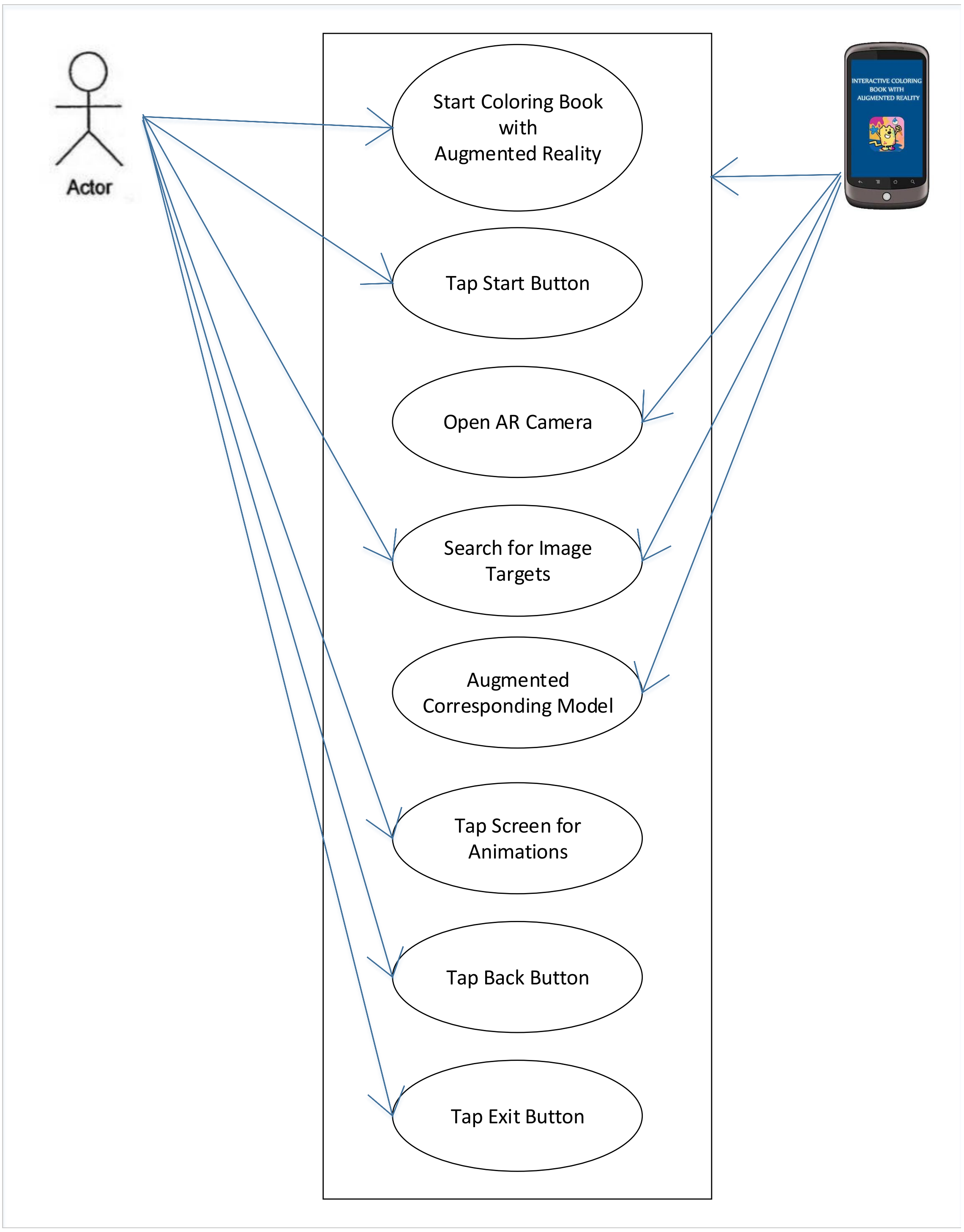

Actor
Start Coloring Book with Augmented Reality
Tap Start Button
Open AR Camera
Search for Image Targets
Augmented Corresponding Model
Tap Screen for Animations
Tap Back Button
Tap Exit Button
INTERACTIVE COLORING BOOK WITH AUGMENTED REALITY

Figure 3.1: Use Case 1

Use Case Name	**Major Application Processes.**
Scenario:	Running Application from start.
Trigger Event:	This use case will be triggered when app-icon will be tapped.
Brief Description:	This use case represents the user interacting with the application by opening it with the app-icon, interacting with GUI provided.
Actors:	Application User, Application.
Stakeholders:	Users, developers.
Pre-condition:	The application needs to be installed first.
Post-condition:	None
Flow of Activities:	1. User starts the application by tapping the app icon. 2. User taps the Start button. 3. AR camera will be opened by the Application. 4. User moves around looking for Image Targets, the Application, on the other hand, looks for the Image Targets in Camera Frames. 5. When found, Application augments the corresponding 3D-model. 6. 3D-models gets animated if user touches/taps the screen. 7. If the user taps the Back button, the previous menu will be displayed. 8. Application exits if the user taps the Exit button.
Expectation Condition:	There can be an alternative flow of activities as well which are described in other use cases, here it is expected for the user to flow these major application processes.

Table 3.1: Use case 1 : Fully Dressed Description

Chapter 4

Design

In this chapter, Coloring Book with Augmented Reality is over-viewed from every aspect related to its design.

4.1 System Architecture

Coloring Book with Augmented Reality is a real-time application program. It works with the live camera video and processes its every frame to look for the Image Targets.

The core functionality of our application is in between detecting an Image Target in a camera frame till the augmentation of the corresponding 3D model. The processes involved in between are illustrated in the Figure 4.1.

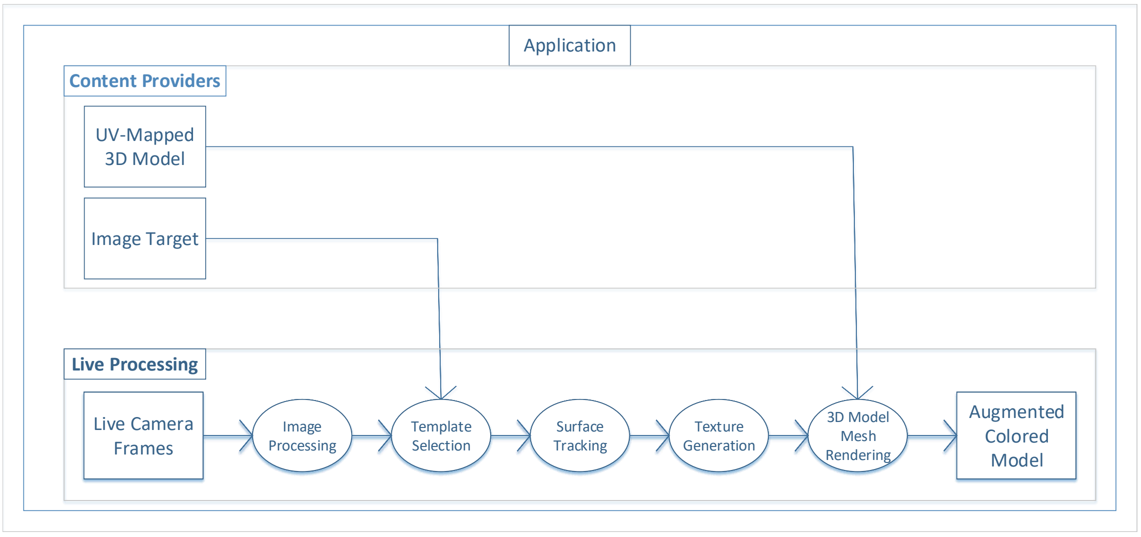


Figure 4.1: Basic System Architecture

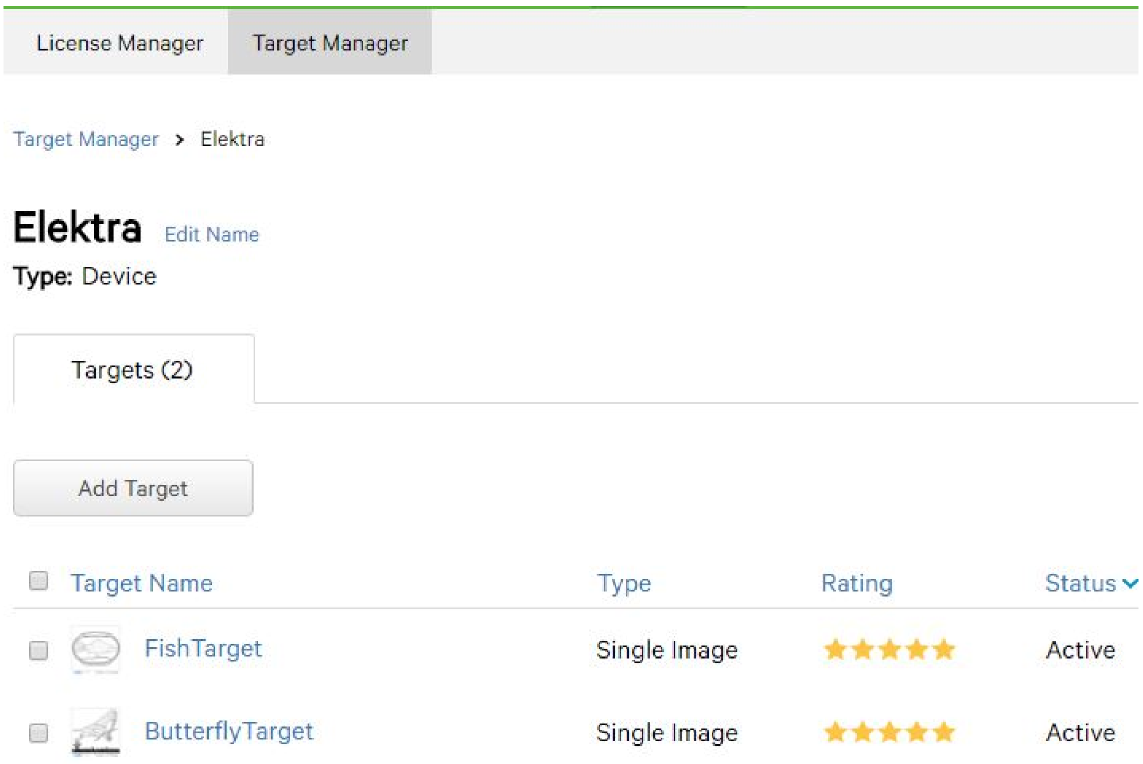
As it can be seen from the Figure 4.1, the application's architecture is divided into two major phases discussed in the following.

4.1.1 Content Providers

A content provider is the backbone of the application. It provides the Image Targets which are to be detected in the camera frame and once detected, its model is augmented which is UV-mapped.

4.1.1.1 Image Targets

Image Targets in Coloring Book with Augmented Reality are the artistic sketches of different characters which are provided in the book. These are at first created in the Adobe Photoshop tool as an image then using the Vuforia's developers portal online, these are uploaded in the Target Manager and processed by Vuforia for features extraction. Vuforia's online portal ranks the uploaded target considering its detection power and it can be seen for two samples in the Figure 4.2.



Elektra

Type: Device

Target Name	Type	Rating	Status
FishTarget	Single Image	★★★★★	Active
ButterflyTarget	Single Image	★★★★★	Active

Figure 4.2: Target with their Ranks

Vuforia furthermore provides the availability to download this image as an Image Target with Unity Package. Unity package of Image Target is integrated into Unity and then our Image created in Photoshop works as an Image Target which the live camera is

bound to detect.
Sample features detected for the Figure 4.3 are shown in the Figure 4.4.

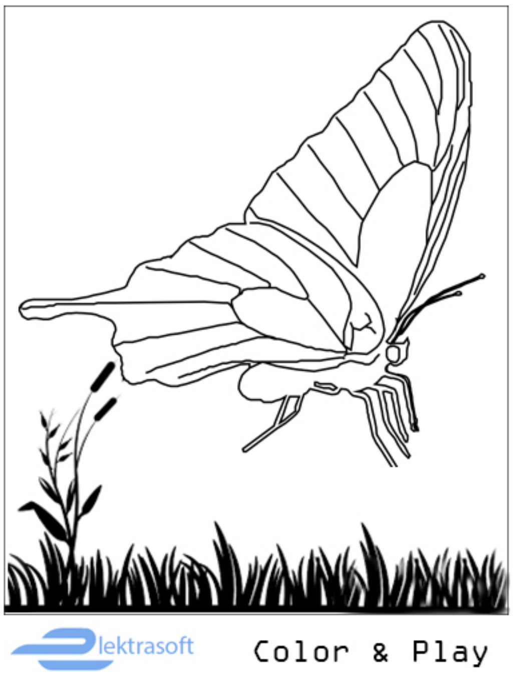


Figure 4.3: Butterfly Image Target

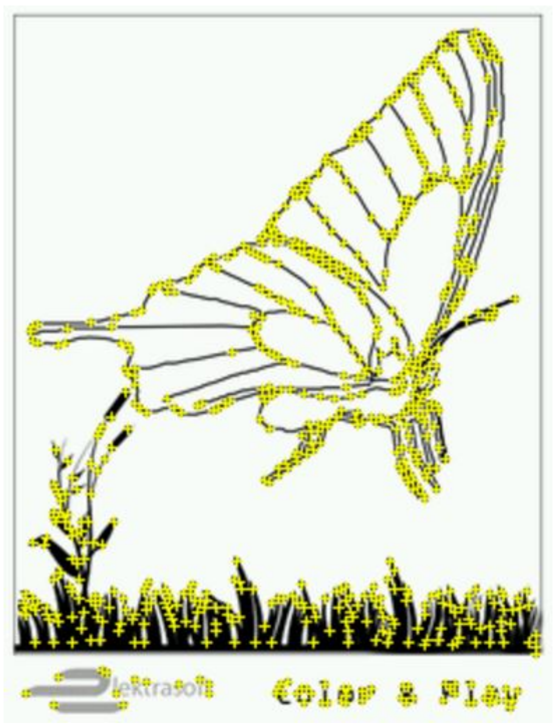


Figure 4.4: Features Detected

4.1.1.2 UV-Mapped 3D Models

UV-mapped 3D model means that where should the pixel at certain x, y coordinate of 2D texture lie in 3D object's mesh. That is why the 3D model is at first UV-mapped in Autodesk Maya according to the Image Target. When we have both of them i.e. Image Target and its corresponding UV-mapped 3D model, we supply the Image Target to be detected in camera frames and a 3D model which is then augmented when detected.

In the Figure 4.5, the *Butterfly* 3D model is being UV-mapped. It can be seen that the model is being UV-mapped with the same image target provided. Here we decide where to put which part of the Image Target on 3D model which helps later to map the different regions of colored Image Target onto 3D model.

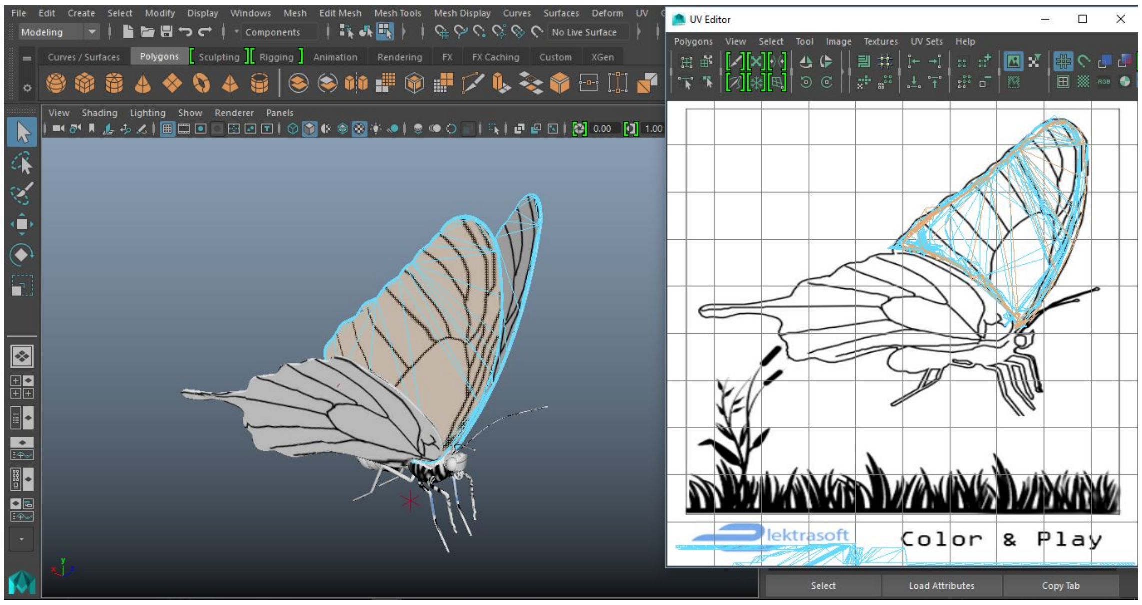

File Edit Create Select Modify Display Windows Mesh Edit Mesh Mesh Tools Mesh Display Curves Surfaces Deform UV
Modeling
Components
No Live Surface
Curves / Surfaces Polygons Sculpting Rigging Animation Rendering FX FX Caching Custom XGen
View Shading Lighting Show Renderer Panels
UV Editor
Polygons View Select Tool Image Textures UV Sets Help
lektrasoft
Color & Play
Select
Load Attributes
Copy Tab

Figure 4.5: Butterly being UV-mapped

4.1.2 Live Processing

Live processing is the major technical and critical part of the project.

4.1.2.1 Image Processing

Image processing is the process where the camera frames are being fetched and processed for the detection purposes, here Vuforia SDK supports this process and detects the Image Target we have provided in the camera frames.

4.1.2.2 Template Matching

This is the process where features are detected while comparing the features of provided Image Target with the features in the camera frame, once these are successfully matched, we get to know which model is to be augmented.

4.1.2.3 Surface Tracking

Before directly augmenting the corresponding 3D model, we track the surface of the Image Target in the camera frame for color extraction purpose. Applying certain programming techniques we can get the position as x, y coordinates and we can assumingly generate the bounding box area of the Image Target knowing its width and height. Here we have the matrix which has our region's color information.

4.1.2.4 Texture Generation

When we have extracted the matrix of the Image Target's bounding box, from here on we convert our matrix into Texture2D to generate the texture for applying to 3D model purposes because simple matrix cannot be used a texture.

4.1.2.5 3D Model Mesh Rendering

When the texture has been created, we fetch our model's mesh from the assets and apply the processed texture. Here we have to take care for fetching the right rendering mesh and apply the texture as the $MainTex$.

It should be noted that this whole live process is in between detecting an Image Target and augmenting its corresponding 3D model, therefore this live process is critical and also needs to be fast.

4.2 Design Constraints

As the main focus of developing the application is to be able to augment the customized colored object and we are using Vuforia SDK for the augmentation purpose, therefore, making certain changes in the this SDK's methodology is the main constraint. Fortunately, using Unity3D and Vuforia together, we can accomplish the desired objective.

4.3 Design Methodology

The application will be created using the pure object-oriented techniques. As the system architecture explains itself, there are different components/classes to be used for the completion of the objective.
The design of the application will be developed in such a way that it can be evolved, other than the *liveprocessing* phase, which is going to be the constant process. The application will evolve using the *contentproviders* phase. The purpose for making this phase separate is the only reason to evolve our application by adding more models with image targets later after the launch.

As the procedure used for the extraction of texture will be adaptable i.e. same procedure could be used for any other model, therefore we can provide as many models as we want without changing the texture extraction methodology.

