Evaluation of Augmented Reality Frameworks for Android Development

Master Thesis

Graz University of Technology

submitted by
Iulia Marneanu

Advisor: Assoc.Prof.PhD. Martin Ebner

Graz, June 2014

Abstract

Augmented Reality (AR) is the evolution of the concept of Virtual Reality (VR). Its goal is to enhance a person's perception of the surrounding world. Augmented Reality is a fast growing state of the art technology and a variety of implementation tools exist today. Due to the heterogeneity of available technologies, the choice of the appropriate tool for a mobile application is difficult to make.

The present thesis offers a systematic way for choosing such an open tool that fits all the requirements for creating an exciting Augmented Reality experience. For this purpose, a thorough research is conducted in the area of Augmented Reality and subsequently evaluating a finite number of free Augmented Reality frameworks. Criteria are defined that might influence the experience of an Augmented Reality application and the chosen tools are tested against them. As a final step, a mobile app is implemented which integrates the above mentioned tools and provides an environment for testing and storing the results.

A number of use cases are defined as a collection of criteria. The importance of such a criterion depends on the use case it defines. The framework that best fits the use case's requirements is determined and the choice explained. Therefore, it is proven that a tool might be capable in one context while outperformed in others.

Zusammenfassung

Augmented Reality (AR) ist die Weiterentwicklung des Konzepts der Virtual Reality (VR). Das primäre Ziel ist es die Wahrnehmung einer Person mit der umgebenden Welt zu verbessern indem diese digital erweitert wird. Augmented Reality ist ein sehr schnell wachsendes (Forschungs-)gebiet und es existieren heute bereits eine Vielzahl unterschiedlicher Implementierungsmöglichkeiten. Aufgrund der Heterogenität der verfügbaren Technologien ist die Wahl des geeigneten Werkzeugs für eine mobile Anwendung schwierig.

Die vorliegende Arbeit versucht eine systematische Methode für die Wahl eines offenen und geeigneten AR-Systems zu finden. Zu diesem Zweck wird eine gründliche Literaturrecherche auf dem Gebiet von AR-Systemen durchgeführt, diese gesichtet und kategorisiert. Es werden Kriterien definiert, welche die Erfahrung mit Augmented Reality-Anwendungen beeinflussen und Usecases definert. Als letzter Schritt wird eine mobile Anwendung programmiert, die die oben erwähnten Werkzeuge integriert und die definierten Kriterien gegenüberstellt.

Die Masterarbeit schließt mit dem Ergebnis, dass je nach Anwendungsfall eine bestimmte AR-Umgebung zu wählen ist.

Acknowledgement

This master thesis has been completed at the Institute of Information Systems and Computer Media of Graz University of Technology.

I would like to thank my supervising professor, Martin Ebner, for guiding me in this work. Special thanks to Katja Kramberger for testing the app and using her skills to merge images together. I would like to thank Markus Berger for emotional support and Timea Turdean for proof reading the thesis.

At this point I would also like to thank Thomas Rößler and Ernesto Rico-Schmidt for facilitating this thesis.

Graz, June 2014 Iulia Marneanu

Contents

List of Figures

List of Tables

1. Introduction

This chapter reveals the incentive behind this thesis by answering a couple of questions. What will the Augmented Reality market size be in the next five years, what are the factors for its continuous growth, what are the key market trends, who are the leading software companies? Given the motivation, the goal is to find the best software for a certain context. The thesis content is outlined within this chapter.

1.1. Motivation

At the end of 2013, a number of research reports have been published concerning the future market of Augmented Reality. It has been forecast that Virtual Reality and Augmented Reality systems will be worth $1.06 billion by 2018 [Mar13a]. Moreover, the Global Augmented Reality market will grow 132.2 percent from 2013 to 2018 [Lim13]. The Juniper Research, in their "Mobile Augmented Reality, Smartphones, Tablets and Smart Glasses 2013-2018" [Res13], has shown that Mobile Augmented Reality will reach 200 million users globally by 2018 from today's 60 million.

There are several factors that contribute to this market growth. Major advancements are done in computer technology and internet connectivity. As the global telecommunication market is upgrading, developed countries upgrade to 4G networks while developing countries upgrade to 3G networks. Thus, by introducing mobile internet packages, the demand for mobile internet applications is growing. This leads to one of the key factors, the increasing number of Augmented Reality applications. The demands are spread over a number of industries such as gaming, education, m-commerce, medical, defense, consumer electronics and industrial.

Augmented Reality applications run on different devices. Gaming and e-learning apps have recently been developed for smartphones, tablets and smart glasses. Military and aerospace applications are used in a more advanced stage for head mounted displays and head up displays [Mar13b]. The usage of Augmented Reality applications will evolve to fit the device, the tablet becoming the go to for multimedia content and second screen apps while the smartphone will be a platform for navigation based apps [Wig13].

Augmented Reality technology has made great progress in mobile phones, making use of the equipped camera and GPS. It has great potential for advanced learning and teaching technologies in education and a growing fascination in game applications. The technology will expand to include fitness and lifestyle apps, with the possibility of breaking the digital-physical wall within the social media. A new generation of users will arise which will combine the personal nature of mobile devices with the Internet's wealth of accessible information [Wig13].

Augmented Reality applications are already adopted by countries like China and Japan. There are a number of Augmented Reality software companies in the European region and the number is expected to continue growing over the next years. The key vendors that provide Augmented Reality software, known from this point on as frameworks, are Total Immersion from France, metaio from Germany, Vuforia from Austria and catchoom from Spain. Other companies mentioned in this thesis are ARLab and ARToolworks.

1.2. Goal of the Research

As previously stated, the Augmented Reality market is growing fast and the demand for new applications is increasing. The question that needs to be answered, from a programmer's point of view, is: which is the best framework for developing an Augmented Reality application for a specific context?

This paper investigates the free Augmented Reality frameworks for mobile technology with a special focus on Android based devices. By free frameworks it is meant that only open source or under demo licensing frameworks are considered. The evaluation is directed at the markerless tracking methods. The goal of this research is to identify the best suited framework given specific constraints. The current thesis will prove that there is no best framework, but rather that different constraints are best suited by different frameworks.

1.3. Methodological Approach

As a first step, a vast research is performed on the available frameworks, at the time of the current thesis. From these frameworks, the ones that support Android development are chosen. A number of these frameworks are then evaluated. For evaluation purposes, various constraints, from now on called criteria, have been proposed. These criteria define specific contexts, from this point on known as scenarios.

For a better visualization of the results, a test app has been developed. Therefore, different frameworks can be tested in real time and the results are saved in a local database. The results are then presented in different visualizations. Furthermore, two or more frameworks can be compared against a criterion or a scenario.

1.4. Thesis Outline

The next chapter defines the notion of Augmented Reality, exemplified by some of its applications. It also includes a few words on the mobile environment and hardware constraints. Chapter three gives an overview of the available Augmented Reality frameworks for Android development, explaining the reasons why some of them have not been subject of evaluation in the thesis.

Chapter four introduces the considered evaluation criteria, the constraints that an Augmented Reality framework has to overcome. What are they and how are they chosen? These criteria are later on used to define scenarios and evaluate each Augmented Reality framework. For this purpose, a testing app is implemented. More about what the app does and how it works is described in chapter five.

Chapter six analyzes the results obtained by the testing app and provides the scenarios definitions as well as the Augmented Reality framework that best fits each of them. The work of this thesis is concluded in chapter seven. There is no best Augmented Reality framework, rather a best suited one which can be determined in specific contexts.

2. Augmented Reality

Augmented Reality is an emerging technology, very promising, with a growing popularity on mobile devices. The Augmented Reality technology augments the real world with digital data. As a result, the perception of the surrounding world is enhanced, real and virtual objects intertwine and interact. The user experience has undergone great innovation by changing the way the world is perceived, what the user hears, feels and sees[Bon14]. The Augmented Reality technology is used in many applications from education to medicine, from games to military and much more.

2.1. Definition

Numerous definitions and variations can be found describing Augmented Reality. Wikipedia [Wik14] defines it as a live copy of a physical real-world environment view whose elements are augmented by adding computer-generated information to it such as sound, video, graphics or GPS data. Kevin Bonsor [Bon14] defines Augmented Reality as the technology that integrates computer graphics into the real world. Ronald T. Azuma was the one to define the three main characteristics of Augmented Reality [Azu97a]:

- It combines real and virtual objects, such that both virtual reality and the reality itself are found in the final product.
- Is interactive in real time, thus just by introducing virtual objects into real scenes without interaction does not lead to Augment Reality technology.
- Is registered in 3D, meaning that no 2D superimposition over live video can be considered Augmented Reality.

There are many descriptions for Augmented Reality but no official definition. P. Milgram, H. Takemura, A. Utsumi and F. Kishino came the closest [PMK94] by defining the Reality-Virtuality Continuum. The Reality Virtuality Continuum is the environment characterized by the space between the Real Environment at one end and the Virtual Environment at the other. The Real Environment is the surrounding view, bound by physics laws, solely composed of real objects. The Virtual Environment is on the other hand purely modeled by virtual objects. Laws of physics do not apply, it might copy the real-world environment, but it is not a requirement as in the end it is a computer generated world.