This can be done using the UV-mapping of the new models with their image targets. When the model is correctly UV-mapped, we just have to provide it with image target to our application.

The texture extraction procedure is adaptable because, during the image processing part of live processing, we just extract our image target from the camera frame i.e. the cropped part from the environment. From here on, when we apply this cropped part as a texture to the 3D meshes, the model itself maps different sub-portions to its correct place due to UV-coordinates already defined of the image targets.

4.4 High Level Design

This section describes in further detail elements discussed in the Architecture. Typical viewpoints are

- Conceptual View
- Activities Overview
- Process Interaction Design, which is usually referred as System Sequence Diagram (SSD).
- Physical Design.
- Modules Design.
- Security Design.

4.4.1 Conceptual View

The view in the Figure 4.6 shows the logical functional elements of the system. Each component represents a similar grouping of functionality.

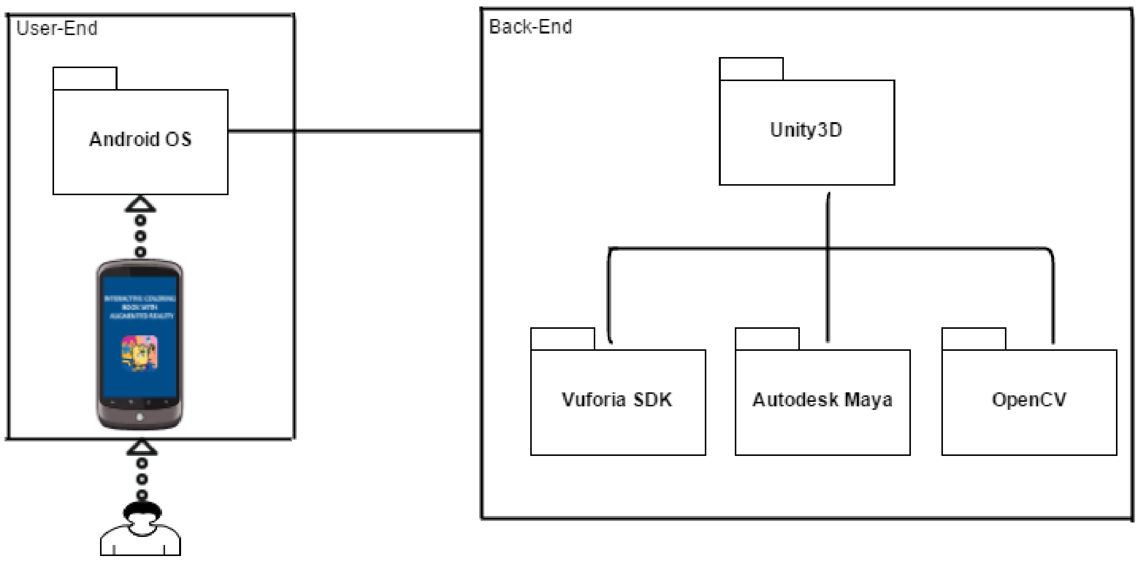


Figure 4.6: Package Diagram

4.4.2 Process Interaction Design

Run-time view of the application is shown in the Figure 4.7. The components are threads or processes or distributed modules.

4.4.3 Physical Design

Physical design refers to the deployment of the application. This application will be deployed to Android Operating System over the internet through Google Play Store as illustrated in Figure 4.8.

4.4.4 Activity Diagram

Major user inputs and the corresponding system's response have been illustrated in the Figure 4.9.

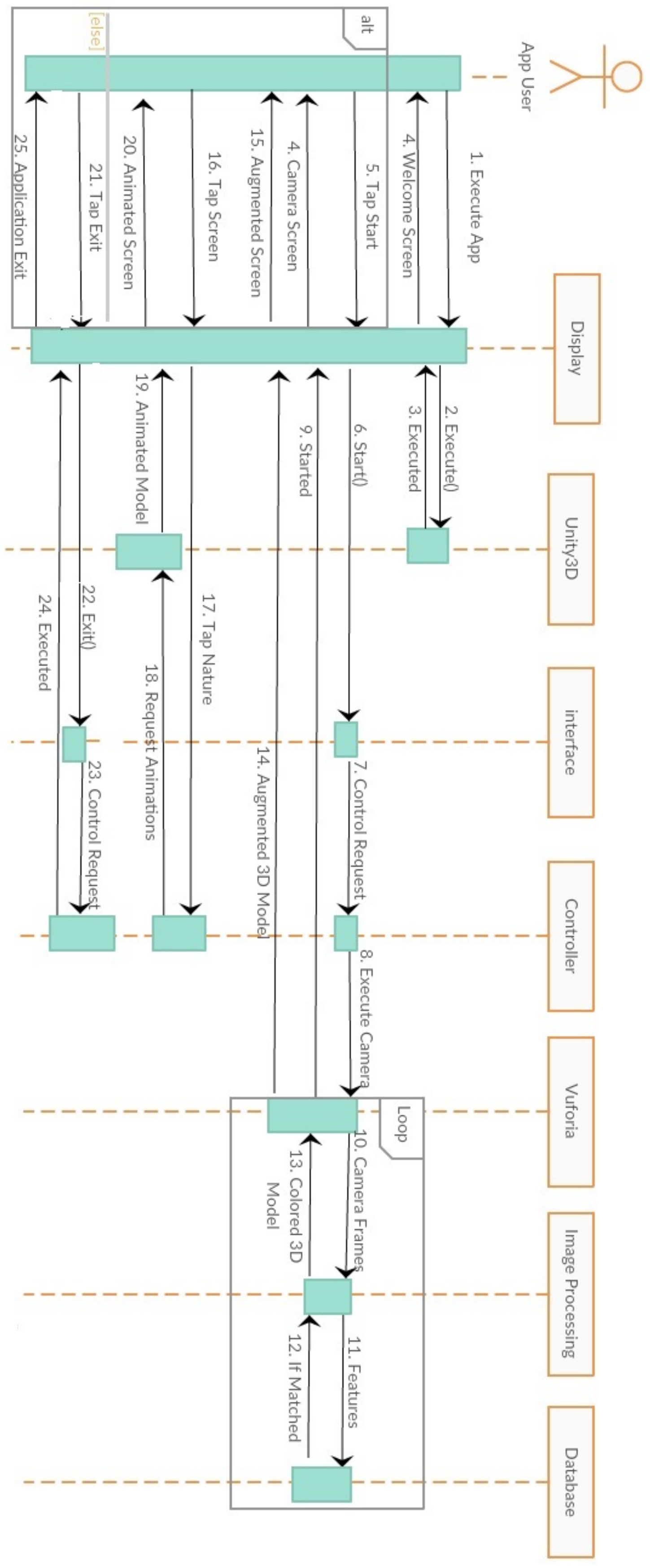


Figure 4.7: System Sequence Diagram

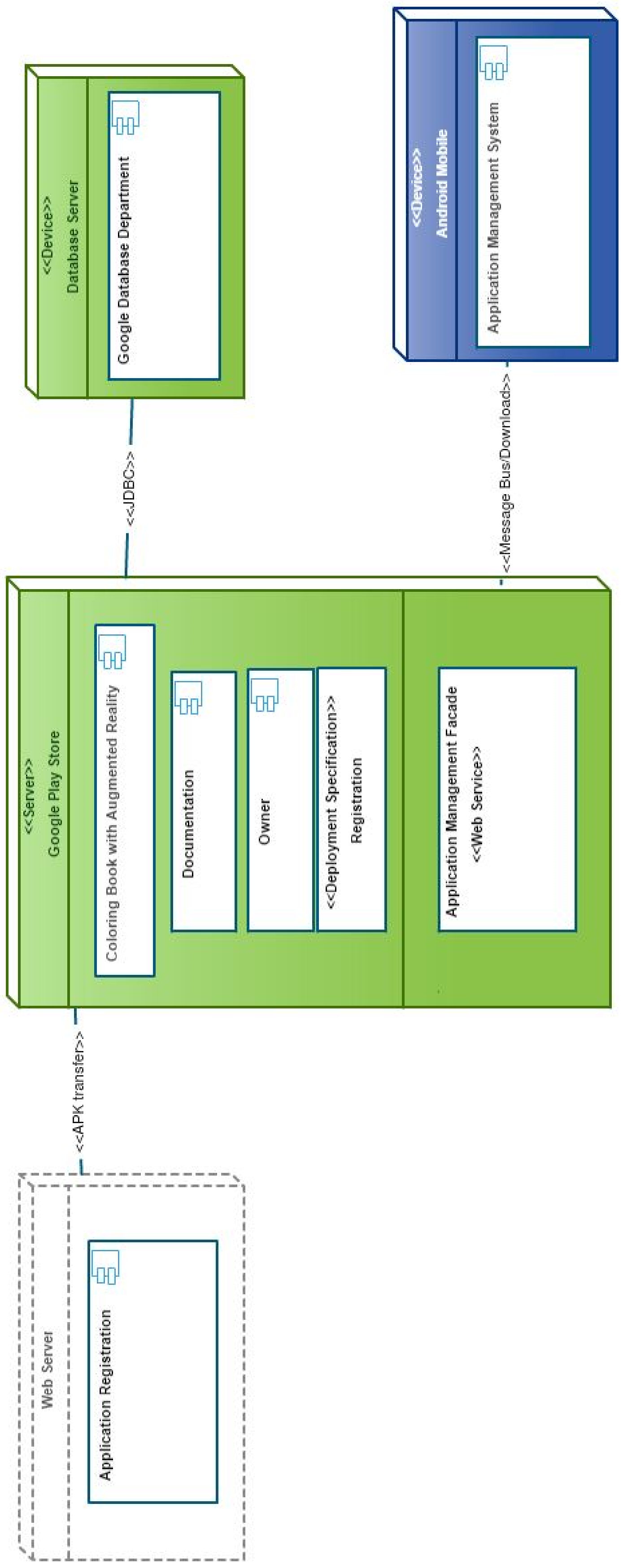


Figure 4.8: Deployment Diagram

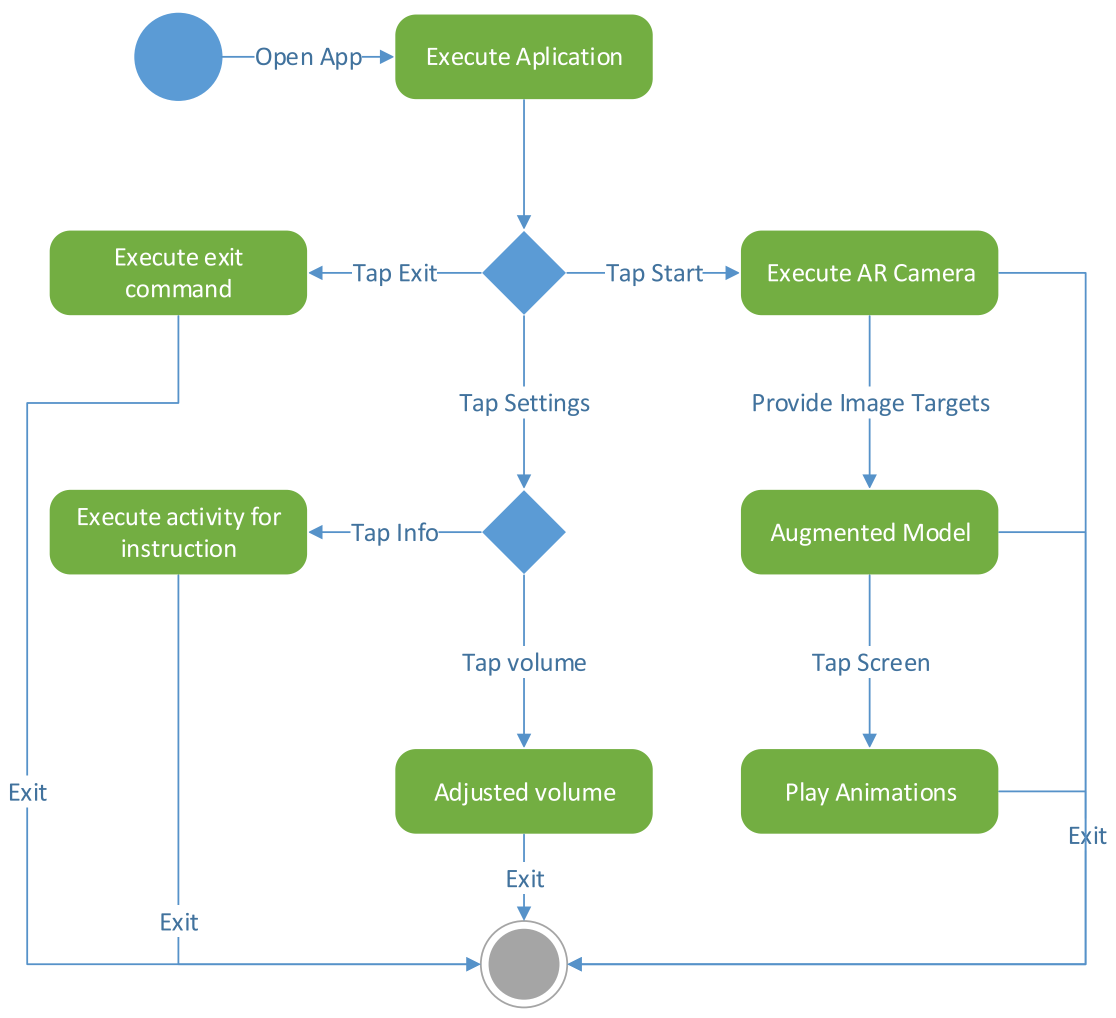
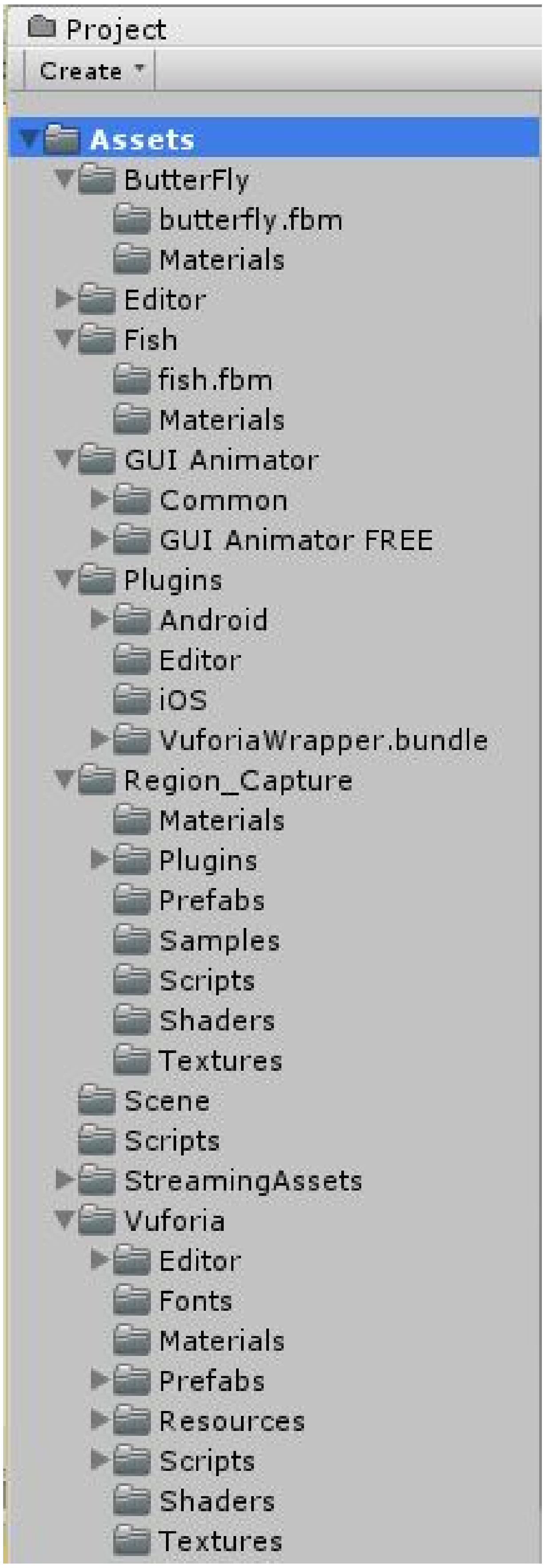


Figure 4.9: Activity Diagram

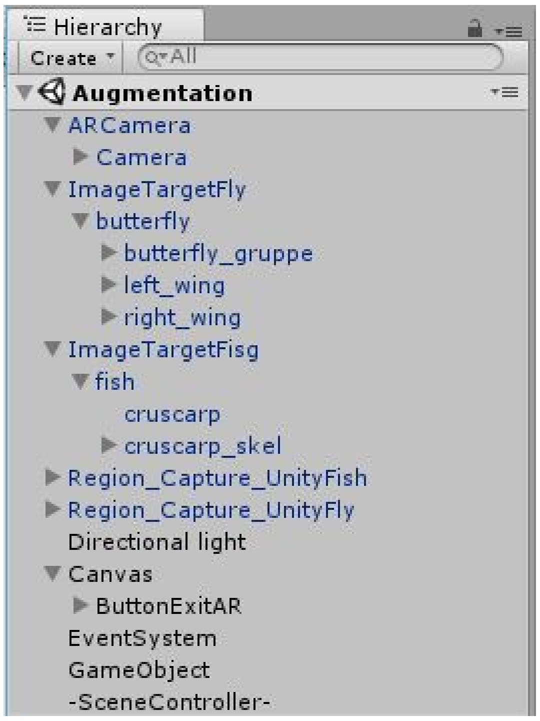
4.4.5 Modules Design

This view is for project management and code organization. Files and directory structure of Animation scene, Augmentation scene, and Assets of two sample models are shown in the Figure 4.10, Figure 4.11 and Figure 4.12, respectively.



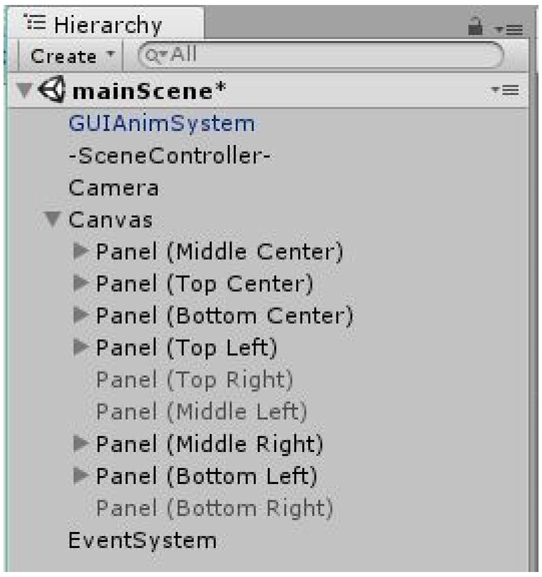
Project
Create
Assets
ButterFly
butterfly.fbm
Materials
Editor
Fish
fish.fbm
Materials
GUI Animator
Common
GUI Animator FREE
Plugins
Android
Editor
iOS
VuforiaWrapper.bundle
Region_Capture
Materials
Plugins
Prefabs
Samples
Scripts
Shaders
Textures
Scene
Scripts
StreamingAssets
Vuforia
Editor
Fonts
Materials
Prefabs
Resources
Scripts
Shaders
Textures

Figure 4.10: Assets Hierarchy



```
Hierarchy
Create ▾  Q▾All
▼ Augmentation
  ▼ ARCamera
    ▶ Camera
  ▼ ImageTargetFly
    ▼ butterfly
      ▶ butterfly_gruppe
      ▶ left_wing
      ▶ right_wing
  ▼ ImageTargetFisg
    ▼ fish
        cruscarp
      ▶ cruscarp_skel
  ▶ Region_Capture_UnityFish
  ▶ Region_Capture_UnityFly
    Directional light
  ▼ Canvas
    ▶ ButtonExitAR
    EventSystem
    GameObject
    -SceneController-
```

Figure 4.11: Files Hierarchy of Augmentation Scene



```
Hierarchy
Create ▾  Q▾All
▼ mainScene*
    GUIAnimSystem
    -SceneController-
    Camera
  ▼ Canvas
    ▶ Panel (Middle Center)
    ▶ Panel (Top Center)
    ▶ Panel (Bottom Center)
    ▶ Panel (Top Left)
      Panel (Top Right)
      Panel (Middle Left)
    ▶ Panel (Middle Right)
    ▶ Panel (Bottom Left)
      Panel (Bottom Right)
    EventSystem
```

Figure 4.12: Animation Scene Hierarchy

4.4.6 Security Design

This view is not applicable to our application. However, the application would be deployed with a single *.apk* file as discussed in the next chapters for reserving the copyrights.

4.5 Low Level Design

4.5.1 Class Diagram

Class diagram of major helpful classes can be seen in Figure 4.13

4.6 GUI Design

Coloring Book with Augmented Reality is all about interactivity. The application's GUI has also been designed fulfilling the same purpose.

There are different interfaces designed with interactive animations. Figure 4.14 illustrates the view of a welcoming screen of the application. If the user taps the *Settings* button, it expands with a nice animation with more options like controlling volume, changing sound and information about the use of the application and it can be seen in the Figure 4.15.

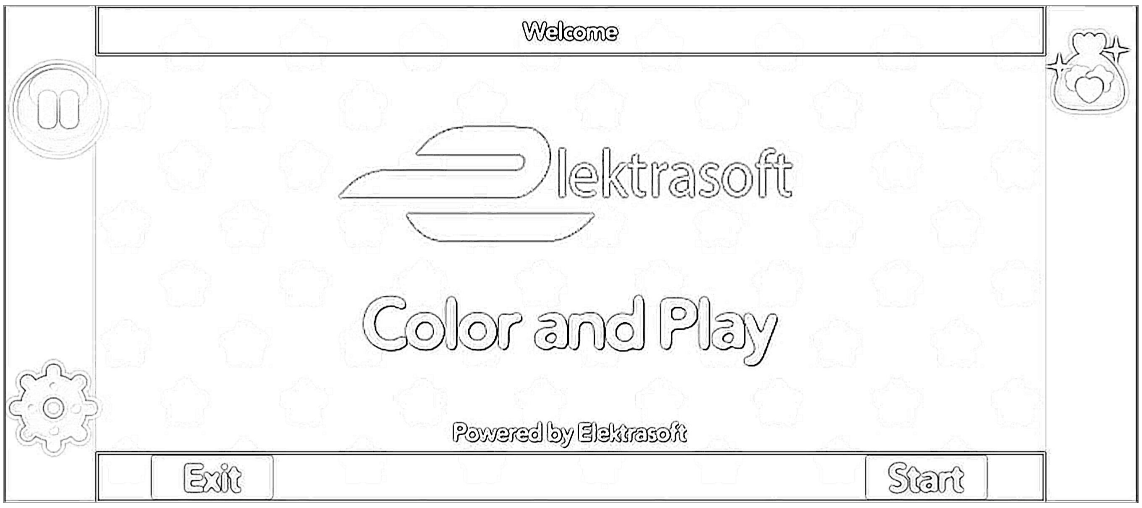


Figure 4.14: Main Screen

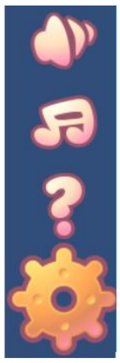


Figure 4.15: Settings View

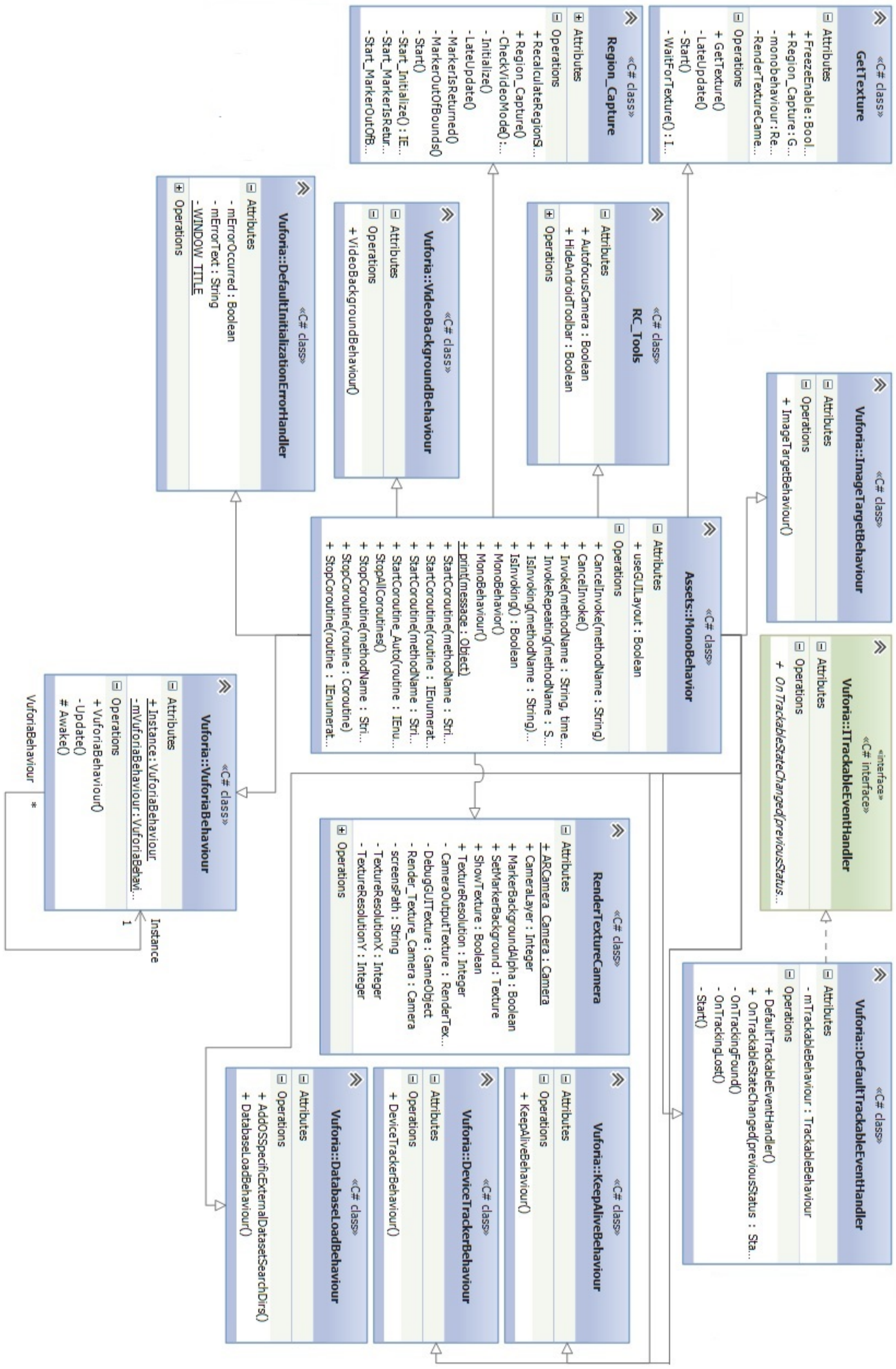

«C# class»
GetTexture
Attributes
+FreezeEnable : Bool...
+Region_Capture : G...
-monobehaviour : Re...
-RenderTextureCame...
Operations
+GetTexture()
-LateUpdate()
-Start()
-WaitForTexture() : I...
«C# class»
Region_Capture
Attributes
Operations
+RecalculateRegionS...
+Region_Capture()
-CheckVideoMode() : ...
-Initialize()
-LateUpdate()
-MarkerIsReturned()
-MarkerOutOfBounds()
-Start()
-Start_Initialize() : IE...
-Start_MarkerIsRetur...
-Start_MarkerOutOfB...
«C# class»
Vuforia::ImageTargetBehaviour
Attributes
Operations
+ImageTargetBehaviour()
«C# class»
RC_Tools
Attributes
+AutofocusCamera : Boolean
+HideAndroidToolbar : Boolean
Operations
«C# class»
Vuforia::VideoBackgroundBehaviour
Attributes
Operations
+VideoBackgroundBehaviour()
«C# class»
Vuforia::DefaultInitializationErrorHandler
Attributes
-mErrorOccurred : Boolean
-mErrorText : String
-WINDOW_TITLE
Operations
«C# class»
Assets::MonoBehavior
Attributes
+useGUILayout : Boolean
Operations
+CancelInvoke(methodName : String)
+CancelInvoke()
+Invoke(methodName : String, time...
+InvokeRepeating(methodName : S...
+IsInvoking(methodName : String)...
+IsInvoking() : Boolean
+MonoBehavior()
+MonoBehaviour()
+print(message : Object)
+StartCoroutine(methodName : Stri...
+StartCoroutine(routine : IEnumerat...
+StartCoroutine(methodName : Stri...
+StartCoroutine_Auto(routine : IEnu...
+StopAllCoroutines()
+StopCoroutine(methodName : Str...
+StopCoroutine(routine : Coroutine)
+StopCoroutine(routine : IEnumerat...
«interface»
«C# interface»
Vuforia::ITrackableEventHandler
Attributes
Operations
+ OnTrackableStateChanged(previousStatus...
«C# class»
Vuforia::DefaultTrackableEventHandler
Attributes
-mTrackableBehaviour : TrackableBehaviour
Operations
+DefaultTrackableEventHandler()
+ OnTrackableStateChanged(previousStatus : Sta...
-OnTrackingFound()
-OnTrackingLost()
-Start()
«C# class»
RenderTextureCamera
Attributes
+ARCamera_Camera : Camera
+CameraLayer : Integer
+MarkerBackgroundAlpha : Boolean
+SetMarkerBackground : Texture
+ShowTexture : Boolean
+TextureResolution : Integer
-CameraOutputTexture : RenderTex...
-DebugGUITexture : GameObject
-Render_Texture_Camera : Camera
-screensPath : String
-TextureResolutionX : Integer
-TextureResolutionY : Integer
Operations
«C# class»
Vuforia::VuforiaBehaviour
Attributes
+Instance : VuforiaBehaviour
-mVuforiaBehaviour : VuforiaBehavi...
Operations
+VuforiaBehaviour()
-Update()
#Awake()
VuforiaBehaviour
*
1
Instance
«C# class»
Vuforia::KeepAliveBehaviour
Attributes
Operations
+KeepAliveBehaviour()
«C# class»
Vuforia::DeviceTrackerBehaviour
Attributes
Operations
+DeviceTrackerBehaviour()
«C# class»
Vuforia::DatabaseLoadBehaviour
Attributes
Operations
+AddOSSpecificExternalDatasetSearchDirs()
+DatabaseLoadBehaviour()

Figure 4.13: Class Diagram

Furthermore, more GUI buttons are also provided with attractive animations like *Gifts*, *Pause* button which can be seen in Figure 4.16.

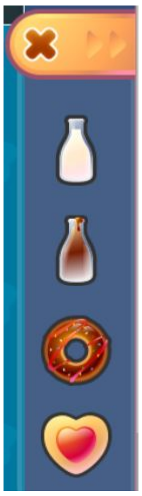


Figure 4.16: Gifts View

and Figure 4.17 respectively.

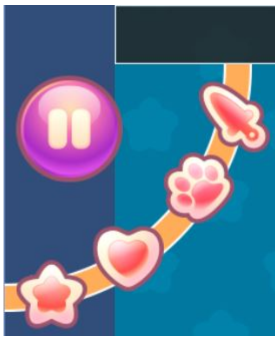


Figure 4.17: Pause View

Although, detailed GUI testing has been carried out discussed later in the Chapter 6, section 6.1, and the referenced design discussed in this current chapter has been revised after considering the feedbacks.

Chapter 5

System Implementation

This chapter represents the implementation details of Coloring Book with Augmented Reality, including Components to be Implemented, Development Environment/Languages used, Tools and Technologies used, Processing Logic, Integration Approach, Implementation Strategies and Conversion Strategy.

5.1 Components of the Application

The application is developed after dividing it into different modules which are developed in different tools and then integrated into one. Table 5.1 shows different components with their description and source of development that in the end integrates into one application.

Components/Modules	**Description**	**Source**
Image Targets	Image Targets are computerized sketches of character to be detected.	Image Targets are to be made with Photoshop/Illustrator.
Image Targets - Unity Package	Image Targets Image Target made with Photoshop uploaded to Vuforia's Target Manager online and generating Unity Package for it.	Vuforia's Target Manager.
3D models	3D models of different characters are to be made.	Autodesk Maya.
3D models (UV-mapped)	UV-mapping of 3D characters to be defined with created Image Target.	Autodesk Maya.
3D models (Animations)	Final stage of 3D models is to define its proper animations for good interactivity.	Autodesk Maya.
Image Processing	Classes to provide Live Image Processing, i.e. to detect area of Image Target from camera frames and apply that to our 3D model.	Unity 3D.

Table 5.1: Components of the Application with Descriptions

5.2 Integrated Development Environments (IDE)

This section introduces the Integrated Development Environment which is used for the development of the application. We have employed Unity 3D to develop the application for Android devices. Unity 3D is specifically chosen for the development of the application due to its compelling environment of development, which expands from simplicity, the power of an integration of different third party software like 3D models from Autodesk Maya.
Unity 3D also supports multi-platform applications. In simple words, one needs to just familiarize with the IDE of Unity 3D and can build the application for different platforms like Android, IOS, Windows, Linux, AppleTV, Nintendo 3DS line, OS X, Unity Web Player (including Facebook), Wii, Wii U, Windows Phone, Windows, Xbox 360, and Xbox One.

5.2.1 Versions and Deployment Types

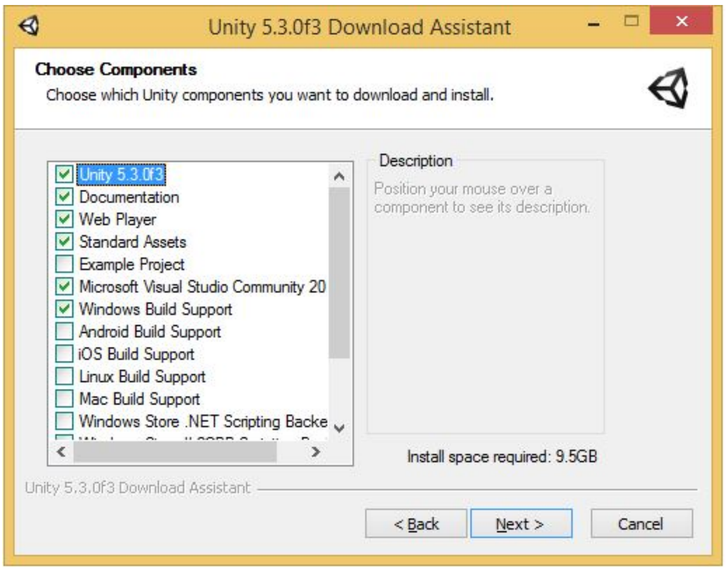
Unity 3D is now available with its latest version series with its massive update, Unity 5.X. It includes features like Enlighten real-time lighting system and physically-based shader which provide the power to render high-quality objects and effects.
Unity 3D comes for different types of users and deployment types.

1. Personal Edition - It is free for educational purposes.
2. Unity Plus - It costs $35 per seat/month.
3. Unity Pro - It costs $125 per seat/month. It is for professionals looking for profit from advanced customization features provide.
4. Unity Enterprise - It is the tailored solution to suit the organization's goals.

5.2.2 Downloading and Installing Unity

Unity 3D provides different versions of software i.e. starting from 3.X and now the 5.X series is available.
Unity 3D can be downloaded in different ways from the internet.

- **Unity Download Assistant -** It is a small .exe program of about 1 MB in size which lets users decide and select which components of Unity are to be downloaded and installed. The download assistant then does the rest as depicted in the Figure 5.1, if one does not know which components to install, the default option can be selected.


Unity 5.3.0f3 Download Assistant

Choose Components

Choose which Unity components you want to download and install.

- [✓] Unity 5.3.0f3
- [✓] Documentation
- [✓] Web Player
- [✓] Standard Assets
- [ ] Example Project
- [✓] Microsoft Visual Studio Community 20
- [✓] Windows Build Support
- [ ] Android Build Support
- [ ] iOS Build Support
- [ ] Linux Build Support
- [ ] Mac Build Support
- [ ] Windows Store .NET Scripting Backe

Description

Position your mouse over a component to see its description.

Install space required: 9.5GB

Unity 5.3.0f3 Download Assistant

Figure 5.1: Unity Download Assistant

- **Installing Unity without the Download Assistant -** Unity 3D can be downloaded with its download assistant. This method can be helpful for the new users who are used to install any windows application with their .exe files, Unity also provides its different components and packages as separate .exe files.
- **Torrent Download -** Users who prefer to download Unity via Torrent can download the BitTorrent file from the Unity download archives. It should be noted that not all versions have the capability to be downloaded via Torrent. The Unity Website's Torrent Download can be viewed in the Figure 5.2

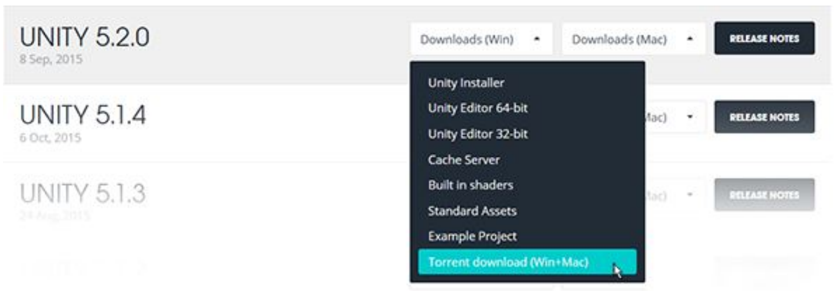


Figure 5.2: Unity Torrent Download

- **Installing several versions at once -** Unity .exe files of its different versions does not effect each other, i.e. if one installs another version, they can keep the previous versions and can explicitly remove them as well.

  On a PC, the install folder is named Unity X.Y.Z[fp]W, where the 'f' is for an official release, and 'p' shows the patch release.

5.2.3 Android SDK Setup

No matter which approach we are using for building Android Application, from the scratch or with Unity 3D, we need to set up Android SDK for running application on Android devices. Following are the step to download and set up Android SDK.

1. **Download the Android SDK -** Android SDK can be downloaded from Android Developer SDK website.

2. **Install the Android SDK -** We can follow the steps provided in the Developer's website and can leave the optional steps for installing Eclipse. We need to be sure to install at least one Android Platform with API equal or above 9 (Platform 2.3 or greater).

3. **Add the Android SDK path to Unity -** If Unity fails to recognize/locate the SDK, it would ask us give the Android SDK path where We installed the Android SDK. The location of Android SDK can also be explicitly defined, in the menu bar go to Unity > Preferences, then click External Tools.