The Reality-Virtuality Continuum is shown in Figure 2.1. Everything in between the two described environments is called Mixed Reality and contains objects from both worlds, at different virtual degrees. The closer to the Virtual Environment, the more virtual objects are found. Augmented Reality finds itself on the left side of the line, closer to reality, thus offers information that would otherwise not be detected. Augmented Reality supplements the real-world, it does not replace it.

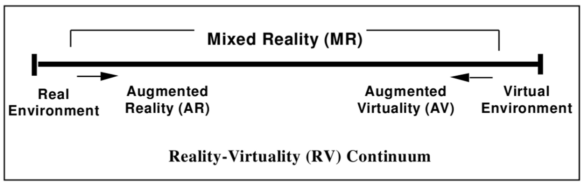


Figure 2.1.: Reality-Virtuality Continuum[1]

2.2. Technology

For Augmented Reality to work, a number of hardware requirements must be met. Additionally, one of the different types of platforms must be chosen on which the Augmented Reality system will run. These components are essential for fixed and mobile environments.

Hardware Requirements

The hardware demands are:

- An input device such as a computer or a mobile device.
- A display, either a monitor or a display screen.
- A camera.
- Sensors for tracking and detecting like GPS, compass, accelerometer, optical sensors, RFID and wireless sensors.

Additionally, a connection to the internet might be necessary as well as CPU power, powerful 3D graphics and fast video processing. Some Augmented Reality systems require a physical object, a marker, that will be recognized and which indicates the place for the virtual object augmentation.

[1] Source: `http://wiki.commres.org/pds/Project_7eNrf2010/_5.pdf` (March 2014)

Figure 2.2.: Augmented Reality Glasses[2]

Platforms

A number of platforms are currently used to run Augmented Reality applications on:

- Personal computers equipped with a webcam: This platform is usually used with portable markers, and more recently in collaboration with XBox.
- Smartphones and tablets: The smartphone is the most popular platform for Augmented Reality apps today. In addition to the camera and display screen, they are provided with compass and GPS capabilities.
- Wearables: In this category are the Augmented Reality glasses, shown in Figure 2.2, head mounted displays, shortly HMDs, and heads up displays, known as HUDs.

Basic Process

Augmented Reality is categorized into fixed and mobile systems, by means of mobility. Fixed systems cannot easily change their location while mobile systems offer free movement. The same process applies for both categories, regardless of the type of Augmented Reality system, either visual based or location based. The Augmented Reality types are explained further down in the chapter.

Greg Kipper and Joseph Rampolla gave a description of the basic process by exemplifying it on the creation of marker-based Augmented Reality in [KR12]. The steps are illustrated in Figure 2.3.

[2] Source: `http://ces.cnet.com/8301-35304_1-57616909/wearables-with-augmented-reality-are-\mind-blowing-and-an-ethical-nightmare/` (March 2014)

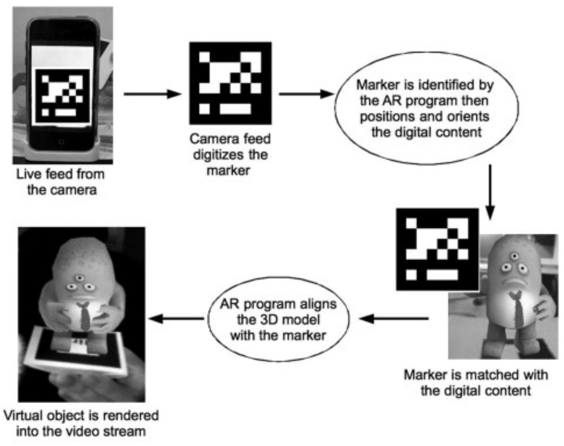

Live feed from the camera
Camera feed digitizes the marker
Marker is identified by the AR program then positions and orients the digital content
Marker is matched with the digital content
AR program aligns the 3D model with the marker
Virtual object is rendered into the video stream

Figure 2.3.: The process for the creation of Augmented Reality[3]

1. Live video feed from the camera of the device.
2. Marker identification by detecting the borders and creating binary encoded patterns.
3. Positioning and orientation of the 3D object with respect to the marker, positioning the digital content with the physical marker.
4. Matching the marker symbol to the assigned digital content.
5. Alignment of the 3D model to the marker.
6. Rendering of the 3D object into the video frame, such that the Augmented Reality content is visible on the display.

Implementation

There are various Augmented Reality systems available. Nevertheless, the basic components of the implementation are the same: videoframe input, an algorithm and a way to augment the digital content into the frames [MC]. The current thesis evaluates the external influences on the videoframe input.

The first step in the workflow of the implementation process is the recognition of an object, such as a marker, an image or a face. The object is then traced in real time, action known as tracking. Tracking is defined by the location and orientation of the camera. As a result, the virtual content, which can be anything from text or video to 3D objects, is overlapped on the object at the right scale and pose.

[3] Source: `http://ftubhan.tugraz.at/han/SCIENCEDIRECT/origin-ars.els-cdn.com/content/image/3-s2.0-B9781597497336000024-f02-10-9781597497336.jpg` (March 2014)

2.3. Applications

Augmented Reality is being used in a number of industries. Applications have been developed for different purposes and their number is expected to grow exponentially over the next years[Mar13a]. Here are some examples of the most popular industries, as well as emerging ones.

Military

The military has been a strong advocate towards the development and use of Augmented Reality systems. As military operations increase in diversity and complexity, the need for new operations and training tools grows with them. Augmented Reality tools have been developed for decades for military specific requirements and capabilities, both for military operations and training programs.

Augmented Reality is used to overlay military information without interfering with the surrounding environment by obstructing the vision. This information is added to the user's view and is presented according to the user's role in the mission [MAL11]. Instructions, maps, enemy locations and so on are considered to be relevant information.

The use of Augmented Reality in the military industry dates back to the launch of HUDs. It has applications from live fire training exercises to complex situation awareness. By superimposing information, like satellite view or blueprints, the soldier can keep track of the location of his/her team members and of the enemies. Moreover, interactive battlefield medical support is provided, as well as basic information such as building and street names, orientation.

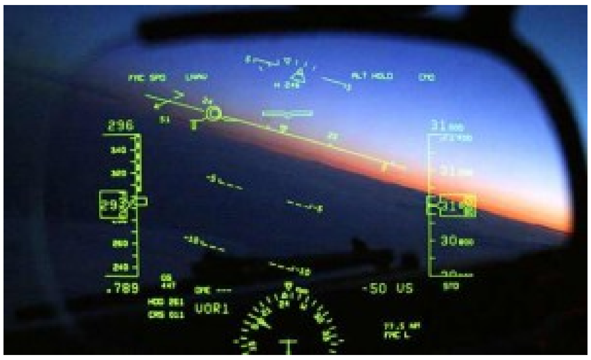


Figure 2.4.: Fighter pilot Augmented Reality application[4]

One of the most common Augmented Reality application is the fighter pilot application (Figure 2.4) which adds navigation and landscape information to the pilot's view.

[4] Source: `http://blog.tamar.com/2010/02/augmented-reality-is-here-some-good-apps-to-try/` (March 2014)

Gaming and Entertainment

One of the most popular development branches of the Augmented Reality technology is Gaming and Entertainment. As stated in the Augmented Reality in Gaming and Entertainment report [Com13], the mobile market is becoming stronger and the demand for new mobile Augmented Reality games is increasing.

The gaming experience is innovated by the introduction of Augmented Reality. The player can solve puzzles, race cars, go treasure hunting, play with virtual objects and more. By pointing the camera, the virtual object comes to life on the screen and the game begins.

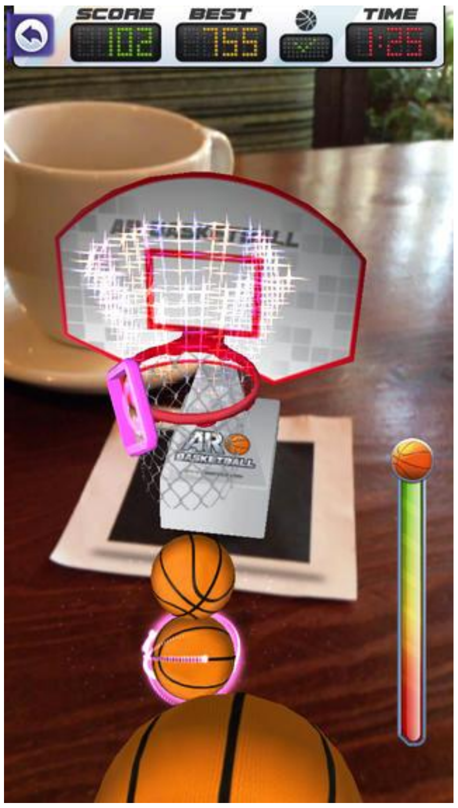


Figure 2.5.: ARBasketball[5]

Last November Google launched Ingress, an experimental Augmented Reality game, where smartphones act as portals to an alternate reality. ARBasketball app (Figure 2.5) shows a hoop on top of a printed marker, and by using the swipe gestures, the ball acts as if it was thrown, accumulating points. PaintBall app mixes real time 3D computing with the real-world seen through the mobile's camera, the goal being to splash your friends using various augmented paintball guns. SpecTreck is a gost hunting game which uses the smartphone's GPS and camera, showing a hand-radar when the mobile is on horizontal position.