5.2.4 Languages supported and used

As Unity 3D is a multi-platform supported Game Engine, it also supports more than one languages.

- C#

- JavaScript - Also known as UnityScript, in fact, it is not the real JavaScript but follows the syntax same as it is.

- Boo

In the article [26], according to editor analytic statistics, the percentage of three different supported languages break down like shown in the Figure 5.3

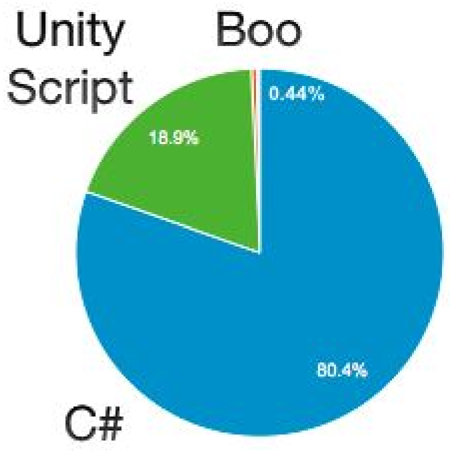


Figure 5.3: Unity Languages Usage (Courtesy: Aleksandr)

Due to very less usage of *Boo*, Unity has decided to stop giving support of this language in the documentation as well as in development environment from version 5.X and allocate these resources in a more constructive way. Now "Create Boo Script" has been dropped down.

Due to well familiarization with C#, we preferred to choose it. Also due to its vast support provided with the documentations and tutorials.

5.2.5 Compiler/Development Platform

Unity 3D supports different development platforms in which the code can be compiled. For example for C#, we can either use Visual Studio or Mono.
Coding and compiling in Visual Studio becomes heavy Unity 3D and Visual Studio runs seperately but of course are integrated, but to running separately, Visual Studio takes a lot of processor and RAM space to run which ultimately slows down the whole development environment.
On the other hand, Mono is a light weighted development platform which is based on .NET Framework and it also allows cross-platform building.
Following are the components of Mono:

- C# Compiler
- Mono Runtime
- .NET Framework Class Library
- Mono Class Library

Most highlighted features of Mono are:

- **Multi-Platform:** Runs on Linux, BSD, OS X, and Microsoft Windows.
- **Multi-Language:** Develop in C# 4.0, VB 8, Python, Java, Eiffel, Ruby, F# Oxygene and others
- **Binary Compatible:** ECMA's Common Language Infrastructure and C#.
- **Microsoft Compatible API:** Runs ASP.NET, ADO.NET, Silverlight and Windows.Forms applications.
- **Open Source, Free Software:** Distributed using the MIT license.
- **Comprehensive Technology Coverage:** Bindings and managed implementations of many popular libraries and protocols.

5.3 Tools and Technology

Following tools and technologies have been used:

- Adobe Photoshop
- Adobe Illustrator
- Unity 3D
- Autodesk Maya
- Vuforia AR SDK
- OpenCV

5.4 Integration Approach

As discussed earlier, (Section 5.1), the system is divided into different components/modules. In this section, the integration of all these modules will be discussed i.e. Image Targets with its Unity Package and importing it in Unity 3D, 3D model made with Autodesk Maya with its different integration methods with Unity 3D and other third party libraries used.

5.4.1 Image Targets Integration

Image Targets are the computerized sketches of different characters. They are made using Adobe Photoshop, a sample Image Target being made is shown in Figure 5.4.

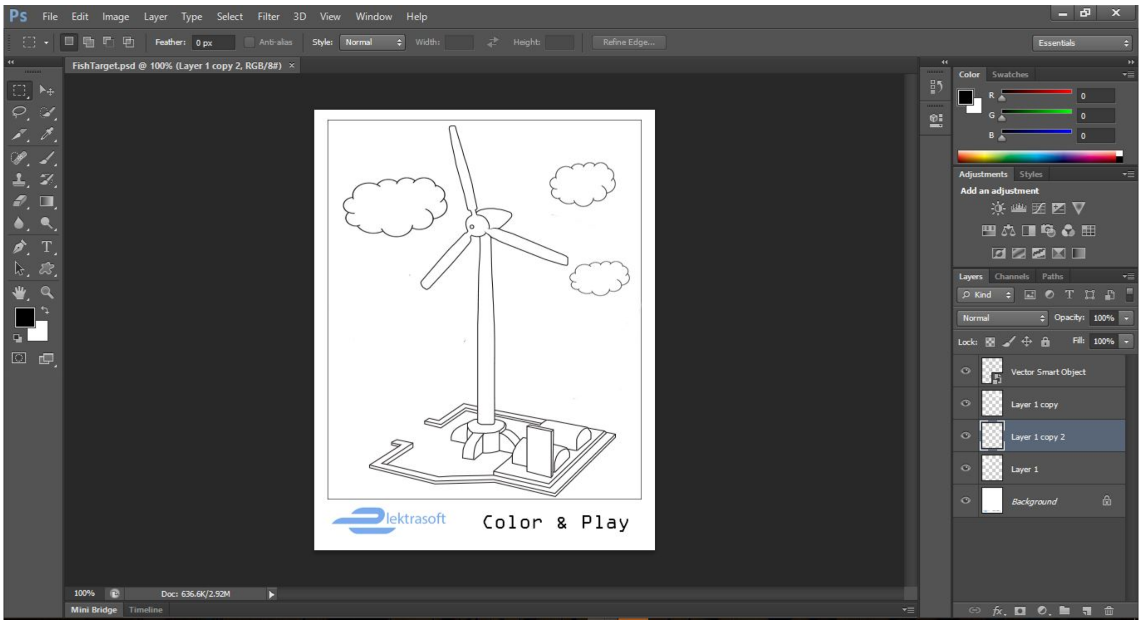
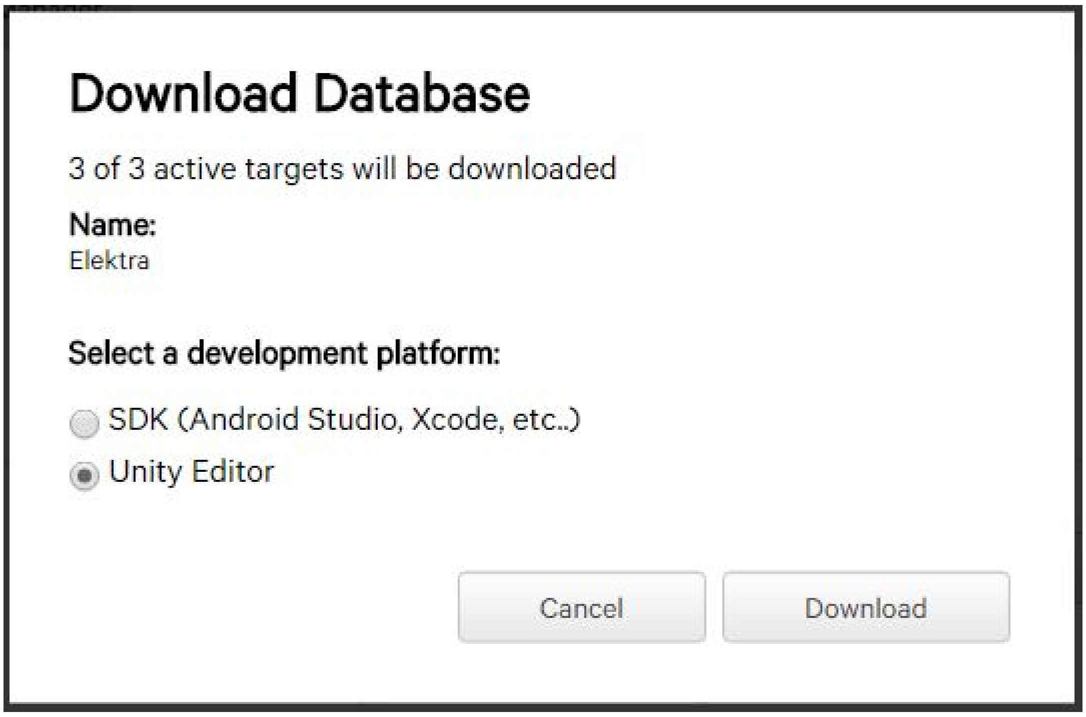


Figure 5.4: Making Image Targets using Adobe Photoshop

These artifacts are saved with the *.JPEG* extensions as an image. They are then uploaded on the Vuforia's Developer Web Portal where they provide the availability to make this Image an Image Target by making this the Unity 3D Package as shown in Figure 5.5.


Download Database

3 of 3 active targets will be downloaded

Name:
Elektra

Select a development platform:

SDK (Android Studio, Xcode, etc..)
[✓ Unity Editor]

Figure 5.5: Image Targets as Unity Package Download Availabilty

After downloading this Unity 3D package, it can be imported into the Unity3D's environment by simply drag and drop method to the Project's asset folder or by going to Assets -> Import Package -> Custom Package as shown in Figure 5.6.

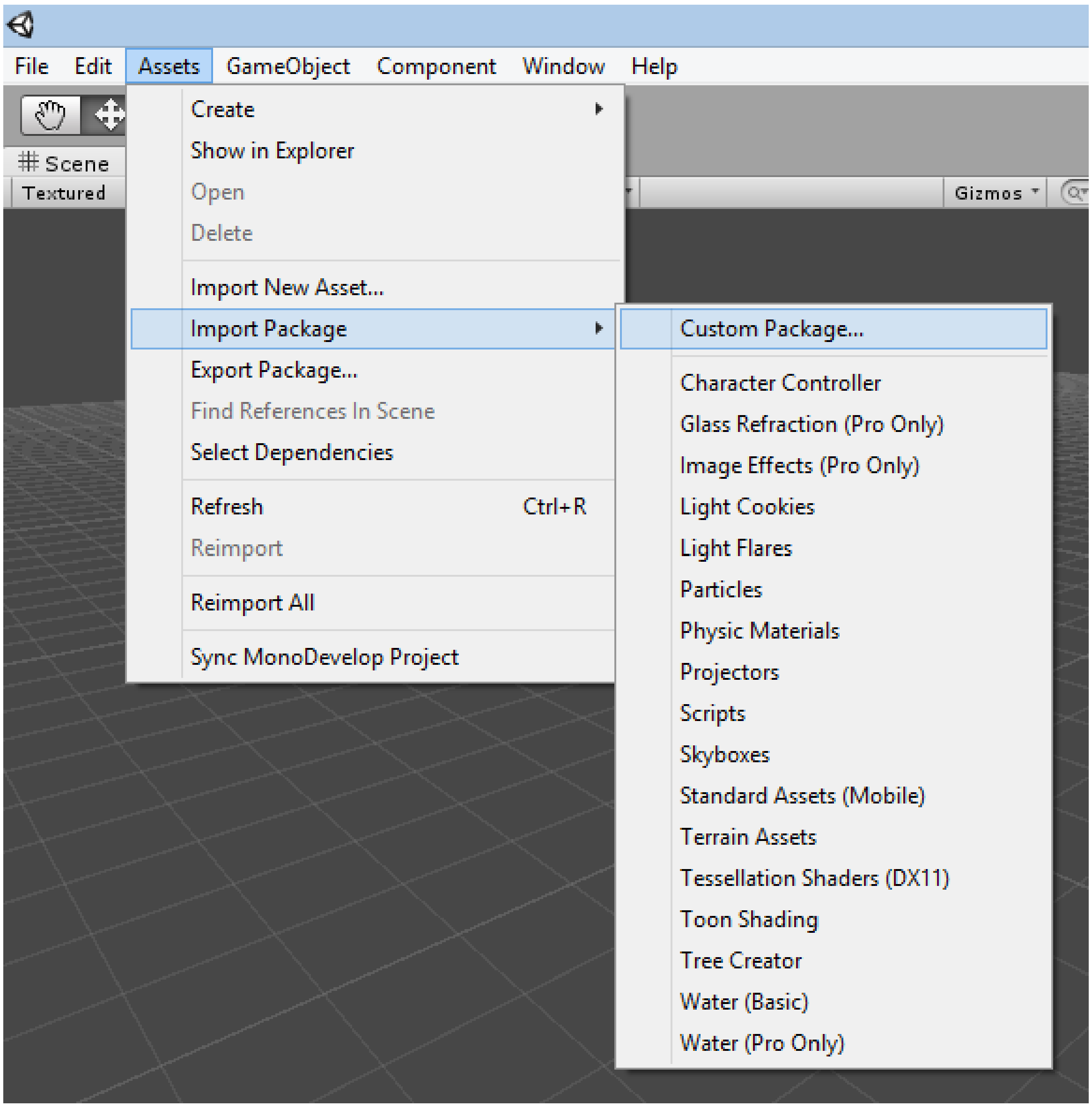


Figure 5.6: Importing Custom Package in Unity

5.4.2 Autodesk Maya 3D Models Integration

Autodesk Maya is a professional 3D modeling tool in which 3D models can be created, animations can be made with its with different Animation Tools and it also provides functionality to map the 3D model meshes to UV coordinates to correctly map the 2D texture to the 3D model.
Once the model is ready with its animations and UV-mapping, Autodesk Maya provides

the availability to export the model in different ways for different platforms. Some of them are listed as follows.

- **Export All:** All the scene components can be exported by selection of this option and it can be exported in different formats i.e MayaBinary, FBX export, mayaASCII and many more.
- **Export Selection:** Selected components from the scene can be selectively exported in different formats as stated earlier.
- **Send to Unity:** Autodesk Maya models/scenes can directly be exported to Unity Game Engine. By selecting this option, we can then browse to our Unity Projects location and either 'export all' for exporting the whole scene or 'export selection' for selective component exports.
  It is also not necessary to set the project again for subsequent exports.

Unity, on the other hand, provides availability to import the MayaBinary format file, FBX can also be imported but the 'Send to Unity' options seems best due to the elimination of handling subsequent exports.

5.5 Processing Logic

Once all the different modules are integrated into the Unity3D IDE. They need to be processed in a proper manner.

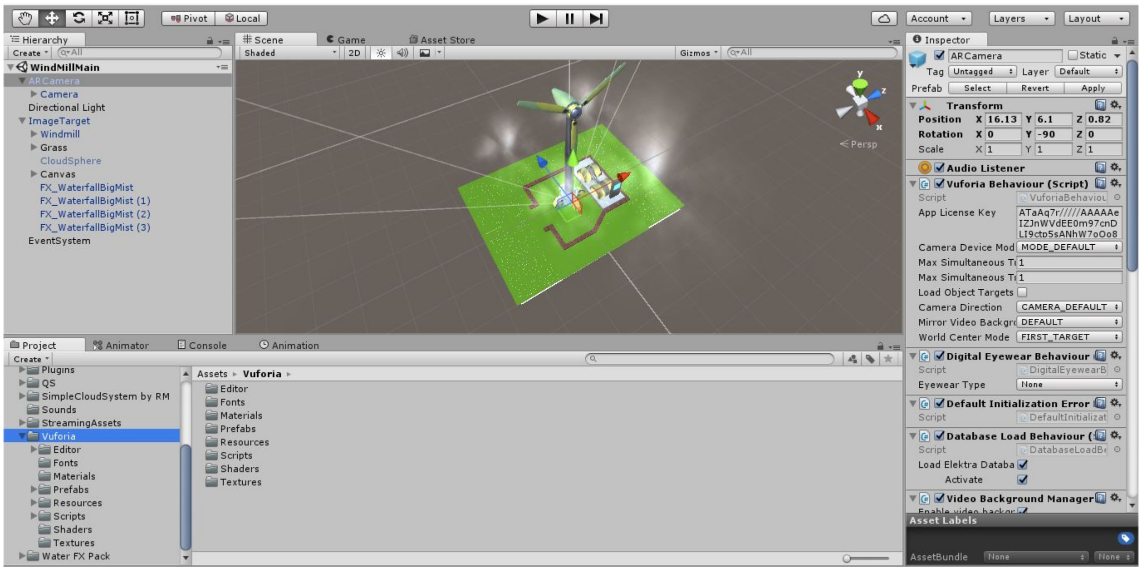
Before going to use them, Vuforia SDK needs to be imported as well. Vuforia provides the Unity Package which includes all the AR development tools we need to use to provide our AR related requirements. Once the Vuforia SDK is imported into Unity project, the folder named 'Vuforia' can be seen in the Assets of the project, which includes all the development tools like Scripts, Prefabs, Materials, Resources etc.

For our requirements, we need to remove the built-in camera component provided by Unity3D and drag-n-drop ARCamera from the Prefabs of Vuforia to the project's hierarchy. Next, drag-n-drop ImageTarget component from the Prefabs below to the ARCamera in the hierarchy.

Once they both are in place, we need to set their some attributes first. By clicking the ARCamera, the properties of the ARCamera would be shown in the Inspector. Here, we need to set the App License Key which is provided by Vuforia itself through its online developer portal. Now the database needs to be loaded by 'Checking' the box next to 'Load Database' and also the check box of 'Activate', these options are under 'Database

Load Behavior' in the Inspector. Note that, this option would only be here after you have imported the Image Target Unity Package discussed in the Section 5.4.1. All other properties are fine to their default values.

Properties of ImageTarget component in hierarchy also needs to be set, its properties can be viewed in Inspector by simply clicking on the component. Database attribute can be set through drop down menu and desired Image Target's database can be loaded. It would only be there if Image Target's package is imported already. Next we need to make our corresponding 3D model the child of ImageTarget component in the hierarchy. It should be noted that we need to first import our 3D model as discussed earlier in Section 5.4.2. By doing this, we are defining that we need to augment that specific model when the Image Target is detected. The positioning of the 3D model can be done using Unity's scene. Sample Project's view is shown in Figure 5.7.



Inspector
ARCamera Static
Tag Untagged Layer Default
Prefab Select Revert Apply
Transform
Position X 16.13 Y 6.1 Z 0.82
Rotation X 0 Y -90 Z 0
Scale X 1 Y 1 Z 1
Audio Listener
Vuforia Behaviour (Script)
Script VuforiaBehaviou
App License Key ATaAq7r/////AAAAAe IZJnWVdEE0m97cnD LI9ctp5sANhW7oOo8
Camera Device Mod MODE_DEFAULT
Max Simultaneous Tı 1
Max Simultaneous Tı 1
Load Object Targets ☐
Camera Direction CAMERA_DEFAULT
Mirror Video Backgrı DEFAULT
World Center Mode FIRST_TARGET
Digital Eyewear Behaviour
Script DigitalEyewearB
Eyewear Type None
Default Initialization Error
Script DefaultInitializat
Database Load Behaviour (
Script DatabaseLoadB
Load Elektra Databa ✓
Activate ✓
Video Background Manager
Asset Labels
AssetBundle None None

Figure 5.7: Sample Unity Scene after Integration

Till here, the augmentation part is complete but main part of the project starts from here, that is Live Image Processing. In Live Image Processing, we need to process the camera component to fetch the frames of the video. When we have the frames of the camera video, we have to apply certain techniques to remove the environment from the frames.

First of all, we need to make a C# script in which we shall register our camera and set the type of frame we want to fetch because there are different types of frames we can fetch i.e. GrayScale, RGB888, RGBA8888, RGB565. We would select RGBA8888 due to its highest quality of the camera frame. As the Image Target can not always be in the camera frame therefore we have to specify that whenever the Image Target is detected we need to get the frame. Therefore, for getting the camera frames only when the Image

Target is on the camera video/frames, we need to initialize the instance of class named as ’TrackableBehavior’. TrackableBehavior is inherited from ITrackableEventHandler, by inheriting this super class we can override the function ’OnTrackableStateChanged’ which is called whenever the state is changed i.e. whether the Image Target is Detected, Tracked or Extended_Tracked, and it brings the attributes ’previousState’ and ’newState’ with it as the parameters through which we can manipulate the previous and current state.