[5] Source: `http://deepknowhow.com/2013/12/11/augmented-reality-games-android-iphone/` (March 2014)

Education

Augmented Reality enhances education by providing an interactive experience, thus redefining the learning process to develop critical thinking and a deeper understanding of the subjects. Teaching is traditionally done behind desks in classrooms. Augmented Reality introduces informal teaching environments, and so the students can take control of their own learning, in their own style, following their own unique discovery path.

Surveys show that at the beginning of the school year in 2013, 70% of the American teens with ages between 13 and 17 owned smartphones [nie13],[ME13]. With such a high number of smartphones, which is constantly increasing, it would be easy to introduce Augmented Reality applications for educational purposes in the teaching curriculum from a technical point of view. By running the technology on a smartphone or a tablet, students can train specific skills, model objects, find the hidden digital information in Augmented Reality books and even take advantage of the discovery-based learning method.

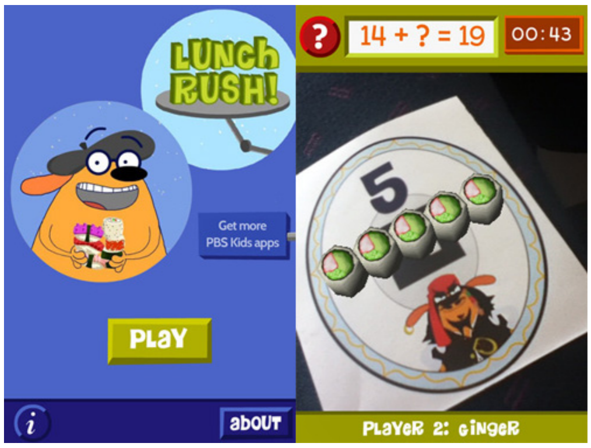


Figure 2.6.: FETCH! Lunch Rush app[6]

There are already a lot of mobile Augmented Reality applications that have educational value. Word Lens app translates words in real time. Google Sky Map app shapes astronomy to be fun and interesting by pointing the smartphone to the sky and identify stars and constellations. GeoGoogle app teaches the fundamentals of geography, latitude, longitude, altitude and distance to the user's surroundings. Used by elementary students to learn math, FETCH! Lunch Rush app (Figure 2.6) teaches addition and subtraction in the real-world. Homework Mini-Lessons is used by the teacher to leave a video on the homework sheet and by scanning the paper, the pupil gets clues to help him/her solve the problem.

[6] Source: `http://www.hongkiat.com/blog/augmented-reality-apps-for-education/` (March 2014)

Medicine

Medicine helps people to live a better life. Augmented Reality provides necessary and relevant information in doing just that. Access to this information can help in life and death situations. Augmented Reality brings the relevant information instantly to doctors right in front of their eyes when they need it [Pap14]. Patients and active users can take advantage of this new technology for their own needs, use it to maintain their health and even study different medical topics and terminology.

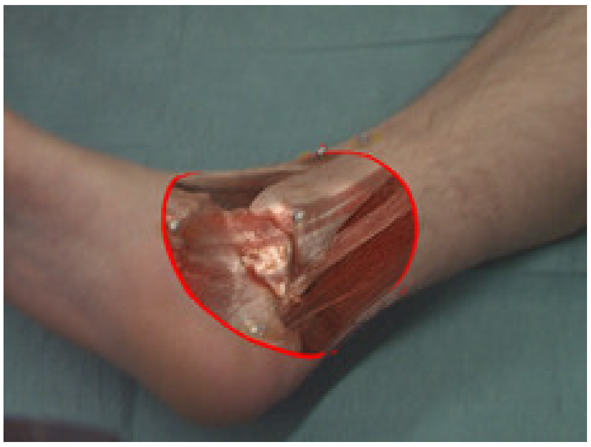


Figure 2.7.: Radiography superimposed on live image[7]

By superimposing graphics on bodies, such as virtual X-rays, view of anatomical parts or ultrasound real time images, the inside of the patient becomes visible. Organ placement, internal tissues, veins and arteries can be identified and the information used for diagnosis and therapy. Figure 2.7 shows just such a 3D bone model augmentation.

New on the market are the wearables. These devices can send real time notifications when the heart rate is too fast or blood oxygen levels are too low. Wearables are used to receive critical health information. Google Glasses are used for surgical purpose and training. The smartphone has become an informal tool in the hospital. There exist a number of Augmented Reality apps, both for medical professionals and for patients.

Nurses use Evena's Eyes-On Glasses, an Augmented Reality platform, to identify a patient's vein in real time. Augmented Reality is useful during complicated medical procedures when concrete knowledge is needed for different parts of the body. Tumor surgeries have been performed using an app developed by the german Fraunhofer Institute for Medical Image Computing MEVIS. A view from such a surgery is provided in Figure 2.8.

[7] Source: `https://www.in.tum.de/forschung/forschungs-highlights/medical-augmented-reality.html` (March 2014)

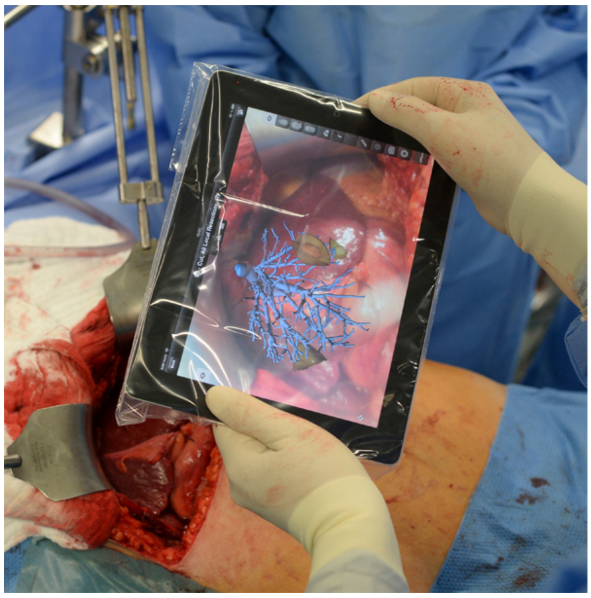


Figure 2.8.: MEVIS[8]

A patient can study the condition of his/her eye using the EyeDecide Augmented Reality app. Doctor Mole-Skin Cancer is used by everyday users to scan a mole for feedback, identifying cancerous moles.

Architecture and Interior Design

Architecture and interior design would benefit enormously by introducing Augmented Reality in their projects. Architects, interior designers and clients can easily and quickly see models of the final design. Moreover, they could solve any misunderstanding of the design or misinterpretation of the client's requirements. The clients would have a better grasp of the ideas and creativity put into the design and have the chance to modify or redefine their needs.

The models can be viewed on smartphones and tablets. Vizard VR software developed an interactive and intuitive application of live 3D rendering of an architecture model. Figure 2.9 provides such an example. Given the floor plan, the customer slides the mobile phone over it and navigates through the scene.

[8] Source: `http://www.medgadget.com/2013/08/augmented-reality-ipad-app-guides-surgeons-during-\tumor-removal.html` (March 2014)

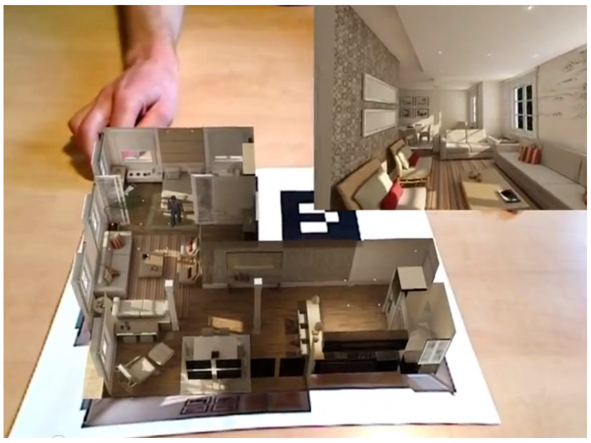


Figure 2.9.: Augmented Reality Architecture Exploration with Interactive Map[9]

Ikea has put on the market a mobile app for inserting products from the Ikea catalogue into the house of the customer, as depicted in Figure 2.10. The customer sees the virtual furniture at the real scale on the phone screen. S/he can change the color and size of the product, testing different products to find the right one.

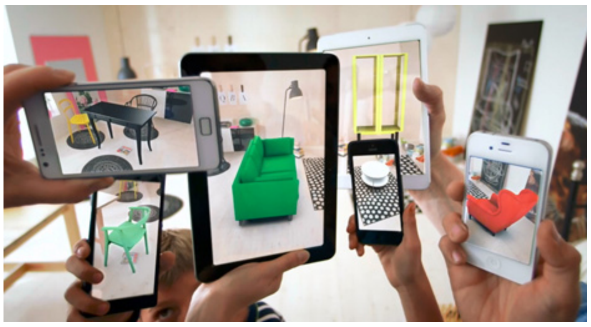


Figure 2.10.: Ikea Catalogue app[10]

mCommerce

eCommerce is the industry that takes most advantage of the power of the internet by facilitating online business. Every transaction, selling or buying of products or services, that is completed online falls under eCommerce, short for electronic commerce. Thus the transaction takes place solely by use of electronic communication without physical exchange.