While we can manipulate the current state of the Image Target whether it is detected or not, we can now fetch the camera frames whenever the newState is ’Detected’ or ’Tracked’. Note that this fetched camera frame is of the type ’Image’, we need to convert it to texture for further processing. To make the texture of this Image we need to initialize the Texture2D object of the same size as the Image. Now this texture can be applied to our 3D model which is already UV-mapped. But it should be noted that our 3D models are UV-mapped according to the Image Targets therefore the texture that contains the Image Target also contains the other non-related environment as well that cannot be directly applied to the 3D model. For removing the non-related environment from the texture we need to further process our texture2D object.

For locating the Image Target and localizing our texture to Image Target (i.e. removing non-related environment) we need two classes named as ‘ImageTarget’ that is the Game Object and ‘ImageTargetBehaviour’ which records the information like size of Image Target in Unity Scene etc. In the live frames of the camera, we can locate the Image Target with our ImageTarget’s property called ‘transform’, it contains the information about Image Targets like position (i.e. position in world space), local rotation (i.e. rotation relative to parent’s transform rotation that is camera in our case) and many more. By using this position information and local rotation we can locate our Image target in the frames irrespective of its rotation. Next major issue is the scale of the Image Target in the camera frames because it depends how far or away the camera is from the Image Target. This can be handled using the class ‘ImageTargetBehaviour’. We can call the method ‘GetSize()’ of the mentioned class to get the scale of ImageTarget and we can receive it as the Vector3 form. After transforming the frames to our new coordinates according to the received Vector, we can reach our coordinates by converting texture2D to matrix and retrieving the sub-matrix accordingly. We can convert the matrix back to texture2D and this converted texture would be the customized colored Image Target which can be applied to the 3D model.

5.6 Development Strategies

5.6.1 Software Development Life Cycle

Due to the nature of project, Waterfall life-cycle model cannot be employed. The model which gives the relaxation of changing the current phase and shifting to another phase for improvements or new requirements is desired. Star Life-cycle Model is the alternate of the Waterfall model which software engineers have designed to accomplish the projects of nature where we need to start different phases independently as shown in Figure 5.8.

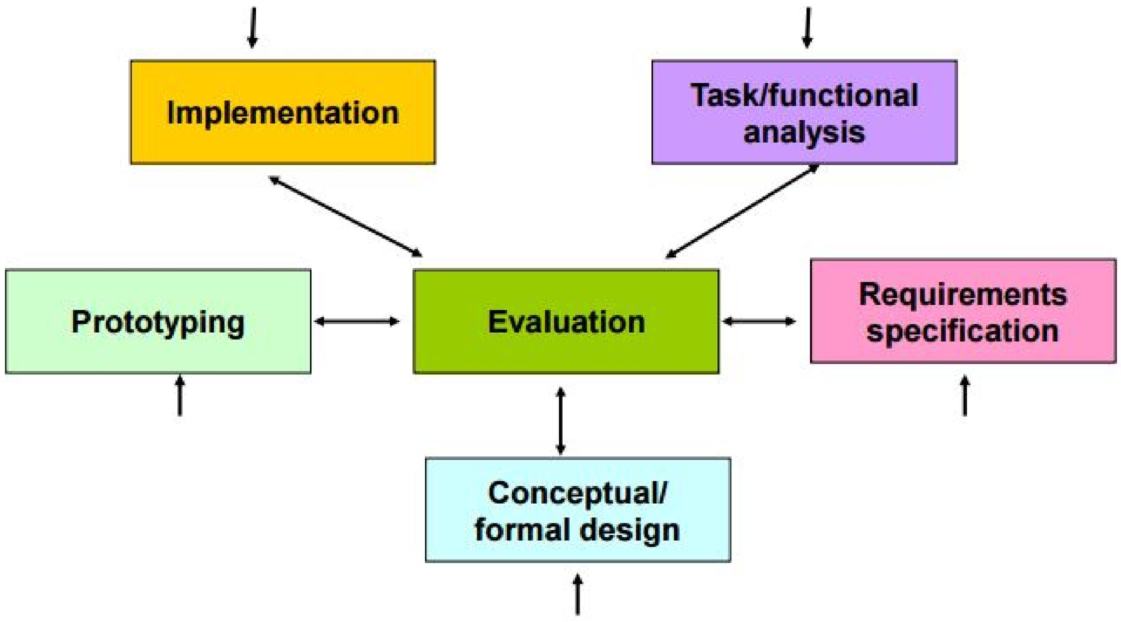


Figure 5.8: Star Life-cycle Model

Most important feature of Star Life-cycle model are:

- It is derived from empirical studies of the interface designer.
- Evaluation is centralized.
- It emphasized on fast prototyping.
- Design phase can be started from any phase.
- Two modes of activities:
    1. Analytic (Top down, Organizing, Formal working from system's view to user's view).
    2. Synthetic(Bottom up, Thinking, Creative, Adhoc working from user's view to system's view).

The development nature of the project is same as the design provided by the Star Model. As we do require to develop the application in increments and implementing a random phase and of course the evaluation is compulsory at the end of every phase, as is provided in SLCM (Star Life Cycle Model). For example, we develop image targets in Illustrator and models in Autodesk for every separate model. We integrate them in the Unity for finalizing the augmentation part of this model and then repeat the process for another model, this shows the nature of incremental development. On the other hand, we may need to redefine 3D model's properties or may have to redesign the image targets therefore we can jump from one phase to another according to the Star Life Cycle Model.

5.6.2 Implementation Strategies with Unity

It requires a separate Unity project for developing a single model with its augmentation and live processing. Although we cannot make separate applications for every single model we develop. However, we would make separate projects for every single model we develop but they can be integrated into one application/project by using the Scenes of Unity and this issue can also be resolved by using multiple image targets in the same project.

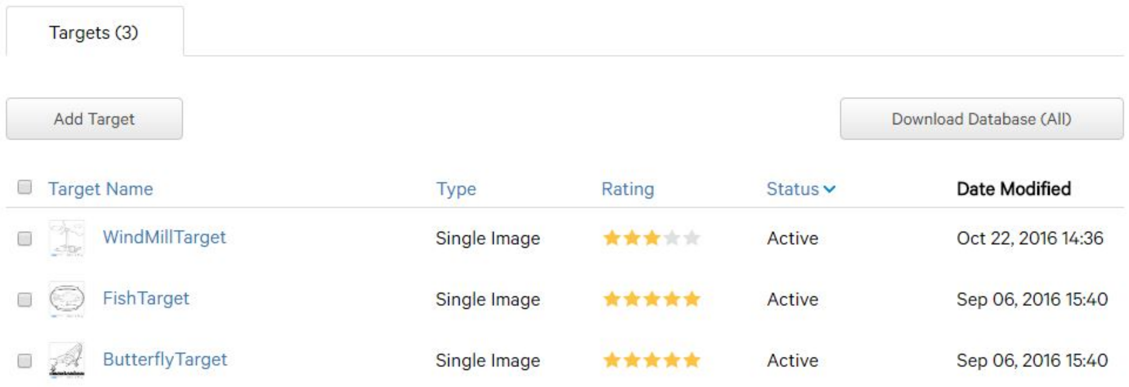
5.6.2.1 Implementation with Unity Scenes

The project made with Unity can be converted into a Scene with all the components and functionalities. This Unity Scene is a single file which can be loaded into any Unity project and it can be integrated with that project. Therefore, every model can be created as a separate project i.e. its Image Target is created and integrated as discussed in the Section 5.4.1, its 3D model with animations and UV-mapping are integrated as discussed in the Section 5.4.2 and all the libraries i.e. Vuforia SDK, OpenCV can be integrated as custom package as shown in Figure 5.6. Live processing is separate in a way for every image target, as we define separately that this specific Image Target needs to be detected and that is unique for every model. When everything is set for a single model, this project is saved as Unity Scene and when all the scenes are made separately, they are loaded into one project.

The major issue with this approach is that we would need to explicitly tell the application whether which image target we want to detect, as every Image Target is in its separate scene and only one scene can be viewed at a single time.

5.6.2.2 Single Scene with Multiple Image Targets

Another approach is to make only one scene for example Augmentation Scene, in which we would create multiple image target objects and define them separately as discussed earlier in the Section 5.5. By having multiple targets in the same hierarchy what happens is that we do not need to define that this specific target needs to be detected. Note that for this purpose, we also need to make the ImageTarget Unity Package accordingly. Now our ImageTarget Package would not only contain a single target, but it be a package of multiple targets as shown in the Figure 5.9.


Targets (3)

Add Target

Download Database (All)

Target Name	Type	Rating	Status	Date Modified
WindMillTarget	Single Image	★★★☆☆	Active	Oct 22, 2016 14:36
FishTarget	Single Image	★★★★★	Active	Sep 06, 2016 15:40
ButterflyTarget	Single Image	★★★★★	Active	Sep 06, 2016 15:40

Figure 5.9: Multiple Targets under same Package

This downloaded Unity Package would contain all our Image Targets and would be set in our Project's hierarchy in our ImageTarget objects as discussed in the Section 5.5. It should be noted that we would still have as many ImageTarget objects in our hierarchy as many image targets are there, because they are unique with different 3D models as shown in Figure 5.10 for two different models in the same project.

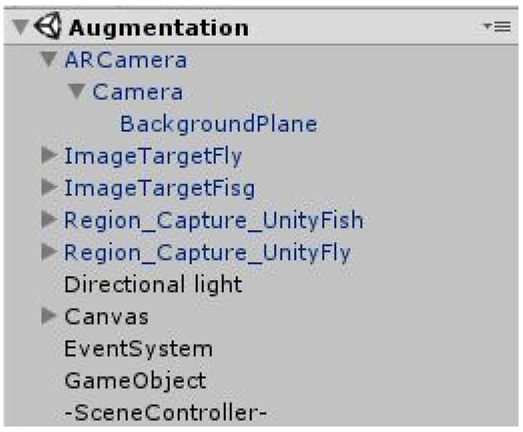
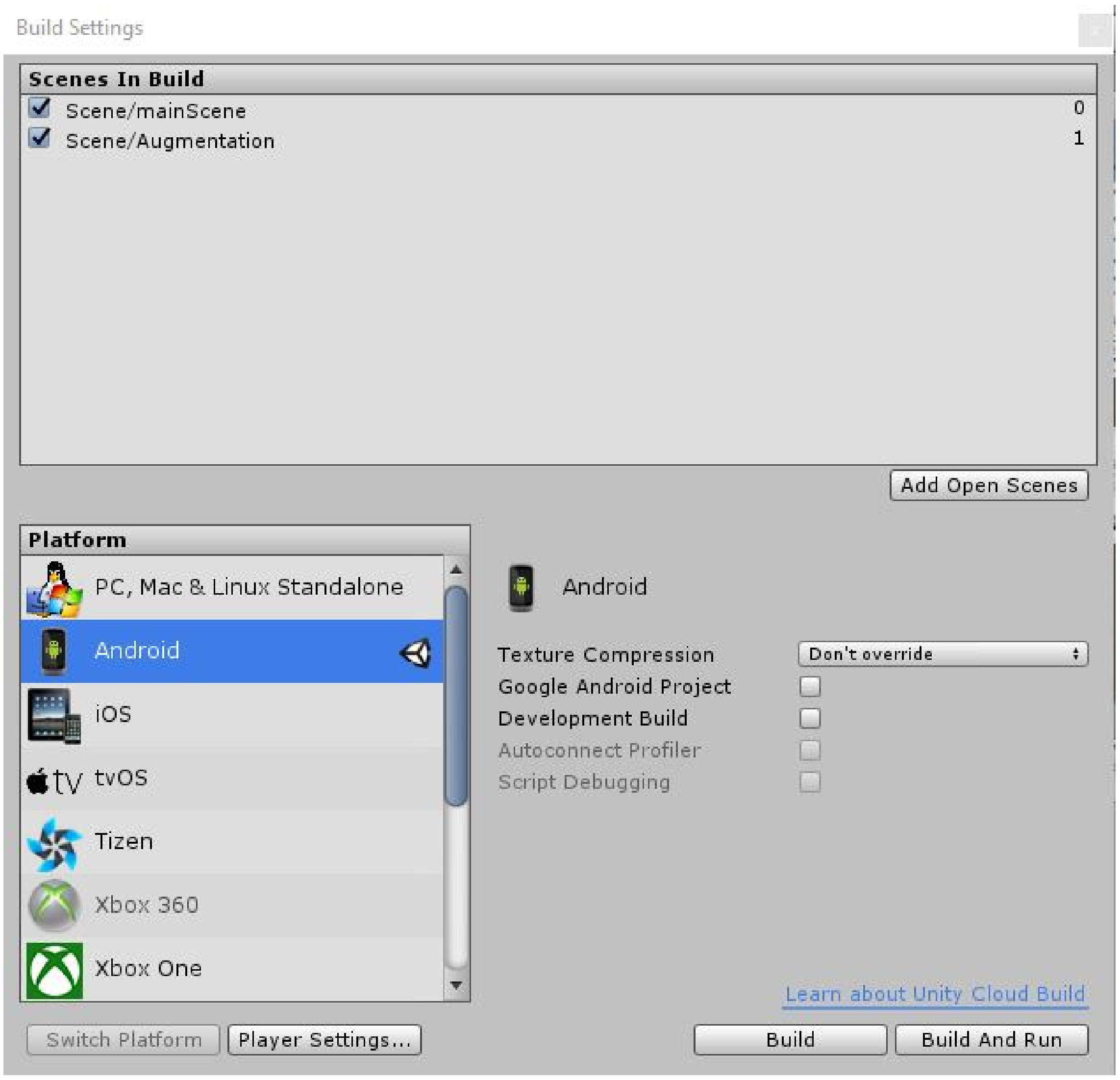

```
▼ Augmentation
  ▼ ARCamera
    ▼ Camera
        BackgroundPlane
  ▶ ImageTargetFly
  ▶ ImageTargetFisg
  ▶ Region_Capture_UnityFish
  ▶ Region_Capture_UnityFly
    Directional light
  ▶ Canvas
    EventSystem
    GameObject
    -SceneController-
```

Figure 5.10: Multiple Targets Hierarchy in same Project

5.7 Conversion Strategy

Unity 3D is a multi-platform supported IDE as discussed in the section 5.2. Unity requires the specific Software Development Kit of specific platform on your computer to provide you the environment to develop applications on it and build for that specific platform e.g.

Android SDK as discussed in the Section 5.2.3. After installing the required SDK, we need to switch to that platform in the Build Settings of the Unity. It can be done by going to File -> Build Settings, or by a shortcut key 'Ctrl+Shift+B'. After selecting your platform (Android in our case), we need to click switch platform just below the list view of different platforms as shown in the Figure 5.11, note that the switch platform button would be disabled if you are already on that platform.



Build Settings

Scenes In Build

- [✓] Scene/mainScene 0
- [✓] Scene/Augmentation 1

Add Open Scenes

Platform

- PC, Mac & Linux Standalone
- Android
- iOS
- tvOS
- Tizen
- Xbox 360
- Xbox One

Android

Texture Compression	Don't override
Google Android Project	☐
Development Build	☐
Autoconnect Profiler	☐
Script Debugging	☐

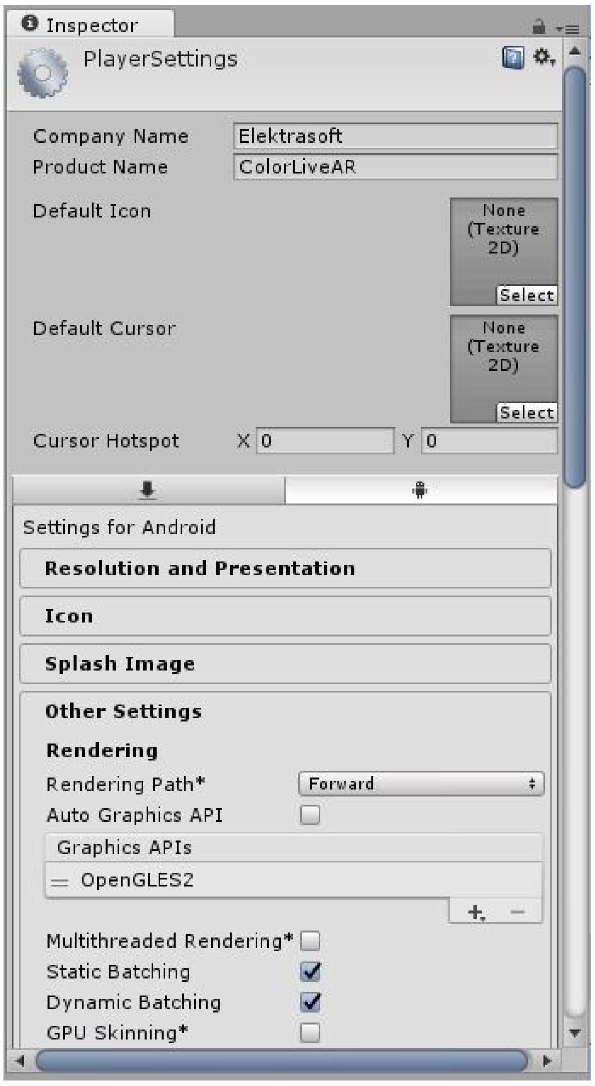
Learn about Unity Cloud Build

Switch Platform | Player Settings... | Build | Build And Run

Figure 5.11: Build Settings - Switch Platform

We can start developing once the platform is selected, it is important to switch platform before starting development because some API's do not work for every platform, i.e. if we are developing the application and running in the Web Player platform it might run smoothly but if we change the platform in the mid of the development, there might be issues with some API's running for Web Player. In that case, all that issues occurring API's need to be re-configured.

Once the project is of the export level i.e. to run on the real mobile device, we have to build the *.apk* and transfer that on the mobile and install it. *.apk* 'Android Package Kit' is the installer file which runs on android devices. Before building for android device, we need to set the 'Player Settings' which can be opened from build setting as shown in Figure 5.11, and its attribute can be set in Player Settings window as shown in the Figure 5.12,



Inspector

PlayerSettings

Company Name: Elektrasoft
Product Name: ColorLiveAR
Default Icon: None (Texture 2D) Select
Default Cursor: None (Texture 2D) Select
Cursor Hotspot: X 0 Y 0

Settings for Android

Resolution and Presentation

Icon

Splash Image

Other Settings

Rendering

Rendering Path*: Forward
Auto Graphics API: ☐
Graphics APIs
OpenGLES2

Multithreaded Rendering*: ☐
Static Batching: ✓
Dynamic Batching: ✓
GPU Skinning*: ☐

Figure 5.12: Player Settings of Unity

'Player Settings' defines the Company Name, Product Name, Resolution Presentation, Icon for the App, Splash Image, Publishing Settings and Other Settings which includes Rendering Path, Graphics API, Identification (Bundle identifier) etc.

Chapter 6

System Testing and Evaluation

Software testing has been carried out from every aspect for the application and it has been discussed thoroughly in this chapter.

6.1 Graphical user interface testing

Graphical user interface testing of Coloring Book with Augmented Reality has been carried out by different critiques for the better understanding of the way the user thinks about the interactivity with the current GUI. The Figure 6.1 was the reference interface of the main menu.