[9] Source: `http://www.worldviz.com/newsletter/augmented-reality-architecture-exploration-\with-interactive-map` (March 2014)
[10] Source: `http://weburbanist.com/2013/08/06/virtual-interior-design-augmented-reality-\ikea-2014-catalog/` (March 2014)

With the advancements done in wireless technology, mobile eCommerce, known as mCommerce, becomes the next generation of eCommerce. Transactions are done on wireless handheld devices such as smartphones, tablets or PDAs.

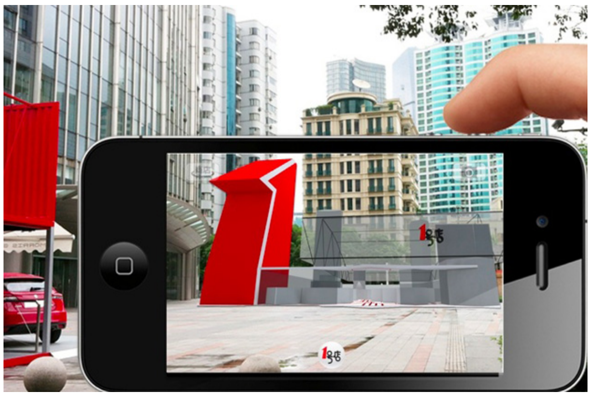


Figure 2.11.: Unlimited Yihaodian[11]

A fashion app launched by eBay offers the customer the opportunity to virtually try on sunglasses. By using the front-facing camera, the user can see himself/herself on the mobile screen and calibrate the glasses' shape on his/her face. Unlimited Yihaodian app uses Augmented Reality to create supermarkets on uninhabited areas (Figure 2.11), such that the user can virtually see the supermarket, walk around and shop for products that will later be delivered at home.

2.4. Types

In order to augment the real world, the Augmented Reality system must know the position and the orientation of the camera. In other words, the location of the user and what is s/he looking at. Thus, given a calibrated camera, the virtual objects are rendered at the right position.

The process that calculates the relative pose of the camera is called tracking. Tracking is a fundamental integrated part of Augmented Reality systems. Tracking quality is measured by different criteria, such as camera quality, light source and so on. These criteria are identified and defined in Chapter 4.

Based on the hardware and technical capabilities, the tracking methods have been classified in optical tracking methods, location based tracking methods and hybrid methods. The real world is enhanced using the main three techniques with respect to the tracking process: marker-based tracking, markerless tracking and GPS tracking.

[11] Source: `http://popupcity.net/can-augmented-reality-supermarkets-revitalize-vacant-urban-lots/` (March 2014)

Marker-based tracking and markerless tracking are both visual tracking methods while GPS tracking is a location based tracking method. Marker-based tracking has been widely used because is easy to implement. A number of toolkits are available for marker-based development.

Markerless tracking is gaining more and more ground to the marker-based tracking, as it does not require generating specific markers but rather uses normal images. This technique smooths the way for the recognition of real objects and their augmentation.

GPS tracking is frequently used, though having a lower accuracy than the visual tracking methods. Paired up with one of the already mentioned methods, a hybrid Augmented Reality system can be built. A more detailed description of these three techniques follows.

The present thesis focuses only on markerless tracking. Marker based tracking has become obsolete [Dol12]. Special fiducial marks have to be created in order to be tracked. They have to be maintained over time, which is also a costly operation. Markerless based tracking on the other hand, can target any part of the surrounding environment. Any image or object can be used as targets without being tied to a specific marker.

2.4.1. Marker Based Tracking

Visual tracking methods determine the position of the camera by making observations from the visible surroundings. It takes a considerable amount of time to collect enough data for such a deduction. The marker based tracking method uses markers to facilitate the estimation of the camera's pose and calculate the shifts in relative pose over time.

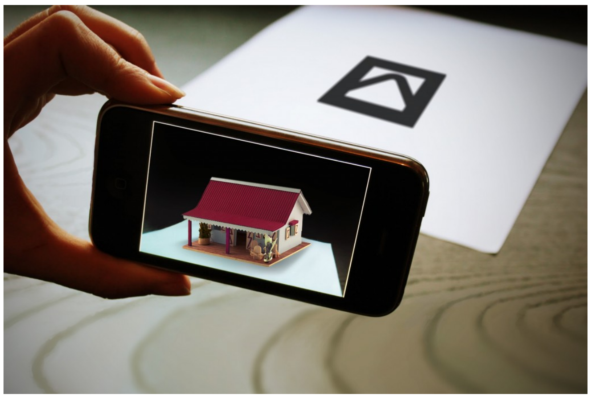


Figure 2.12.: Marker Based Tracking[12]

Markers are 2D physical images with clear patterns, easy to extract from the environment and recognizable by the Augmented Reality system. Markers are reference points in the real world, placed at specific locations. The marker is detected, then identified and used

[12] Source: `http://www.arlab.com/blog/markerless-augmented-reality/` (March 2014)

to determine the pose of the camera. To each marker, data or an interaction is associated. Thus basically augmenting a 3D object on top of the marker as shown in Figure 2.12.

Sanni Siltanen divides markers in two main categories: template markers and 2D barcode markers in [Sil12]. Marker identification for template markers is done by matching against sample markers. However, the 2D barcode markers are decoded, deciphering the encoded data within the marker.

As expected, introducing markers reduces the computation time to the time needed to detect and recognize the marker. It provides high accuracy with respect to the position of the camera. Thus the method is considered to be robust. One drawback is the costly maintenance of a functional marker over time, considering they must be kept for future use. Sometimes the markers can have a disruptive behavior in the real world, being difficult to place without obscuring something in the environment.

2.4.2. Markerless Tracking

Markerless tracking, as the name suggests, does not need markers. Instead, this method uses real objects in the real world environment as reference points. Any part of the real world can be targeted and thus tracked for superimposing the virtual data. The tracking process relies on natural features, so that by detecting the visual features in the images, the environment is learned from the movements between frames.

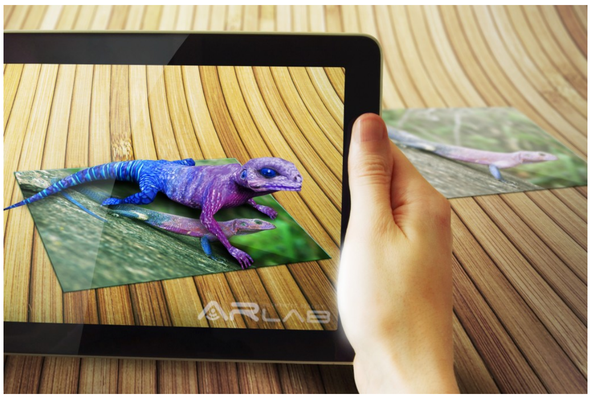


Figure 2.13.: Markerless Tracking[13]

Images of the 3D objects themselves can be used as targets. A popular choice in the advertisement industry as the client can see the product in 2D on paper or enhanced by Augmented Reality into the 3D version on the phone. By replacing the markers, some of the marker-based tracking limitations are overcome. For example, to some extent the occlusions of the image do not interfere with the rendering of the 3D virtual object as the image is tracked from edges, corners and even textures. By introducing virtual buttons, the markerless tracking method is more interactive.

[13] Source: `http://www.arlab.com/blog/markerless-augmented-reality/` (March 2014)

Markerless Augmented Reality uses the natural features detection technique. Natural features, like edges, corners or textures, are extracted from images or objects. These features are highly distinctive and invariant to transformations, such as translation, rotation and scale. They are however partially invariant to illumination and perspective changes. Classification and tracking rely on them.

As a first advantage, markerless methods recognize objects in the surrounding environment without the need of the invasive markers. These methods extract information and characteristics about the environment which may be useful again later. Moreover, markerless systems use specialized and robust trackers already available [Meg13]. On the other hand, in comparison with the marker-based systems, tracking and aligning of the real and virtual objects, also known as registration, are more complex.

2.4.3. GPS Tracking

Location based tracking methods are used to display relative information with respect to the current position. The most common sensor based tracking method is GPS tracking. The GPS examines the current location by the use of satellites. A number of other sensor based methods exist, including gyroscopes, cameras, hybrid vision, accelerometers and so on.

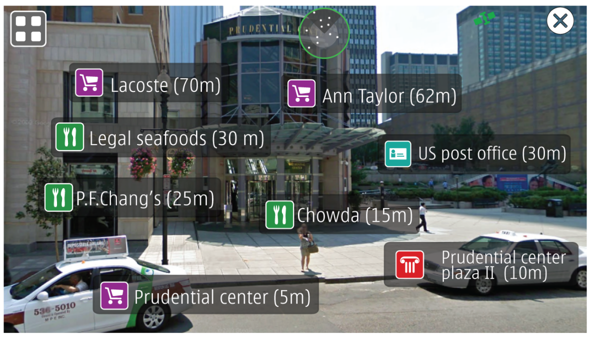


Figure 2.14.: Nokia's Live View Location based Augmented Reality Browser[14]

The process is similar to the marker-based tracking method, with the difference that the digital information is associated to a set of grid coordinates, instead of a marker. The grid coordinates show the current position of the user and based on this information locates interest points in his/her vicinity. The associated digital information is then displayed and it can range from distance to other interest locations or directions as shown in Figure 2.14 or from landmark information to user ratings.

Location tracking expands from indoor tracking systems, inside a room for example, to outdoor systems. The system requires a very exact position tracking. The GPS tracking has two major downfalls. First, where it is used: indoors versus outdoors. Indoors the GPS does

[14] Source: `http://www.sounds-like-me.com/news/wp-content/uploads/2011/08/Viewfinder.png` (March 2014)

not work as well as outdoors, because of a weak or simply no signal to the satellites. The outdoors, on the other hand, provide a better environment for the GPS given a clear sky. Tall buildings and trees could easily disturb or disrupt the connection to the satellites. This diversity leads to the second disadvantage, the accuracy which can vary from millimeters to several meters. As a result, it follows that GPS precision and update rate do not provide the means for high quality tracking.

3. Mobile Frameworks

Augmented Reality frameworks provide the necessary tools and libraries to develop Augmented Reality applications. The majority of these tools are packed as a SDK, short for Software Development Kit, used to integrate Augmented Reality in applications. The focus of this theses is on the Augmented Reality software used to develop Android apps.

3.1. Available Frameworks

Thirty five Augmented Reality frameworks for Android development have been found on the largest list of Augmented Reality SDKs [Dav12]. Most of them are sold to other companies or private developers in order to be used to create Augmented Reality experiences. Furthermore, the SDKs are bought under a commercial license which allows the customer to develop an Augmented Reality application and distribute it on Google Play.

Seven SDKs have been chosen for further evaluation considering several guidelines. Obviously, the framework must support markerless tracking. From the 35 frameworks, 28 support markerless tracking. An important criterion is the availability of the library. Several SDKs are available only for commercial use or just not available to the general public. Others are accessible just for a trial period. Some are depreciated, like in the case of Popcode, or even present language barriers as Koozyt, where information is provided only in Japanese.

Table 3.1 indicates the frameworks that are not evaluated in the current thesis, as well as the reasons for which they are excluded. The country column represents the place origin where the framework has been and still is being developed, and thus giving a general overview of how spread Augmented Reality development really is.

As it can be observed, the most common problem encountered as a developer is the unavailability of the SDK. In some cases the SDK was temporarily unavailable at the time of this inquiry, as it was for **ARMedia** and **Augmented Pixels**. However, in most cases, the developing companies could not provide an academic license for the purpose of the current research.

Aurasma is a self-service platform, easy to use for adding digital information on a target chosen by the user. Unfortunately, the upgrade to **Aurasma 2.0** and implicitly, the **Aurasma SDK**, implies a paid model license. **Qoncept** provides Augmented Reality technologies only for commercial purposes. **Xloudia** has expressed that for an interested party, a developer, a money transfer is required. And last but not least, **xpose** offers functionality integration for prices from €5000 for the development assistance and an annual fee of €3500 for a license, including 100 images.

	Framework	Country	Reason
1	ARmedia	Italy	SDK not available
2	Augmented Pixels	Ukraine	SDK not available
3	Cortexica	United Kingdom	SDK not available
4	DroidAR	Germany	SDK not available
5	ObviousEngine	United Kingdom	SDK not available
6	SSTT	Austria	SDK not available
7	Zenitum Feature Tracker	South Korea	SDK not available
8	Aurasma	United Kingdom	commercial use only
9	Qoncept	United Kingdom	commercial use only
10	Xloudia	Japan	commercial use only
11	xpose	Netherlands	commercial use only
12	IN2AR	Netherlands	development environment
13	Voxelogram	South Korea	development environment
14	YVision	Portugal	development environment
15	Popcode	United Kingdom	depreciated
16	Studierstube ES	Austria	depreciated
17	ALVAR	Finland	only for internal use & selected partners
18	Robocortex	France	only corporate business
19	Awila	United Kingdom	no SDK key
20	layar	Netherlands	30 days trial

Table 3.1.: The list of frameworks not evaluated

The development environment has also been an important factor in selecting the frameworks for evaluation. Considering Android technology, Java environment has been chosen for the development of the test app. **IN2AR** is used by mobile Flash developers and requires the FlashDevelop mobile development platform. **Voxelogram** has implemented FVVLib, "Free-Viewpoint Video" file format, and only supports Visual Studio 2000. **YVision** SDK is implemented in C# with many applications in Silverlight. **Wikitude** uses JavaScript for communicating with the native project, and moreover it is built almost entirely on web technologies.

A couple of frameworks are depreciated, which means development has stopped and no further support is provided. That is the case for **popcode** that stopped development in 2011 when the company changed its direction, producing a full end-to-end Augmented Reality solution without releasing a developer kit. However, **popcode SDK** is still available for download, but only supports the Visual Studio environment. A research team from the Graz University of Technology has stopped developing **Studierstube** in 2008 and moved on to newer projects. Nevertheless, the mobile **Studierstube ES** would not have been available as it is not open source.

From the emails exchange with the companies, it is revealed that **ALVAR** mobile SDK is only available for internal use and selected partners, and not available for general licensing. **Robocortex** has developed Rox Odometry and stated that the SDK is available only for corporate business.

Awila offers the SDK on a trial basis for which the developer must request an unique development key. **Layar** can be used for a maximum of 30 days trial period, after which it will become inactive. For this reason it is not included in the test app.

AndAR, an Augmented Reality framework developed by Tobias Domhan in 2010 [Dom10], has also been considered. Nevertheless, as AndAR is just a pure Java API to the ARToolKit framework and moreover, displays bad video rendering, it was taken off the evaluation list.

The seven frameworks chosen for evaluation are described in more detail in the remaining of this chapter. For a quick overview, Table 3.2 enumerates these frameworks.

	Framework	**Marker-based Tracking**	**Markerless Tracking**	**GPS Tracking**	**Availability**
1	ARLab	x	x	x	demo SDK
2	Arpa Solutions		x	x	demo SDK
3	ARToolKit	x	x		academic license
4	catchoom		x		demo SDK
5	D'Fusion	x	x	x	watermark
6	metaio	x	x	x	watermark
7	Vuforia	x	x		free

Table 3.2.: The list of evaluated frameworks

Identity	Determine if the software has its own domain name and a logo.
Country	The country where the software was and is still being developed.
Year	The year the company started the development.
Classification	Determine the tracking methods the framework supports: marker-based, markerless or GPS.
Features	Determine additional features that are supported, such as visual search and face tracking.
Platforms	Enumerate the supported platforms, from desktop (Windows, Linux, Max) to web and mobile environments (iOS, Android, Windows Phone, BlackBerry), including tablet (iPad etc).
Content creation	Specify if a studio or creator application is available for creating Augmented Reality target to digital content scenarios without needing prior programming knowledge.
Plug-ins	Determine if Unity is supported for gaming development.
Products	Enumerate the different SDKs releases.
Applications	Enumerate some applications built using the products.

Table 3.3.: Framework description

The frameworks are described following a common pattern. Table 3.3 identifies the software legend. A well-established software provides more confidence if it has its own domain and backs up the product with documentation. By supporting more than the markerless method, different tracking methods can be used in the same application to provide a more accurate management of the environment. An active community that supports the framework with discussions, bug fixes and new application ideas offers a nice advantage.

The information in Table 3.4 is used to describe the SDK's technical details. From a developer's point of view the minimum Android version on which the SDK can run represents vital information. Frameworks available on a fixed period trial basis have not been evaluated.

Version	The SDK's version tested in the current thesis.
Android	The minimum version of Android required for the SDK to work.
CPU	The minimum CPU requirement.
Memory	The minimum memory requirement.
GPU	The minimum graphic card requirement.
Source	The link to the source SDK.
Documentation	Determine if documentation is available for the software by providing the link.
Support	Determine if the software is supported by an active community by providing the link to the forum.
Licensing	Determine the conditions under which the framework can be used.
Sample projects	The list of projects that exemplify the use of the SDK.
Internet connection	Does the app need internet? Set to offline or online.

Table 3.4.: SDK details

3.2. ARLab

ARLab is an Augmented Reality laboratory which after years of research and development became a leading company in Augmented Reality and computer vision. The aim is to build the most advanced computer vision filters so that developers can easily build applications. The claim of **ARLab** is "Build your application in less than 10 minutes" [ARL14b]. Table 3.5 provides more information about the framework.

ARLab offers both Augmented Reality applications developed by in-house software experts and Augmented Reality technologies for independent development. Factoring in the growing number of developers using the **ARLab** technology, linkAR has been created to support the **ARLab** community for mobile developers. LinkAR provides access to the SDK and documentation while being a platform for sharing information amongst users, applications and ideas.

Identity	http://arlab.com/
Country	Spain
Year	2006
Classification	Markerless Marker-based GPS
Features	Visual search Face tracking
Platforms	Mobile (Android, iOS) Tablet
Content creation	-
Plug-ins	-
Products	Image Matching SDK[15] AR Browser SDK[16] 3D Engine SDK[17] Image Tracking SDK[18]
Applications	MuseARt[19] BikeInUmbria[20]

Table 3.5.: ARLab

ARLab delivers markerless tracking by performing a visual search in image pools of 50 to 60 images, with support for thousands of target images. This method applies to both the Image Matching SDK and the Image Tracking SDK. Image Tracking, as the name suggests, tracks an image after being detected by returning information about its current position. 2D images and videos are superimposed on the target image. The Image Tracking SDK is available only for iOS development [ARL14d].