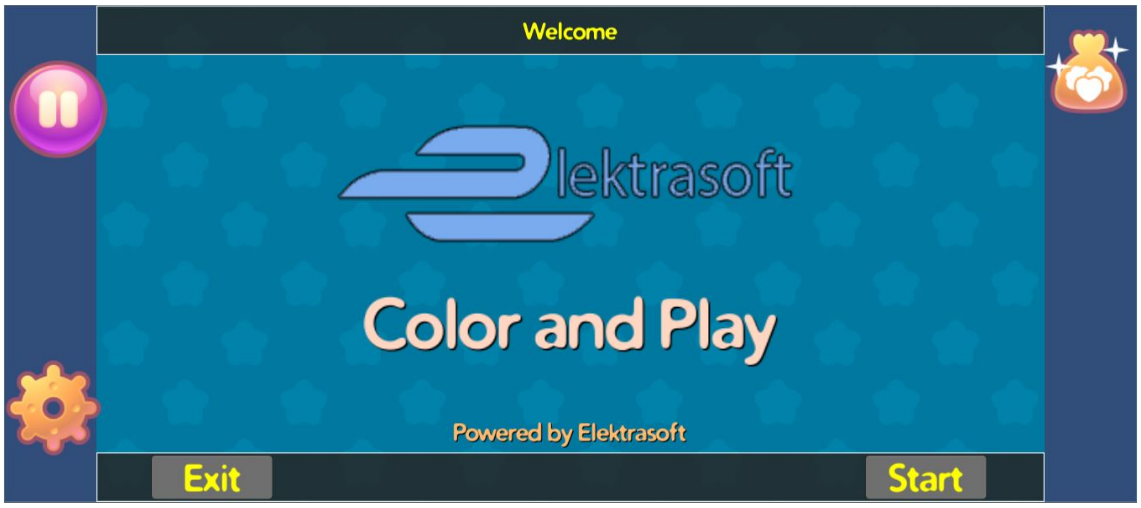


Figure 6.1: Reference Main Menu

Although users response was positive about the interactivity and about the animations provided in the main menu but there were also few feedbacks about some changes related to the display of the name and better background color. These feedbacks were duly noted and after the consideration of new design of display name and background color, it was revised.

As the augmentation scene also includes the interface to go back to the main menu, it was also asked the users to evaluate the availability of the interface to go back and also evaluate the design. Figure 6.2 shows the reference of the interface being evaluated.

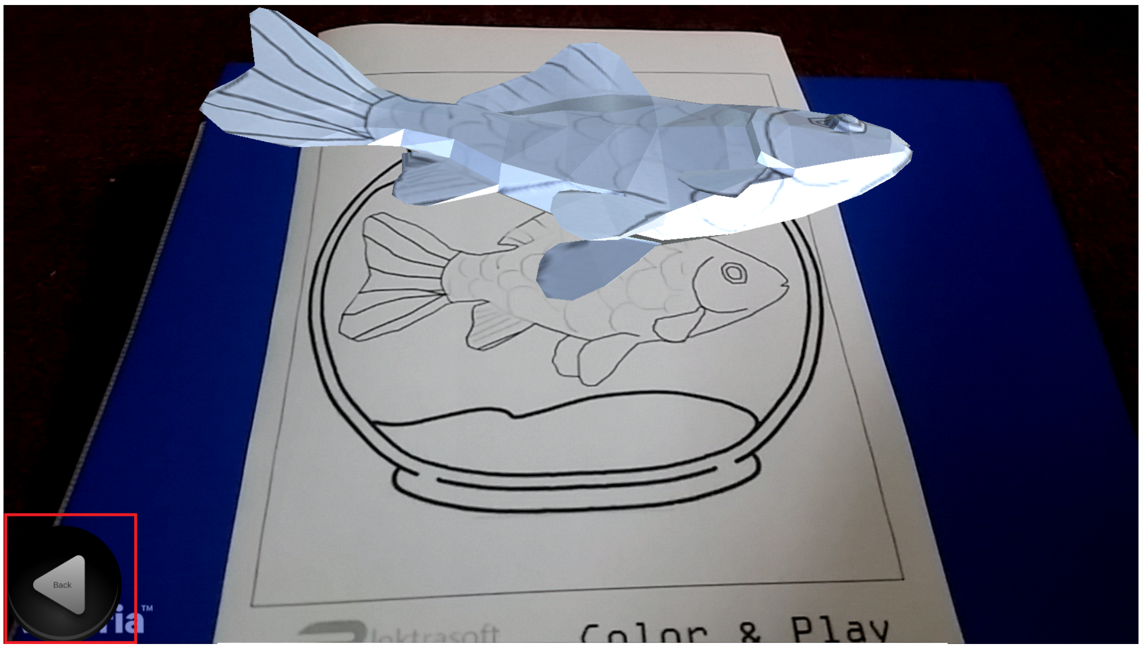


Figure 6.2: Referenced Interface - Red Square

It was deduced after the evaluation that the availability of this interface is required but it was needed to be redesigned for the better look of the environment. After consideration of feedbacks, new GUI design has been employed with the help of Adobe Photoshop and Adobe Illustrator. The new splash screen of the application is also re-designed and can be seen in the Figure 6.3

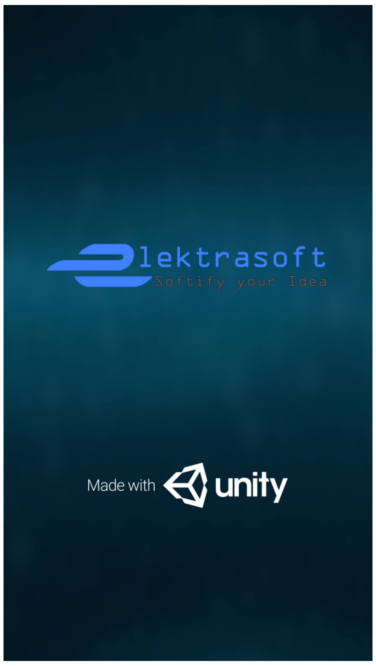


Figure 6.3: Application Splash Screen

New Main Screen with different menus has also been re-designed after the evaluation of previous one, and it can be seen in the Figure 6.4

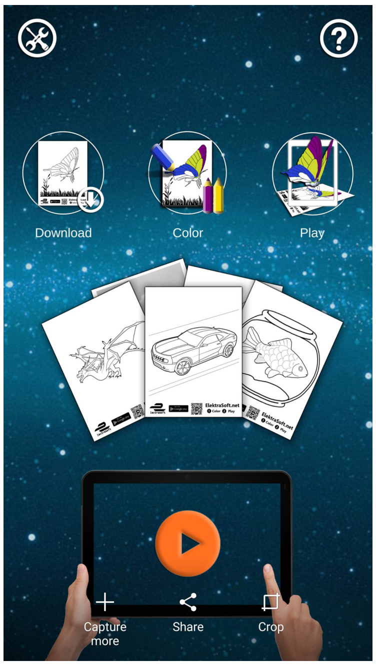


Figure 6.4: Application Main User Interface

Different hint with the dialog boxes are also provided, i.e. how to Download, Print and Color. View of these dialog boxes can be seen in the Figure 6.5

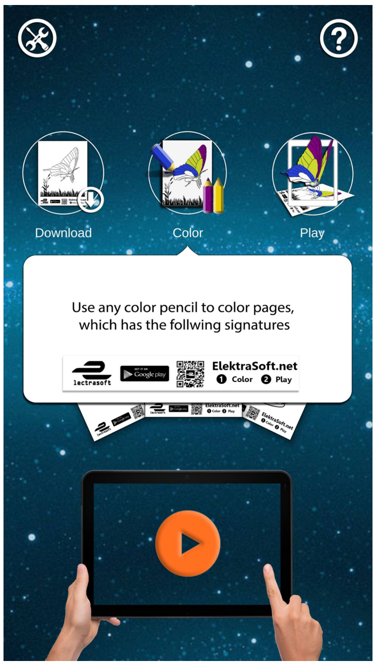

Download
Color
Play
Use any color pencil to color pages, which has the follwing signatures
ElektraSoft.net
Color
Play

Figure 6.5: Hint Dialog Box

Furthermore, help UI is also provided which explains how to use the application and this help has been shown with an attractive slider which shows the procedure step-by-step, sample view can be seen in the Figure 6.6

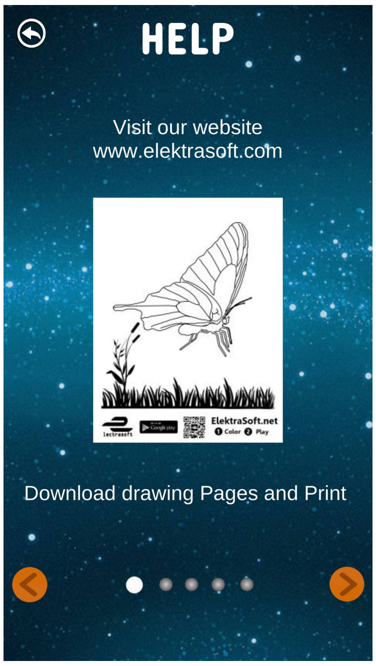

HELP
Visit our website www.elektrasoft.com
ElektraSoft.net
Download drawing Pages and Print

Figure 6.6: Help with Slider

For giving the power to user to change the feel of the application, multiple backgrounds are provided and users can select whichever they like by going to settings, sample view can be seen in the Figure 6.7

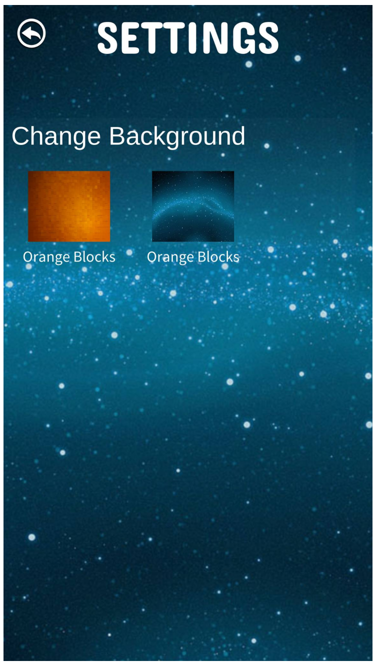


Figure 6.7: Changing Backgroud UI

Finally, after the feedbacks to change the previously designed of UI for going back to the main menu from augmentation scene, new UI has been redesigned and can be seen in the Figure 6.8

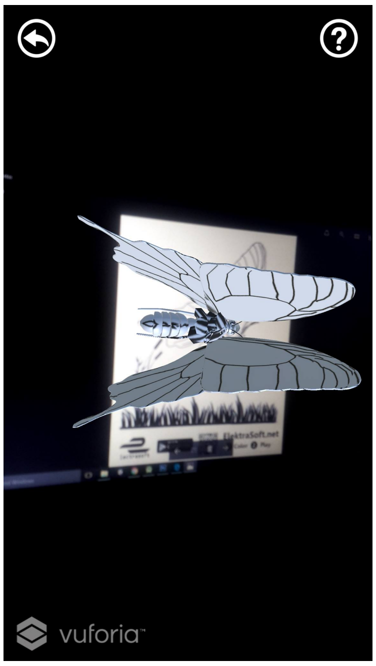


Figure 6.8: Augmentation Scene with New UI

To access help from the augmentation scene, help UI is also provided in the new design and all revised interface has been tested and it was concluded that the new interface is a lot better than the previous one.

6.2 Usability Testing

Usability testing has been carried out by testing it with the real users due to the nature of testing components in it. Usability testing components covered are Learnability, Efficiency, Memorability, Error, Satisfaction. Each of them is discussed in the sections below.

6.2.1 Learnability

Learnability testing is related to how much users struggle with using the application and how easy is it for them to perform basic tasks. As stated in the Section 3.3.3.1, we have tried to provide as less control as we can to the users and also maximizing the functionality without their much interaction only because our target users are children. After the evaluation by the users, it was concluded that interacting with the application, starting the augmentation and playing with the animations was very trivial.

6.2.2 Efficiency

This component of usability testing analyzes the user's ability to perform tasks quickly once they have learned the flow of the application. After testing efficiency by the users, it was concluded that after using the application for the first time the users were not stumbling in the opening and playing with the animations once they had already used the application before.

6.2.3 Memorability

Memorability testing is related to the situation where users are asked to test the application after familiarizing the application for the first time and using the application again after a considerable interval in between to see how well they recognize the steps to open and perform tasks. After the evaluation and due to the simple and natural interface, it was concluded that playing with the animations and overall application functionalities was trivial for the users.

6.2.4 Error

Errors here are the code generated errors, these are the errors/mistakes users make while using the application. After the evaluation of the mistakes, it was concluded that while in the augmentation scene, users were sometimes not being able to realize that the image targets need to be in the camera frames, and being extended tracked for the visualization of the augmentation. They were usually assuming that once image target is detected, it may remain on the screen even when the image target is not there. This was only due to the age they are off, children usually get used to the concepts a little later with experience.

6.2.5 Satisfaction

This section is the main part of the usability testing, where we get to know that users will come back to use the application or not and after the evaluation process it was perceived that users were really excited about their imaginative colorings to the characters coming to life with their natural movements and they also showed more creativity about coloring the 2D characters.

6.3 Software Performance Testing

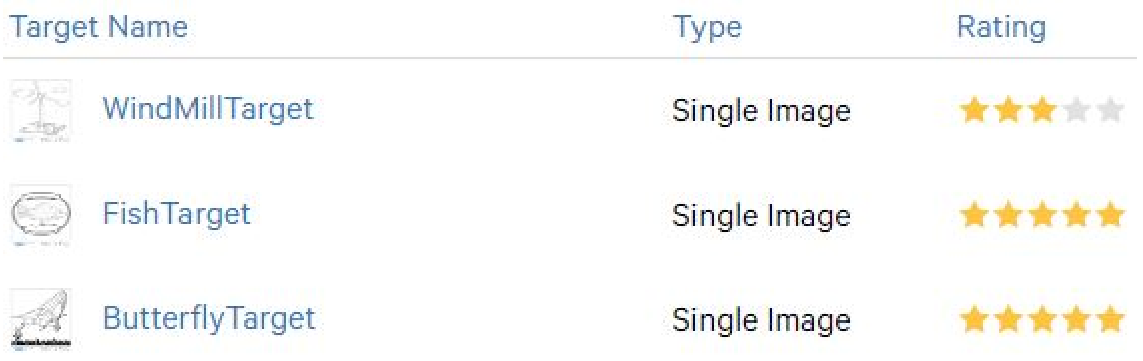
Software Performance Testing as a whole was performed by different users but before that, it was important to test different units/modules first separately and after the integration. Therefore Unit Testing and Integration Testing was also performed which are discussed in the sections below.

6.3.1 Unit Testing

As discussed earlier in the Chapter 5, the application is divided into different modules and tested separately.

6.3.1.1 Image Targets Testing

Image Targets are the artistic drawing made with Adobe Photoshop. As they are not related to the coding part, therefore, they were needed to be tested as a black box. It just had to be looking attractive and able to be colored on. Furthermore, these artistic images are rated separately whether how good they are for the detecting in the live camera frames stream. Sample rating of the Image Targets are shown in Figure 6.9, more the number of stars, more detectable the target image is.



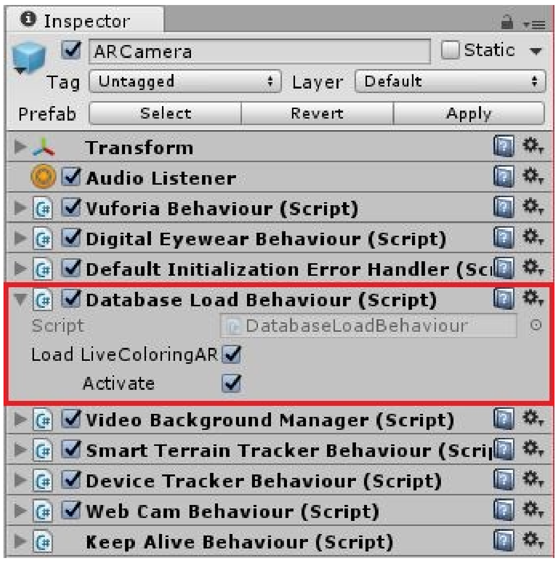
Target Name	Type	Rating
WindMillTarget	Single Image	★★★☆☆
FishTarget	Single Image	★★★★★
ButterflyTarget	Single Image	★★★★★

Figure 6.9: Image Targets Rating

6.3.1.2 Image Targets Unity Package Testing

Image Targets are download as the Unity Package as discussed in the Section 5.4.1, once they are downloaded, they are required to be tested in Unity environment after importing them. Two methods are there to check if Image Targets are imported correctly.

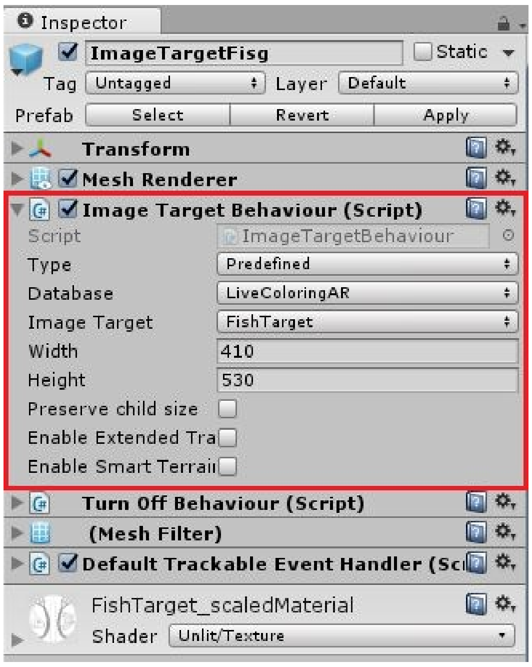
- In the ARCamera component, its attribute named as ‘Database Load Behavior’ gives the checkbox to load the dataset as shown in Figure 6.10.



```
Inspector
✓ ARCamera                                   ☐ Static
Tag  Untagged          Layer  Default
Prefab   Select   Revert   Apply
▶ Transform
  ✓ Audio Listener
▶ ✓ Vuforia Behaviour (Script)
▶ ✓ Digital Eyewear Behaviour (Script)
▶ ✓ Default Initialization Error Handler (Sc
▼ ✓ Database Load Behaviour (Script)
   Script              DatabaseLoadBehaviour
   Load LiveColoringAR ✓
          Activate     ✓
▶ ✓ Video Background Manager (Script)
▶ ✓ Smart Terrain Tracker Behaviour (Scri
▶ ✓ Device Tracker Behaviour (Script)
▶ ✓ Web Cam Behaviour (Script)
▶   Keep Alive Behaviour (Script)
```

Figure 6.10: ARCamera - Defining Database

- In the ImageTarget component, under the ‘Image Target Behavior’ script, there comes the options to explicitly define the database and the image target because there can be multiple image targets as shown in Figure 6.11.



Inspector

ImageTargetFisg ☐ Static

Tag: Untagged — Layer: Default

Prefab: Select | Revert | Apply

Transform

☑ Mesh Renderer

☑ Image Target Behaviour (Script)

Script	ImageTargetBehaviour
Type	Predefined
Database	LiveColoringAR
Image Target	FishTarget
Width	410
Height	530
Preserve child size	☐
Enable Extended Tra	☐
Enable Smart Terrai	☐

Turn Off Behaviour (Script)

(Mesh Filter)

☑ Default Trackable Event Handler (Sc

FishTarget_scaledMaterial

Shader: Unlit/Texture

Figure 6.11: Image Target - Defining Image Targets

6.3.1.3 Autodesk Maya 3D Models

3D models made with Autodesk Maya are also tested separately in the Autodesk Maya's environment. There are two major components of every model that needs to be tested and these are discussed below.