Location based tracking can be integrated using the AR Browser product offered by **ARLab**. Both Android and iOS systems are supported for including the geolocation view offered by the AR Browser [ARL14a]. The 3D engine SDK is used to create complete 3D and 2D animation applications.

[15] `http://www.arlab.com/imagematching` (March 2014)
[16] `http://www.arlab.com/arbrowser` (March 2014)
[17] `http://www.arlab.com/3drendering` (March 2014)
[18] `http://www.arlab.com/imagetracking` (March 2014)
[19] `http://www.arlab.com/blog/category/use-cases/` (March 2014)
[20] `http://www.arlab.com/blog/category/featured-apps/` (March 2014)

The SDK

Image Matching SDK is a real time image recognition engine SDK that can match thousands of images saved locally on the smartphone [ARL14c]. The SDK works offline and guarantees fast matching by providing results in milliseconds. The licensing falls into the Demo License category. Table 3.6 provides documentation and forum links.

Version	beta 12.08.03
Android	2.3.3+
CPU	-
Memory	-
GPU	-
Source	https://github.com/arlaboratory/demos/tree/master/Android/arlab_imm_and_beta120803
Documentation	http://developers.arlab.com/doc
Support	http://developers.arlab.com/questions
Licensing	ARLab Beta License
Sample projects	1
Internet connection	Offline

Table 3.6.: Image Matching SDK

The SDK contains sample projects which can be downloaded and tested for free. HelloImageMatching project demonstrates matching a target image from the local memory of the smartphone in a matter of milliseconds. When the image is recognized, a text is superimposed at the bottom of the display screen. The text is predefined for each image and clarifies the content of the detected image.

Features

As specified in the product description [ARL14c], the features of the Image Matching SDK are:

- Performs real time matching.
- Target images can be added and removed from the image pool during execution time.
- More than one target image at a time can be matched.
- Target images can be loaded locally from the mobile resources or providing an url.
- The target images can be customized, the developer can upload his own images.
- Supports QR code detection, can match up to four codes in the same time.
- Offline training is not necessary.
- Guarantees high accuracy matching rate.

- Provides custom cropping by selecting the frame to be matched.
- Defines the built-in camera view.
- Supports processing of a single image or a stream of images.
- Takes advantage of hardware acceleration.

Integration and Implementation

The library is called EADMatching.jar and the native libraries package includes libEADMatching.so, libEADMatchingNeon.so, libEADUtilities.so and libEADInterface.so. More details can be found in Chapter 5.

3.3. Arpa Solutions

Arpa Solutions is an award winning company in the field of Augmented Reality. The projects have a large range of applications, from markets such as marketing and advertising, e-commerce and museums to aeronautic manufacturing for plane programs. Over the years **Arpa** Solutions has sold more than 200 Augmented Reality projects. In addition to being a high-tech company, **Arpa** Solutions is also an engineering company, having developed more than 20 research and development projects [ARP13e]. More products are listed in Table 3.7.

The **ARPA** GLASS SDK has been released for developing Augmented Reality apps for the Google Glass [ARP13a]. The SDK supports both 2D image and text recognition as well as 3D object and face detection. Additionally, more than one target at a time can be detected and tracked. GPS tracking is also included in the **ARPA** GLASS SDK.

ARPA Plugin Unity is a plug-in for developing Augmented Reality games for desktop (Windows and Mac), mobile (Android and iOS), tablets (iPad) and wearables (Google Glass) [ARP13c]. **Arpa** Solutions also offers a modular platform that can be integrated with other product management and enterprise management software. **ARPA** Industry provides customized Augmented Reality solutions for large industrial applications [ARP13b].

Identity	http://arpa-solutions.net/en
Country	Spain
Year	2006
Classification	Markerless GPS
Features	Face tracking
Platforms	Mobile (Android, iOS, Windows Phone) Tablet (iPad) Wearables (Google Glass) Desktop (Winfows, Mac)
Content creation	-
Plug-ins	Unity Pro Unity Free 4
Products	ARPA SDK 2.2[21] ARPA GLASS SDK[22] ARPA Plugin Unity 2.0[23] ARPA Industry[24]
Applications	ARMirror[25] Augmented CES Las Vegas[26] Augmented Furniture[27]

Table 3.7.: ARPA

The SDK

ARPA SDK for Android automatically detects images as natural markers in real time video streams and superimposes multimedia data over them. In addition to the commercial license, **Arpa** offers a Demo License. The **ARPA SDK** License gives permission to test the performance of the SDK for 30 seconds, after which the app stops detecting and rendering the 3D object. Furthermore, the **Arpa** Solutions watermark is displayed on the screen. Table 3.8 provides more details about the SDK.

[21] http://arpa-solutions.net/en/ARPA_SDK (March 2014)
[22] http://arpa-solutions.net/en/ARPA_GLASS (March 2014)
[23] http://arpa-solutions.net/en/ARPA_Plugin_Unity (March 2014)
[24] http://arpa-solutions.net/en/ARPA_Industry (March 2014)
[25] https://itunes.apple.com/us/app/armirror-augmented-reality/id463774949?mt=8 (March 2014)
[26] https://play.google.com/store/apps/details?id=net.arpasolutions.libarpanftandroid&hl=es_419 (March 2014)
[27] https://play.google.com/store/apps/details?id=com.interactiveinteriordesign.augmentedfurniture&hl=en (March 2014)

Version	ARPANFTAndroid SDK 2.2
Android	4.0+
CPU	1 GHz dual-core CPU
Memory	1 Gb RAM
GPU	Quad-core GPU
Source	http://www.arpa-solutions.net/en/Access
Documentation	http://www.arpa-solutions.net/Developer_Guide/Android/Index.html
Support	http://www.arpa-solutions.net/forum
Licensing	ARPA SDK License
Sample projects	3
Internet connection	Offline

Table 3.8.: ARPA SDK

The SDK is released together with three sample projects. ARPAImage exemplifies the detection of the natural features given a target image and overlays the **Arpa** logo on top of the image. ARPAMultiMarker demonstrates the detection of more than one marker at a time and the superimposes a 3D object on each marker, ending up with a compact 3D model. Overlaying of a single 3D object on top of a target image is shown in the ARPAPyramid project.

Features

The trial SDK has most of the features provided by the SDK Pro package, without the six months support. These features are [ARP13d]:

- Recognition, detection and tracking of 2D images and 3D objects.
- Supports 3D face tracking.
- Possibility of detecting and tracking multiple images at the same time.
- Rendering in real time.
- Support for various multimedia content including audio, video, text, 2D image and animated 3D object.
- Switch between video-live and photo mode.
- It is also available for iOS and Windows Phone development.

Integration and Implementation

The library is called libarpanftandroid.jar and the native libraries package includes libarpanft.so, libcrypto.so, libiconv.so, libicuuc.so, libObjectRenderer.so, libObjects.so, libssl.so and libxml2.so. More details can be found in Chapter 5.

3.4. ARToolKit

ARToolworks is the home of the **ARToolKit** product family [ART14]. ARToolworks provides tools for developing Augmented Reality applications for desktop, web, mobile and app plug-ins. Their most popular product is the **ARToolKit** Professional. **ARToolKit** is being widely used for the development of other Augmented Reality software products. Along with **ARToolKit** Professional, ARToolWorks released a number of other products, enumerated in Table 3.9.

Identity	http://www.artoolworks.com/ AR TOOLWORKS INC
Country	USA
Year	2001
Classification	Markerless Marker-based
Features	-
Platforms	Mobile (Android, iOS) Tablet Web Desktop (Windows, Mac, Linux)
Content creation	-
Plug-ins	Unity Pro Flash Silverlight
Products	ARToolKit[28] NyARToolKit[29] FLARToolKit[30] FLARManager[31] SLARToolKit[32] AndAR[33] osgART[34]
Applications	Clash of the Titans[35] Softbank Hawks[36] Living Sasquatch[37] BuildAR[38]

Table 3.9.: ARToolWorks

[28] http://www.artoolworks.com/products/mobile/artoolkit-for-android/ (March 2014)
[29] http://www.artoolworks.com/products/desktop/nyartoolkit/ (March 2014)
[30] http://www.artoolworks.com/products/web/flartoolkit-2/ (March 2014)
[31] http://www.artoolworks.com/products/web/flarmanager/ (March 2014)
[32] http://www.artoolworks.com/products/web/slartoolkit/ (March 2014)
[33] http://www.artoolworks.com/products/mobile/andar/ (March 2014)
[34] http://www.artoolworks.com/products/desktop/osgart/ (March 2014)

ARToolKit has releases for both mobile environment and desktop development. **ARToolKit** for Desktop is used for creating augmented and virtual reality projects through marker-based and scalable natural feature tracking methods. NyARToolKit is an optimized **ARToolKit** library with multi-platform support for Java, C#, C++ and ActionScript3. OsgART is a C++ rendering library for Augmented Reality interaction and application development by combining the **ARToolKit** tracking technology with the graphic library OpenSceneGraph [ART13c].