- **UV-mapping of the 3D model with corresponding Image Target.**

  UV-mapping is the concept of defining the x and y coordinates of the 2D image to the 3D coordinates. For doing this, we split the meshes of the 3D model in a 2D environment put that to their exact location in the 2D image. For example in Figure 6.12, butterfly's 3D model is opened in a UV-editor where the model's meshes are displayed as 2D faces. We can move these faces to where we want to map it with, as shown in the figure where we are mapping the right-wing of the butterfly to the image where the right-wing is drawn, by doing this, when the user colors that wing, those colors are picked from that area of the image and mapped to the 3D right-wing of the model.

  Here, we need to define each UV-coordinated accurately otherwise, the colors won't be mapped accordingly.

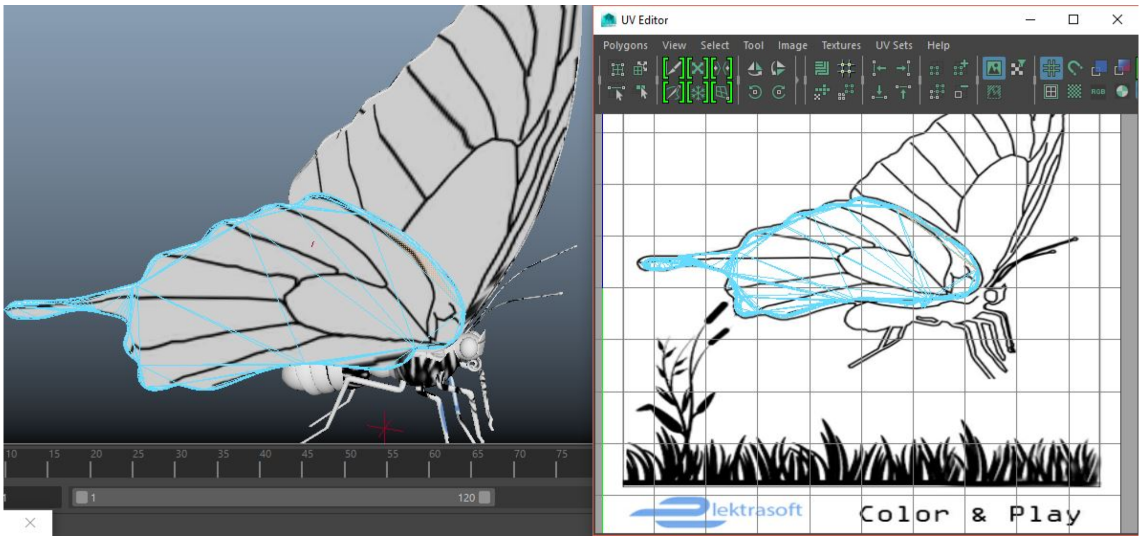


Figure 6.12: UV-Mapping of the right-wing

- **Animation of the 3D model.**

Animations of the 3D model are also tested before exporting them to Unity. As we define the animations according to a specific timeline, therefore, we can also test whether each defined timeline is played accordingly. Timeline is shown in Figure 6.13

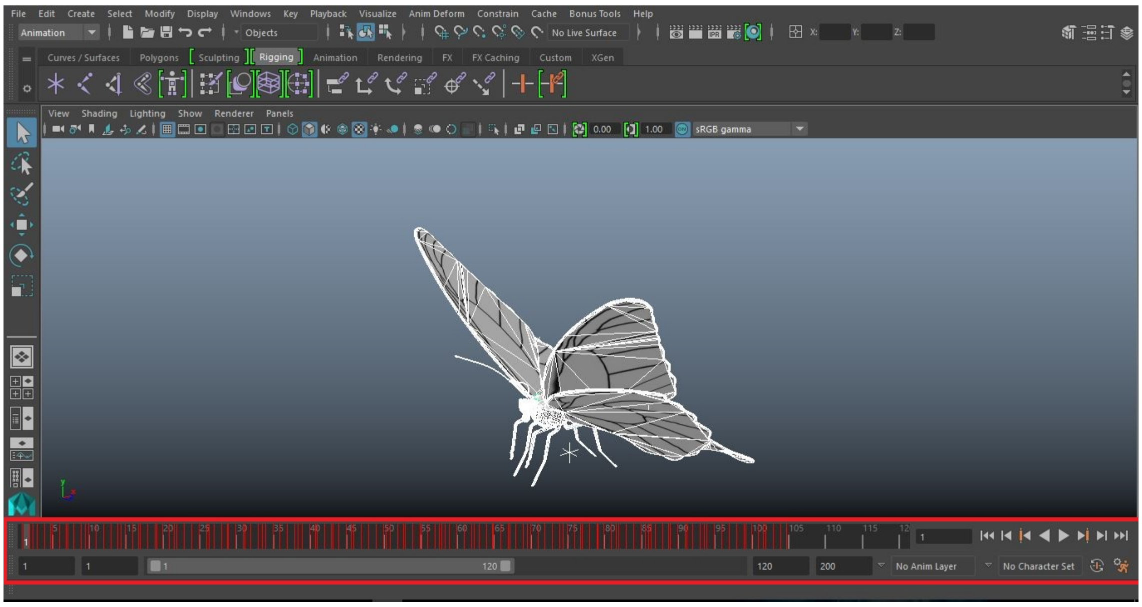

File Edit Create Select Modify Display Windows Key Playback Visualize Anim Deform Constrain Cache Bonus Tools Help
Animation
Objects
No Live Surface
Curves / Surfaces Polygons Sculpting Rigging Animation Rendering FX FX Caching Custom XGen
View Shading Lighting Show Renderer Panels
sRGB gamma
No Anim Layer
No Character Set

Figure 6.13: Animation Time-line

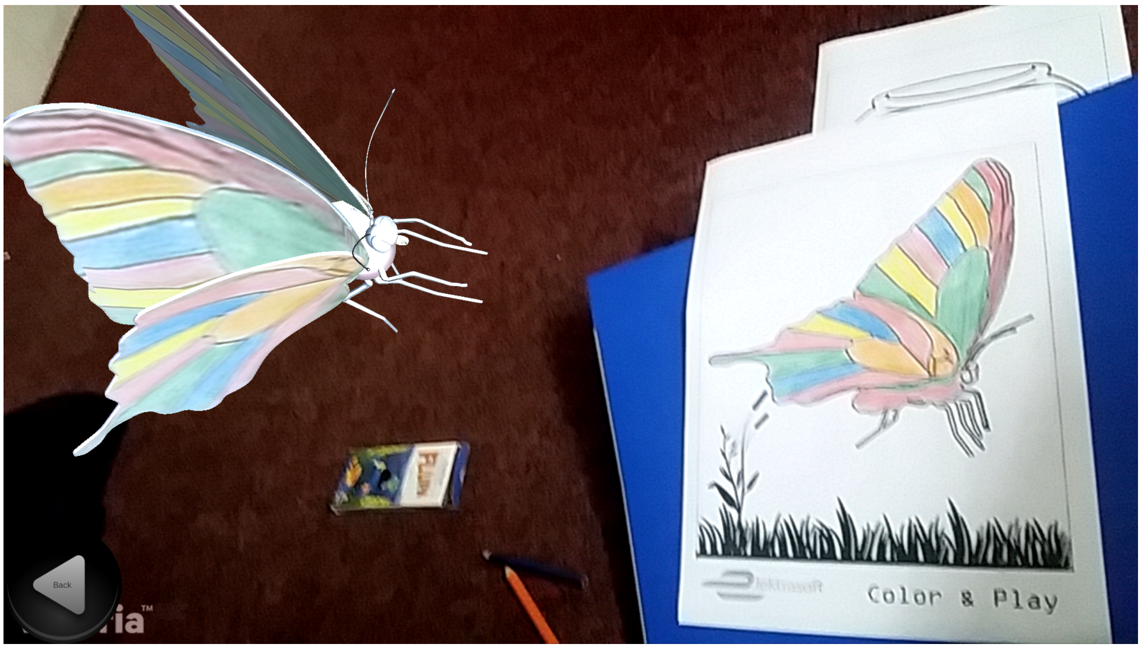


Figure 6.14: Sample Colored Screenshot 1

6.3.2 Integration Testing

Integration has been carried out as discussed in the Chapter 5. Each component was tested at the Unity3D IDE after importing all the modules into Unity3D i.e. Image Targets Unity Package, 3D models and performing live image processing techniques. The collaboration of the Models with scripts i.e. handling animations was also tested. Applying textures to 3D models after image processing was also separately tested.

Gradually after integrating every module and testing for a sample Image Target, the overall performance of detecting the image target, processing for colors and augmenting the 3D model became efficient. Sample screenshot of colored augmentation is shown in the Figure 6.14 and Figure 6.15.

6.4 Compatibility Test

Compatibility of the application was considered from the start of the development, as the decision was to make the application for Android platform, therefore, it was imposed that the application must run in all the Android supported device i.e. Tablets, mobile phones. Therefore by applying certain techniques in which we take the width and height of the current device and set our canvas and UI accordingly we resolved the compatibility issues of the screen sizes. The application was run on different devices i.e. tablets and mobile phones and there was the same experience on both the devices.

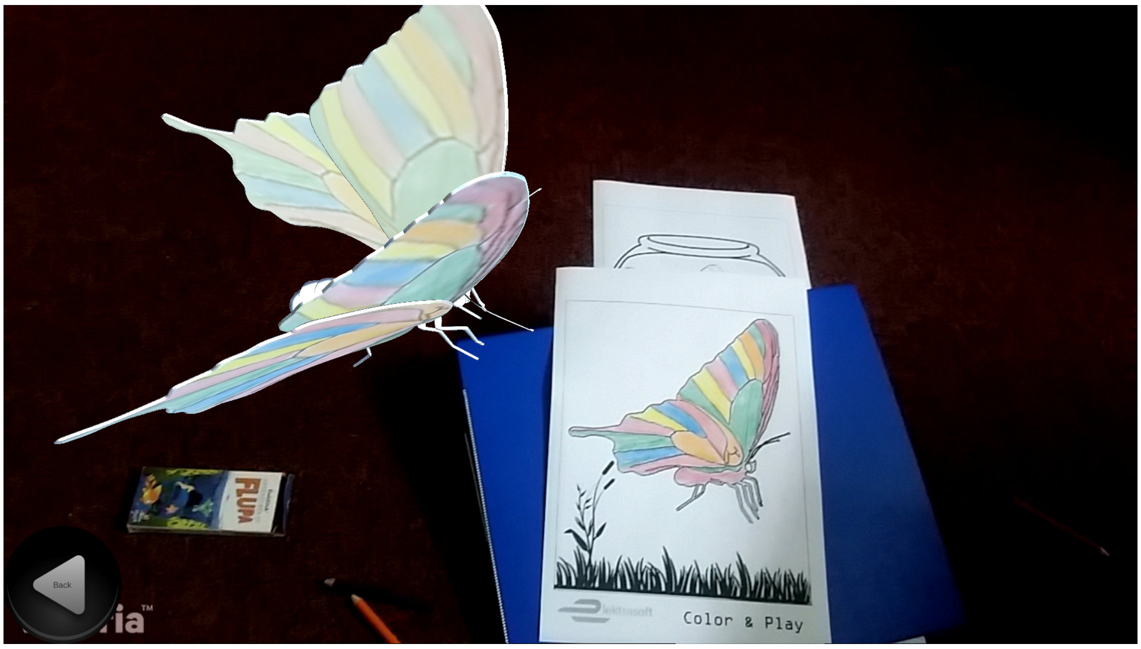


Figure 6.15: Sample Colored Screenshot 2

6.5 Exception Handling

Exception handling has been carried out throughout the development of the application. The major part of exception handling was related to registering camera for frames till image processing. Major exceptions occurred when the camera was not able to be registered and the format of the frames, but they have been handled accurately after trying multiple formats as discussed in the section 5.5. Image/Frame processing was also a complex part, in which different conversions were to be made like texture2D to Matrix and vice-versa, and accessing different locations of the image/matrix for operations like extracting sub-matrix. Although after deep concentrations about inputs, exceptions were gradually handled.

6.6 Load Testing

Application optimization was the first priority throughout the development. Although there is no such data intensive load implied on this application due to the nature of it, therefore we cannot supply the stream of data to it and check the behavior of the application.

6.7 Security Testing

Coloring Book with AR is related to children, and there is no such database stored which maintains any type of confidential records, therefore, there is no such database security implied in the application. Although for the copyrights issue, we would deploy the application with a single file called Android Package Kit discussed in the Section 5.7, this file can be extracted and some sort of structure can be viewed as there are always exceptions, but no one can make the solution out of that and build the application from a single *.apk* file.

6.8 Installation Testing

As the application is built for the Android platform, therefore there are numerous Android devices currently available. Of course, there are also some constraints such as Android version, which we have set to the minimum of 2.3.1 'Gingerbread' (API level 9). The device with this or above this version would be able to install the application and the devices with the camera devices. The *.apk* file has been installed on different devices of different companies and ran successfully. We are launching the application globally on Google Play Store.

6.9 Test Cases

Test Case ID	TC01
Test Case Title	Installing the application on Android Device
Description	This test case is related to transferring *.apk* to the Android device and then locating it on the mobile device and running the setup.
Test Steps	1. Plug-in the device to the computer where the setup is residing. 2. Transfer the setup to the desired location. 3. Unplug the device and locate the setup on the mobile device. 4. Open the setup file and accept the permissions the application requires to install. 5. Let the installer finish the installation and see it notify the successful message.
Expected Result	Application should be installed with the application icon in the menu.
Status	Pass

Table 6.1: Test case 1 - Installing the Application

Test Case ID	TC02
Test Case Title	Starting the Coloring Book with Augmented Reality
Description	This test case is related to opening the application from the start to test whether it runs without crashing at the start.
Test Steps	1. Switch on the mobile device or unlock the device on which the application is already installed. 2. Locate the icon of the application or look for Coloring Book with Augmented Reality named application. 3. Open/click on the icon with the specified name or icon of the application.
Expected Result	Mobile device should show splash screen of the application and bring the control to the main menu of the application.
Status	Pass

Table 6.2: Test case 2 - Opening the application

Test Case ID	TC03
Test Case Title	Tap Start Button
Description	This test case is related to opening the augmentation screen from the main menu to test whether the camera starts successfully to detect image targets.
Test Steps	1. While the application is open and the current activity/scene is the main menu, locate the ’Start’ button, which would be at the right bottom side of the screen and tap on it.
Expected Result	Main menu activity should be dismissed.
Status	Pass

Table 6.3: Test case 3 - Tapping the Start Button to dismiss the main menu activity

Test Case ID	TC04
Test Case Title	Open AR Camera (System evaluation)
Description	This test case is related to the response of the system about the tap on ’Start’ button.
Test Steps	1. After the ’Start’ button has been tapped, wait for about a second. 2. See if the camera has been opened and rotate to see if it normal camera view.
Expected Result	Camera view should be seen on the screen and it is to be done by the system.
Status	Pass

Table 6.4: Test case 4 - Opening AR camera

Test Case ID	TC05
Test Case Title	Search for Image Targets
Description	This test case is related to searching for Image Targets after the AR camera is opened with a normal camera view.
Test Steps	1. Print the provided high definition Image Targets from the printers 2. If the printer is not available, the image targets can also be opened on some digital device. 3. Color the Image Targets with the desired colors (uncolored image targets can also be used for augmentation). 4. Bring the Image Targets on the scene and make sure it can be seen in the camera view.
Expected Result	Cropped Image Target from the camera frame should be displayed in the small rectangle screen view (Note that this small screen showing the detected image target is only for the testing purpose and it won't be in the publishing/deployed version).
Status	Pass

Table 6.5: Test case 5 - Searching for Image Targets

Test Case ID	TC06
Test Case Title	Augmented Corresponding Model.
Description	This test case is related to the system responding to the Image Target detected or being tracked.
Test Steps	1. While on the camera view and pointing the camera towards the Image Target, keep the camera pointing towards it and wait for the response of the system to augmented the corresponding model whether colored or uncolored.
Expected Result	the Corresponding model should be augmented, colored augmented model in the case of colored image target.
Status	Pass

Table 6.6: Test case 6 - Analyzing the corresponding augmented 3D model

Test Case ID	TC07
Test Case Title	Tap Screen for Animations
Description	This test case is related to analyzing the animations after the model is augmented.
Test Steps	1. While the augmented model is there on the camera view, tap on the screen other than the UI elements. 2. Analyze the animation being played. 3. Adjust the camera view to properly see the augmentation but do not loose the image target from the camera view.
Expected Result	3D model should perform movements provided.
Status	Pass

Table 6.7: Test case 7 - Analyzing the animations of the augmented models

Test Case ID	TC08
Test Case Title	Tap Back Button
Description	This test case is related to going back to the main menu from the augmentation scene.
Test Steps	1. While the control is with augmentation scene, press the 'Back' button at the left bottom corner of the screen. 2. Wait till the control is shifted back to the Main Menu activity.
Expected Result	Main Menu activity should be displayed.
Status	Pass

Table 6.8: Test case 8 - Analyzing the working of 'Back' button of the augmentation scene

Test Case ID	TC09
Test Case Title	Tap Exit Button
Description	This test case is related to exiting from application and analyzing the working of Exit button provided in the main menu.
Test Steps	1. While on the main menu screen, tap the Exit button located at the left bottom corner of the main screen.
Expected Result	Application should be exited.
Status	Pass

Table 6.9: Test case 9 - Fully Dressed Description

Chapter 7

Conclusions

We presented an augmented reality based coloring book which will allow children to use their customized color combinations to color 2D characters and then analyze the corresponding animated 3D-model of their drawings. User study has been carried out and it was concluded that this application motivates the children to draw with more creative color combinations to see their imagined colored character in a real environment, furthermore, it improves the concept of the character which they cannot examine in their early age. The application is built upon different helping components but there are two novel techniques that helped in providing the major functionality (1) UV-mapping of 3D-model and (2) Live Image Processing. The first one is related to mapping 3D-model's meshes to the corresponding image target. By doing this, the material we apply to the 3D model is mapped correctly. For the second technique, live video frames are fetched and processed for locating the image targets area in it for removing the non-related environment. The exact area of the image targets in the frames needed to be extracted by applying certain techniques like finding the transform attributes which defines the scaling, rotation, and position of the image targets as the UV-mapping is done according to the image targets without the non-related environment. Once the image target is extracted from the frames, it can be applied to the mesh renderer of the 3D-model.

As the developed system is a mobile application, therefore, there are limited hardware resources and the sizes of applications also matter. Due to this issue, models in hundreds cannot be provided in a single application. A good extension to this application can be employing the cloud services, where the online database of image targets and 3D-models can be stored and there can be a user interface be provided a form where users can view the desired drawing and download that right away and see the augmentation.

Appendix A

User Manual

Some of the Image Targets made for the application are shown below which are the part of our Coloring Book. We have also included the QR-code. If user brings the QR-code reader towards this QR-code, QR-reader will take the user to download this application from Play Store.