For web development, ARToolWorks released three products. FLARToolKit is the Flash extension of the **ARToolKit** library, the flash-based Augmented Reality library used for creating quick web Augmented Reality applications. On the other hand, SLARToolKit is the Silverlight marker tracking library for developing real time Augmented Reality applications using Silverlight. The Flash Augmented Reality framework is released under FLARManager as a system for adding, updating and removing markers in a robust way. FLARManager manages multiple patterns [ART13d]. ARToolWorks also supports Augmented Reality games by integrating the **ARToolKit** plug-in for Unity into the product family.

The SDK

ARToolWorks released **ARToolKit** for Android, and by doing so providing the **ARToolKit** Professional technology for developing powerful mobile Augmented Reality apps. ARToolWorks offers two licensing models, the Open Source and commercial licenses. More technical details are given in Table 3.10.

Version	4.6.9
Android	2.2+
CPU	-
Memory	-
GPU	-
Source	https://omega.artoolworks.com/dist/artoolkit4/release/4.6.9-Android/
Documentation	http://www.artoolworks.com/support/library/ARToolKit_for_Android
Support	http://www.artoolworks.com/community/forum/viewforum.php?f=26
Licensing	Academic License
Sample projects	9
Internet connection	Offline

Table 3.10.: ARToolKit for Android SDK

[35] `http://clash-of-the-titans.warnerbros.com/release-the-kraken/` (March 2014)
[36] `http://softbankhawks.co.jp/event/stamprally/` (March 2014)
[37] `http://www.livingsasquatch.com/` (March 2014)
[38] `http://www.buildar.co.nz/` (March 2014)

The SDK contains several development paths, both C/C++ for native coding and Java, to be used as preferred by the developer. In the release package are included nine sample projects. These projects demonstrate how to use the **ARToolKit** libraries and perform texture tracking. The rendering of OpenGL and use of the OpenSceneGraph framework for loading and rendering 3D model content are exemplified. The sample projects are divided into three categories [ART13b].

The Java-based examples are written all in Java code and use the ARBaseLib classes. ARSimple exemplifies the detection and recognition of an image target and the augmentation of a 3D cube over the target. ARSimpleInteraction extends the ARSimple example by shuffling between stopping or resuming the rotation of the 3D cube as a result to user interaction. The user taps the screen informing the renderer and additionally, the phone vibrates.

Two sample projects exemplify mixing Java and C/C++ code, meaning the projects are written in both Java and native code. As a consequence, both the ARBaseLib classes as well as the libARWrapper for the C/C++ API are needed. The ARSimpleNative provides the same 3D cube example with the difference that rendering happens in C and not in Java. ARSimpleNativeCars displays a 3D car model by the use of a native OBJ model loader.

The remaining five sample projects access the full native **ARToolKit** API directly as most of the code is written in C/C++. ARNative and nftSimple perform rendering with OpenGL. ARNativeOSG and nftBook load and render a 3D airplane with help from the OpenSceneGraph framework. Both natural feature tracking samples, nftSimple and nftBook, together with the ARMovie project exemplify texture tracking. ARMovie exemplifies the loading and rendering of a short 2D video.

Features

Some of the features are listed below [ART13a]:

- Tracking is performed at full camera frame rate, up to 30fps.
- Uses the OpenSceneGraph 3.x featured renderer.
- Possibility to switch between front and rear camera.
- The camera image size is adaptable.
- The thresholding is performed automatically and adjustable filtering is supported.
- Supports multiple targets tracking.
- Includes calibration and texture training tools, thus new targets can be locally generated. Moreover, a complete range of developer utilities is available.
- Full source sample projects are provided, as well as most of the libraries code.

Integration and Implementation

The project library ARBaseLib must be included in the workspace and added to the Build Path of the new project. More details can be found in Chapter 5.

3.5. Catchoom

Catchoom developed a recognition algorithm which is at the core of the CraftAR Service SDK. The software is used to augment digital information on the detection of real life objects, such as printed media and packaging. The software can also be used solely for its image recognition function. Target images can be uploaded on the online **catchoom** panel [Cat14b] as well as their corresponding Augmented Reality digital content. More about **catchoom** can be read from Table 3.11.

Identity	http://catchoom.com/ catchoom
Country	Spain
Year	2011
Classification	Markerless
Features	Visual search
Platforms	Mobile (Android, iOS) Web
Content creation	Yes
Plug-ins	-
Products	CraftAR[39] CraftAR Mobile SDK[40] Cloud Image Recognition[41]
Applications	G-Live[42] FotoFarma[43] WShopping[44]

Table 3.11.: Catchoom

CraftAR, "The ultimate AR toolbox" as **catchoom** likes to call it [Cat14c] is a web application for creating customized Augmented Reality target images. In a matter a minutes, the user

[39] http://catchoom.com/product/craftar-augmented-reality-image-recognition/ (March 2014)
[40] http://catchoom.com/product/mobile-sdk/ (March 2014)
[41] http://catchoom.com/product/cloud-image-recognition-api/ (March 2014)
[42] http://catchoom.com/blog/powered-by-catchoom/augmented-reality-how-glamourous/ (March 2014)
[43] http://catchoom.com/wp-content/files/2014/02/Catchoom_Almirall-FotoFarma_CaseStudy.pdf (March 2014)
[44] http://catchoom.com/wp-content/files/2014/02/Catchoom_ShoppingNight_CaseStudy_HiRes.pdf (March 2014)

can build a target image by uploading an image from the local hard drive and adding digital content to it. Images and videos are supported as 2D content.

An image that is uploaded using CraftAR is being saved in the Cloud Image Recognition. The Cloud Image Recognition offers a very fast object recognition time with a very low false positive error margin. The image recognition technology behind is robust in terms of lighting conditions, sizable database search and partial occlusions.

The SDK

CraftAR Mobile SDK is available since March 2014, the second version of the software and supports Augmented Reality development. The first version focused on the Cloud Image Recognition performance. The SDK is available for download, link is provided in Table 3.12. The Demo License allows the SDK to be tested in a flashing environment, meaning that the screen continuously flashes a red background. The possibility of an Academic License has been proposed, for a discount of 60% for the first 3 months of use and 30% for the next 6 months.

Version	v2
Android	4.0+
CPU	-
Memory	-
GPU	-
Source	http://catchoom.com/product/mobile-sdk/#download-mobile-sdk
Documentation	http://catchoom.com/documentation/sdk/android/
Support	online support team
Licensing	Catchoom CraftAR Mobile SDK License
Sample projects	1
Internet connection	Optional

Table 3.12.: CraftAR Mobile SDK

A sample project can be downloaded from GitHub, https://github.com/Catchoom/catchoom-example-android, under Open Source license. The project offers three examples: how to add Augmented Reality programatically, how to add Augmented Reality using the CraftAR service or how to use just the recognition algorithm. Together with the source code, complete SDK documentation is provided. The SDK library must be manually inserted in the sample project.

Features

Among the features offered by the CraftAR Mobile SDK [Cat14a] are the following:

- The possibility to connect to CraftAR Image Recognition API.
- The reliable tracking of real objects.
- The recognition of CD/DVD covers, newspapers and magazines, logos and brands, posters, packages or monuments and places.
- The possibility of rendering online digital content.
- The possibility of customizing by programmatically extending classes.

Integration and Implementation

Three libraries must be added to the application, android-tracking-sdk.jar, ofandroidlib.jar and sanselan-0.97-incubator. The native libraries package includes libCatchoomSDK.so, libCatchoomSDK_neon.so, libneondetection.so and libtrack.so. More details can be found in Chapter 5.

3.6. D'Fusion

Total Immersion is one of the main providers of Augmented Reality software. The envisioned future of the company has been reached by the development of Augmented Reality applications that can fit in a pocket [Tot14c]. What started with a fitness simulation program in 1998, it developed into the original creation of Augmented Reality software in 2004. **D'Fusion** is the fusion of real and virtual worlds. Over the years, many products have been developed, as it can be seen in Table 3.13.

Total Immersion has developed two main products: **D'Fusion** and TryLive. The **D'Fusion** Suite is a commercial package and includes **D'Fusion** @Home, **D'Fusion** Mobile, **D'Fusion** Pro and **D'Fusion** Studio [Tot14b]. **D'Fusion** @Home offers Augmented Reality solutions on the web or right on a CD/DVD. A professional product for capable hardware demands, like multiple HD cameras and multi-core CPUs, is released under **D'Fusion** Pro. **D'Fusion** Studio is a free platform for creating 3D scenarios to be integrated into Augmented Reality applications.

D'Fusion offers an Adobe Flash SDK which can bring the Augmented Reality experience to the web. The SDK is a recognition and tracking library using the Total Immersion technology which additionally integrates Adobe development tools. **D'Fusion** for Adobe Flash Player needs Adobe Flash 10.x or higher to run and provides company brand exposure thus marketing brand awareness [Tot13].

TotalImmersion supports the retail and e-commerce industries by introducing TryLive. TryLive is a virtual try-on solution for online shoppers ready to visualize themselves in a virtual dressing room. Moreover, TryLive offers the possibility to virtually fit furniture and other products right in the customer's home. TryLive Eyewear allows online customers to test and buy different eyewear products without going to the store. TryLive has its own dedicated website, http://www.trylive.com/.