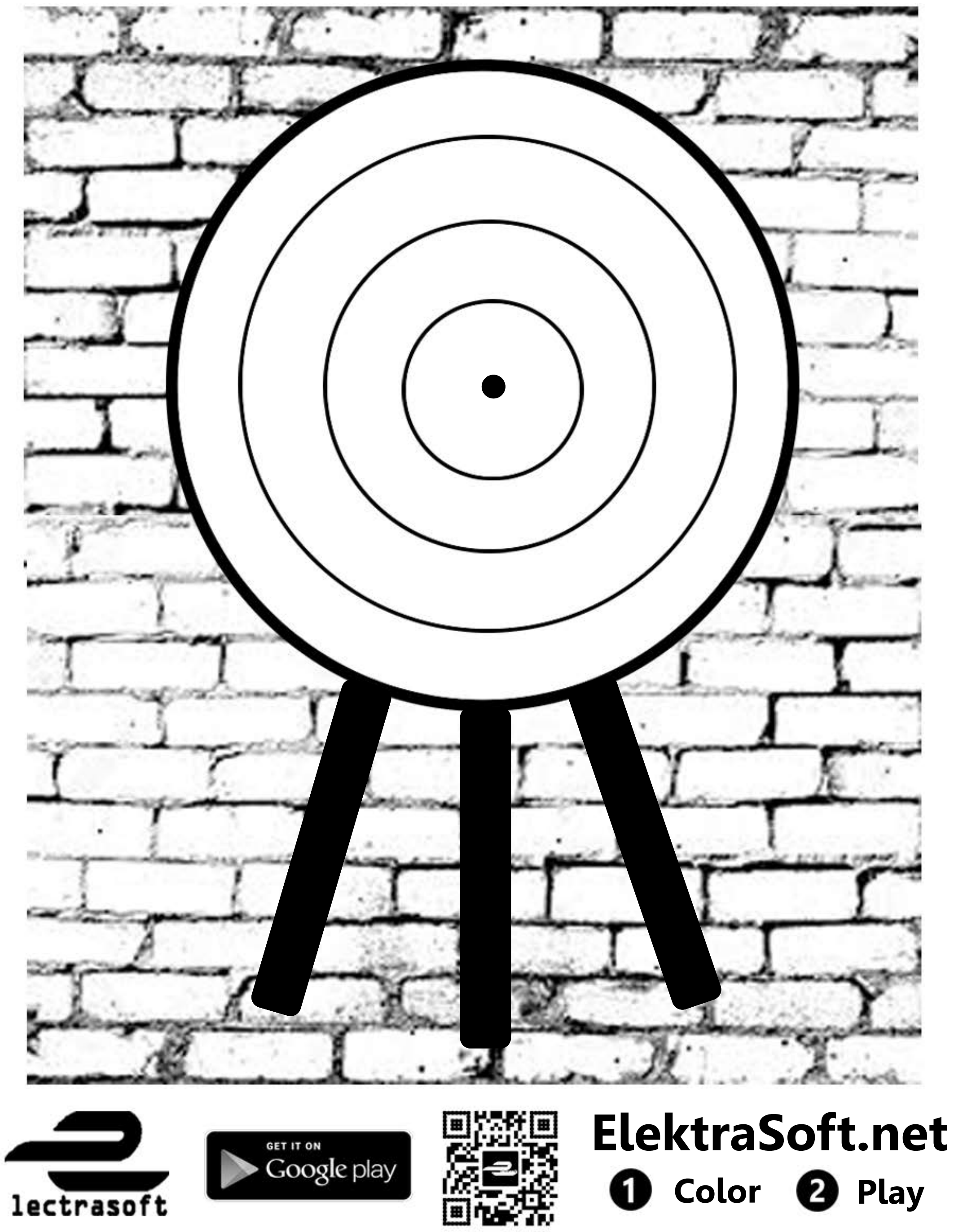


Figure A.1: Archer Image Target

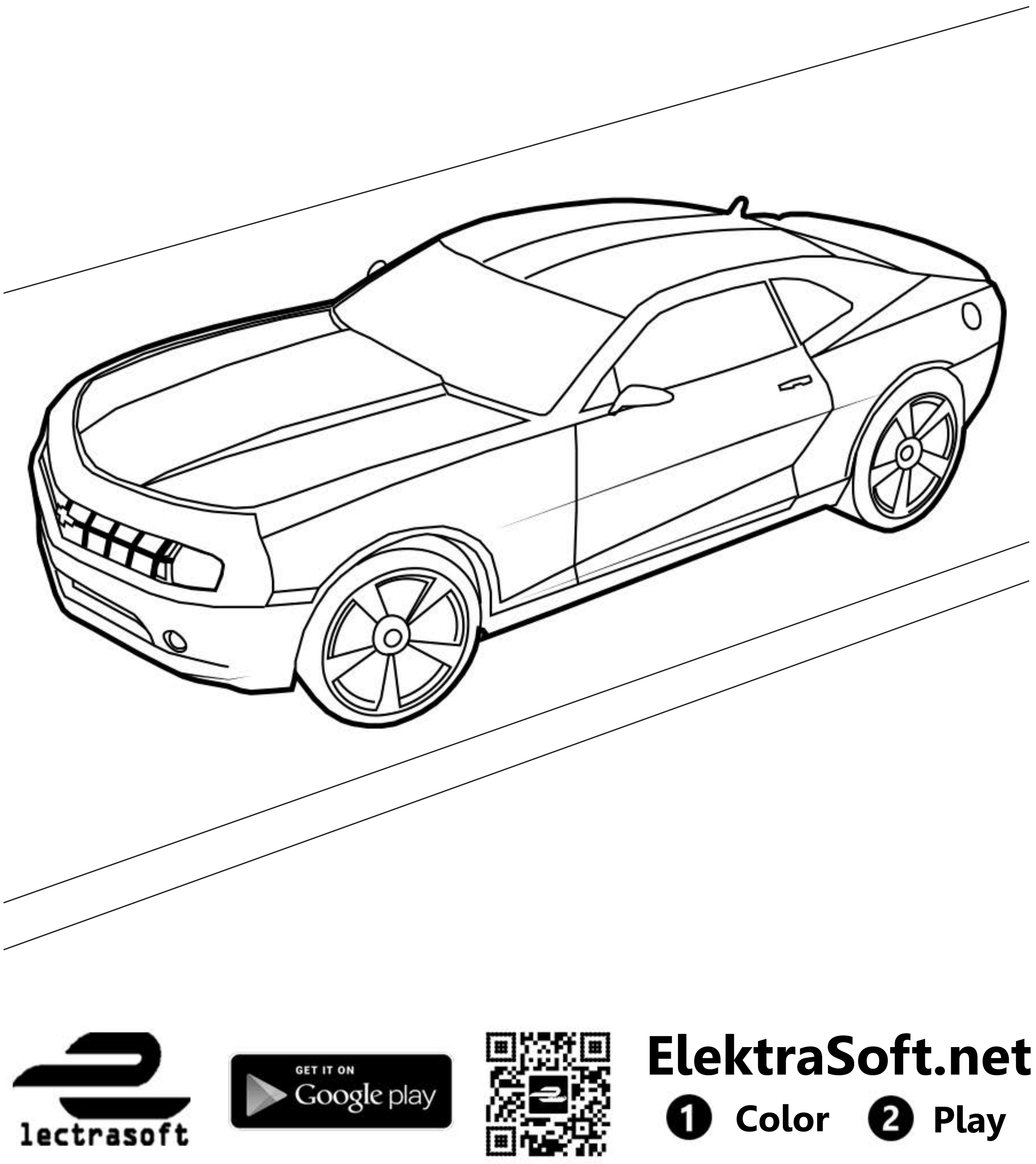


Figure A.2: Car Image Target

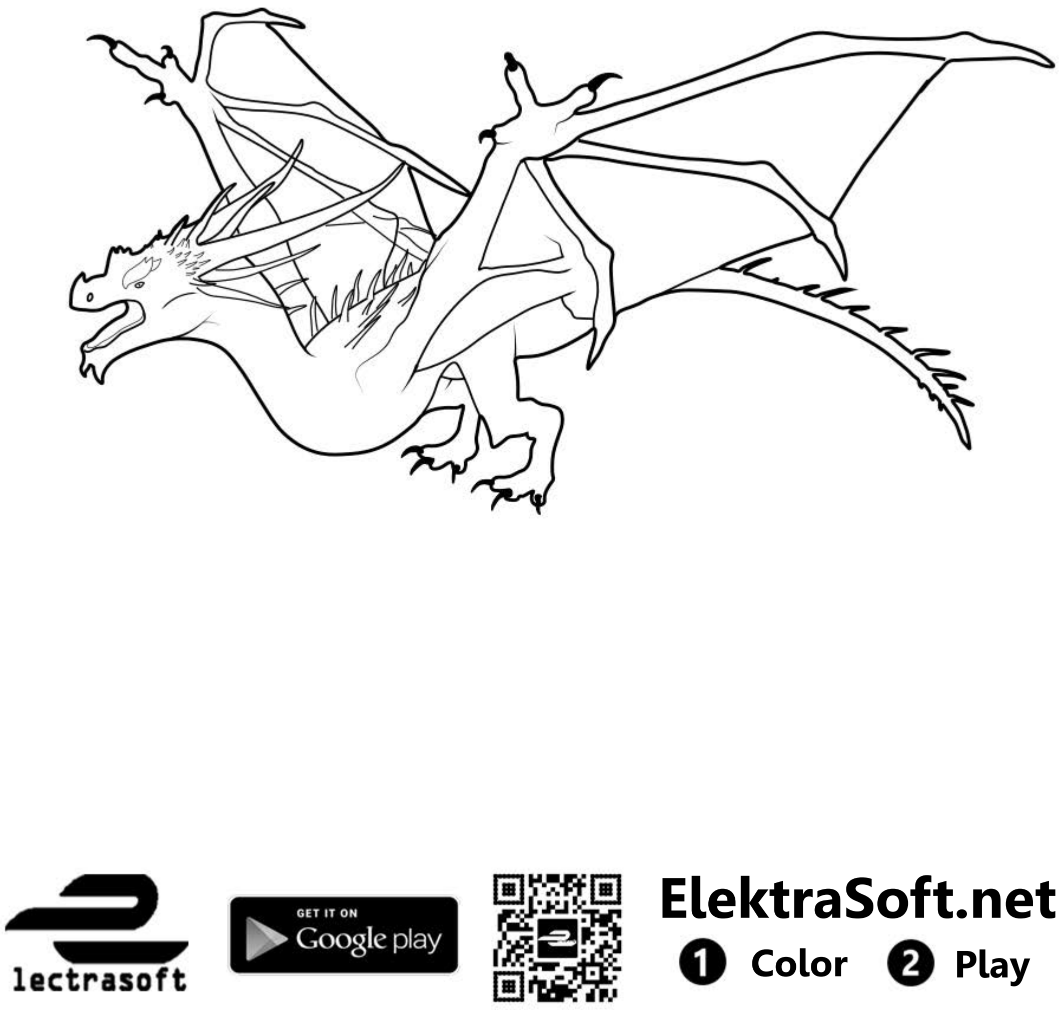


Figure A.3: Dragon Image Target

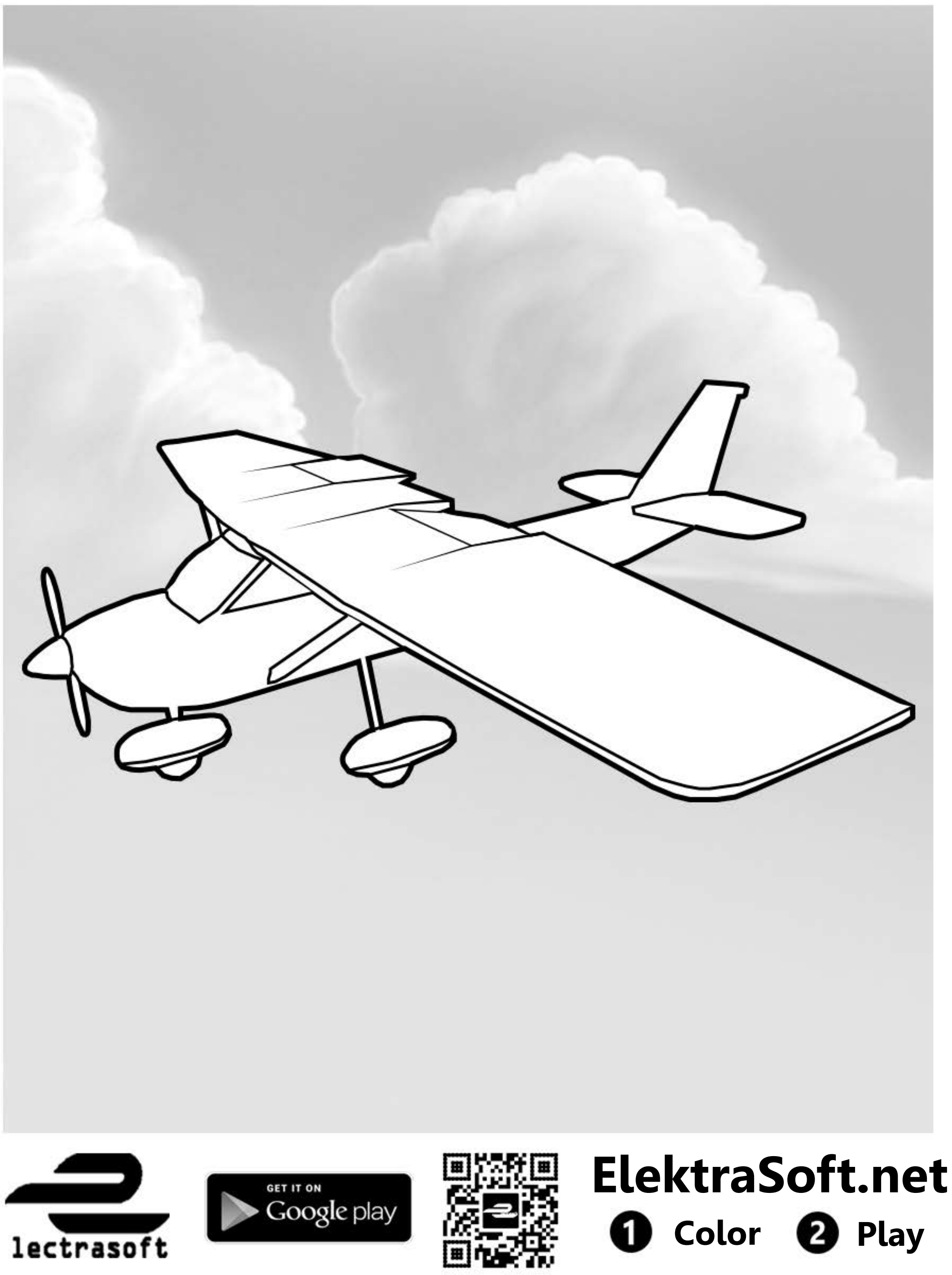


Figure A.4: Plane Image Target

References

[1] Margaret Rouse. What is augmented reality (ar) ? http://whatis.techtarget.com/definition/augmented-reality-AR. Accessed: 2016. Cited on p. 2.

[2] David G Lowe. Object recognition from local scale-invariant features. 2:1150–1157, 1999. Cited on pp. 3 and 4.

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. pages 404–417, 2006. Cited on pp. 3 and 4.

[4] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. pages 2548–2555, 2011. Cited on p. 3.

[5] Charles B Owen, Fan Xiao, and Paul Middlin. What is the best fiducial? pages 8–pp, 2002. Cited on p. 3.

[6] Mark Fiala. Comparing artag and artoolkit plus fiducial marker systems. pages 6–pp, 2005. Cited on pp. 3 and 7.

[7] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11):1330–1334, 2000. Cited on p. 4.

[8] Vuforia developer portal. https://developer.vuforia.com/. Accessed: 2016. Cited on pp. 4 and 9.

[9] Daniel Wagner, Gerhard Reitmayr, Alessandro Mulloni, Tom Drummond, and Dieter Schmalstieg. Pose tracking from natural features on mobile phones. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 125–134. IEEE Computer Society, 2008. Cited on p. 4.

[10] Mathieu Salzmann, Vincent Lepetit, and Pascal Fua. Deformable surface tracking ambiguities. pages 1–8, 2007. Cited on p. 5.

[11] Florent Brunet, Adrien Bartoli, and Richard I Hartley. Monocular template-based 3d surface reconstruction: Convex inextensible and nonconvex isometric methods. *Computer Vision and Image Understanding*, 125:138–154, 2014. Cited on p. 5.

[12] Read video frame data. http://www.mathworks.com/help/matlab/ref/videoreader.read.html. Accessed: 2016. Cited on p. 5.

[13] Ronald T Azuma. A survey of augmented reality. *Presence: Teleoperators and virtual environments*, 6(4):355–385, 1997. Cited on p. 7.

[14] Tobias Domhan, Kurs TIT07INA, and Gutachter der Dualen Hochschule. Augmented reality on android smartphones. *Studiengangs Informationstechni. Dualen Hochschule Baden-Württemberg Stuttgart*, 2010. Cited on p. 7.

[15] Lorenzo Porzi, Elisa Ricci, Thomas A Ciarfuglia, and Michele Zanin. Visual-inertial tracking on android for augmented reality applications. pages 35–41, 2012. Cited on p. 8.

[16] J. R. Smith and Shih-Fu Chang. Single color extraction and image query. In *Image Processing, 1995. Proceedings., International Conference on*, volume 3, pages 528–531 vol.3, Oct 1995. Cited on p. 8.

[17] Christine Guillemot and Olivier Le Meur. Image inpainting: Overview and recent advances. *IEEE signal processing magazine*, 31(1):127–144, 2014. Cited on p. 8.

[18] Stéphane Magnenat, Dat Tien Ngo, Fabio Zünd, Mattia Ryffel, Gioacchino Noris, Gerhard Rothlin, Alessia Marra, Maurizio Nitti, Pascal Fua, Markus Gross, et al. Live texturing of augmented reality characters from colored drawings. *IEEE transactions on visualization and computer graphics*, 21(11):1201–1210, 2015. Cited on pp. 8 and 11.

[19] Dieter Schmalstieg, Tobias Langlotz, and Mark Billinghurst. *Augmented Reality 2.0.* Springer, 2011. Cited on p. 9.

[20] Phoenix Toews. *An augmented reality framework for layering media over the contemporary landscape*. Cited on p. 9.

[21] Ramon Ivan Barraza Castillo, Osslan Osiris Vergara Villegas, and Vianey Guadalupe Cruz Sanchez. A mobile augmented reality framework based on reusable components. *IEEE Latin America Transactions*, 13(3):713–720, 2015. Cited on p. 9.

[22] Adrian Clark and Andreas Dünser. An interactive augmented reality coloring book. pages 7–10, 2012. Cited on pp. 9 and 11.

[23] Nicolas Imbert, Frederic Vignat, Charlee Kaewrat, and Poonpong Boonbrahm. Adding physical properties to 3d models in augmented reality for realistic interactions experiments. *Procedia Computer Science*, 25:364–369, 2013. Cited on p. 9.

[24] Sung Lae Kim, Hae Jung Suk, Jeong Hwa Kang, Jun Mo Jung, Teemu H Laine, and Joonas Westlin. Using unity 3d to facilitate mobile augmented reality game development. pages 21–26, 2014. Cited on p. 9.

[25] Disney research. https://www.disneyresearch.com/. Accessed: 2016-09-04. Cited on p. 11.

[26] Aleksandr. https://blogs.unity3d.com/2014/09/03/documentation-unity-scripting-languages-and-you/. Accessed: 2016-09-19. Cited on p. 40.