Identity	http://www.t-immersion.com/ Total Immersion™
Country	France
Year	1999
Classification	Markerless Marker-based GPS
Features	Visual search Face tracking
Platforms	Mobile (Android, iOS) Tablet (iPad) Web Desktop (Windows, Mac, Linux)
Content creation	Yes
Plug-ins	Adobe Flash
Products	D'Fusion Suite[45] D'Fusion for Adobe Flash[46] TryLive[47] TryLive Eyewear[48]
Applications	Magic Mirror[49] Baseball Cards[50] Generator Rex[51] Opel Astra Sports Tourer[52] Hallmark Cards[53] Dior[54] TryLive[55]

Table 3.13.: Total Immersion

[45] `http://www.t-immersion.com/products/dfusion-suite` (March 2014)
[46] `http://www.t-immersion.com/product/dfusion-adobe-flash` (March 2014)
[47] `http://www.t-immersion.com/trylive/trylive%E2%84%A2-augmented-reality-fashion` (March 2014)
[48] `http://www.t-immersion.com/trylive/trylive-eyewear` (March 2014)
[49] `http://www.t-immersion.com/project-gallery/endless-possibilities-augmented-reality-ipad2` (March 2014)
[50] `http://www.t-immersion.com/project-gallery/topps-3d-hit-makes-baseball-cards-relevant-again` (March 2014)

The SDK

D'Fusion Mobile SDK is backed up by substantial documentation and an active community of developers. The sample projects included in the downloaded package are meant to help developing a new Augmented Reality mobile app along with understanding the available features. The current SDK version is under Watermark License and the Total Immersion watermark is overlaid on the video stream. **D'Fusion** Mobile needs a dedicated GPU for a successful execution on an Augmented Reality application. More technical details are given in Table 3.14.

Version	3.26MR1
Android	2.1+
CPU	-
Memory	-
GPU	-
Source	https://community.t-immersion.com/samples/android
Documentation	https://community.t-immersion.com/sites/default/files/files/DFusion%20Mobile%20-%20User%20guide.pdf
Support	https://community.t-immersion.com/forums/dfusion-mobile
Licensing	Watermark License
Sample projects	8
Internet connection	Offline

Table 3.14.: D'Fusion Mobile SDK

Eight sample projects are included in the sample package. The simplest example is the SamplePlayer project which loads the scenario of a 3D yellow robot. The example supports user interaction such that by double tapping the robot, the animation starts and a mechanical sound can be heard. Full-screen video playback is exemplified in the SampleVideoFileAndStreaming. The video is either loaded locally or streamed form an internet address. This is the only sample project where the **D'Fusion** Mobile SDK is not integrated.

SampleViews project demonstrates the use of tabs. The application displays three tabs, one for the Augmented Reality view overlaying the 3D yellow robot, one for a web view searching for "yellow robot" and a third view showing an Augmented Reality Google Map view where

[51] `http://www.t-immersion.com/project-gallery/cartoon-network-incredible-launch-new-show` (March 2014)

[52] `http://www.t-immersion.com/project-gallery/opel-putting-ar-excitement-consumer%E2%80%99s-hands` (March 2014)

[53] `http://www.t-immersion.com/project-gallery/hallmark-venerable-company-opens-new-market` (March 2014)

[54] `http://www.t-immersion.com/project-gallery/dior-making-their-campaign-%E2%80%9Cpop%E2%80%9D` (March 2014)

[55] `http://www.trylive.com/customers/mister-spex` (March 2014)

the 3D robot is superimposed over France. The use of an activity for each tab leads to an unnecessary time to unload/reload the scenario. SampleViewFlipper illustrates the use of multiple views in one group without destroying the other views when one is displayed.

The phone's sensors are tested in the SampleInputs application. The choice in sensors is between touchscreen events, accelerometer, GPS and compass. The SampleRecordInputs saves the tested sensors values in xml files. For more testing scenarios, SampleTests covers 28 different tests, 3 of them being available only in the **D'Fusion** Studio Suite Pro package. Additional to the tests performed in SampleInputs, this sample project includes back and front camera testing, touchscreen testing, sound testing and face tracking testing with additional automatic detection of the face. Other tests exemplify the difference between high quality and low quality rendering, taking a snapshot of the virtual 3D object, scaling the object, rotating the screen and so on.

Features

The mobile SDK offers optimized tracking in addition to using a refined 3D rendering engine and GPS services. The SDK can be further characterized by [Tot14a]:

- Secure information, the 3D objects and videos are protected from pirating and data corruption by encryption and signature.
- The highest quality of tracking considering the effective tracking and optimized algorithms for the mobile environment.
- The possibility of acquiring extra information from the mobile sensors, the GPS, accelerometer, compass.
- Face tracking capabilities by automatic detection of the face.
- Locally storing up to 500 images to be recognized, cutting down the time to access the server for retrieving target images.
- A fast recognition of target images and 3D objects from video streaming, without the need of taking pictures.
- The detection of bar codes.
- Providing simple tracking scenarios, user interface and touchscreen interaction.
- The combination of location services and digital content.
- Supporting Open GLES2.
- Generating high quality video rendering.
- Splitting the work between the cores given a multicore system, thus speeding up tracking, recognition and rendering.
- Offering data recovery services.
- Displaying Google Maps view and video playback.

Integration and Implementation

The library is called dfusionmobilesdk.jar and the package includes the libtiAndroidAR2.so native code dynamic library as well as some optional resources for the application runtime. More details can be found in Chapter 5.

3.7. Metaio

Metaio started delivering Augmented Reality solutions in 2003 to the automated and automobile industries. Now they offer solutions in marketing, sales, entertainment, engineering, publishing, education, consulting, development and the list goes on [Met14d]. The first Augmented Reality application was built for fitting virtual furniture, an application released under the name KPS Click & Design in 2005. One year later the first Augmented Reality browser plug-in was released, followed by the Augmented Reality library for mobile development [Met14a]. More product releases are listed in Table 3.15.

metaio Creator is an easy to use tool for creating and deploying Augmented Reality scenarios without the need for previous programming knowledge [Met14f]. The scenarios are hosted in the **metaio** Cloud, a plug-in for managing apps and contents globally, from where they can then be published to either an existing app or the Junaio Augmented Reality browser. **metaio** Cloud is also accessible through the Junaio API [Met14e]. Junaio is a free, fast and easy to use Augmented Reality browser which offers 2D and 3D tracking, as well as location based services. It is supported by both Android and iOS systems [Met14c].

Metaio also commits to providing scalable solutions as the **metaio** CVS, specialized tools for industrial Augmented Reality such as the **metaio** Engineer and an Augmented Reality engine, AREngine, for improving hardware performance. For large scale Augmented Reality projects that require more than 150 target images, **metaio** released **metaio** CVS based on the Continuous Visual Search method. Up to 1 million detectable target images are uploaded to the servers, from which the target image identifier and tracking configuration are retrieved when the target image is matched against a sample image [Met14g].

Industrial custom projects can be developed using the **metaio** Engineer tool. Augmented Reality can offer the possibility to see and apply digital information in context with the actual situation by comparing it to the planned state, and so much more [Met14h]. AREngine, the Augmented Reality IP from **metaio**, computes the most demanding Augmented Reality processes on the dedicated hardware IP resulting in freeing the cores. Furthermore, power consumption is significantly reduced without affecting the Augmented Reality experience [Met14b].

Identity	http://www.metaio.com/
Country	Germany
Year	2003
Classification	Markerless Marker-based GPS
Features	Visual search Face tracking
Platforms	Mobile (Android, iOS) Web Desktop (Windows)
Content creation	Yes
Plug-ins	Unity 3D
Products	The Metaio SDK[56] The Metaio Creator[57] Metaio Cloud[58] Metaio CVS[59] Junaio[60] Metaio Engineer[61] The AREngine[62]
Applications	Atelier Pfister[63] deCIDER[64] duo schreib & spiel Interactive Catalogues[65] McMission[66] Project Q[67] AR Puzzle[68] TV Movie TrailerViewer[69] Galileo Smart Quiz[70]

Table 3.15.: Metaio

56 `http://www.metaio.com/products/sdk/` (March 2014)
57 `http://www.metaio.com/products/creator/` (March 2014)
58 `http://www.metaio.com/cloud/` (March 2014)
59 `http://www.metaio.com/visual-search/` (March 2014)
60 `http://www.metaio.com/junaio/` (March 2014)
61 `http://www.metaio.com/engineer/` (March 2014)
62 `http://www.metaio.com/products/arengine/overview/` (March 2014)
63 `http://www.metaio.com/customers/case-studies/atelier-pfister-plan-your-living-environment/` (March 2014)
64 `http://www.metaio.com/customers/case-studies/bulmers/` (March 2014)
65 `http://www.metaio.com/customers/case-studies/duo-schreib-spiel-interactive-retail-catalog/` (March 2014)
66 `http://www.metaio.com/customers/case-studies/mcdonalds-mcmission/` (March 2014)
67 `http://www.metaio.com/customers/case-studies/project-q/` (March 2014)
68 `http://www.metaio.com/customers/case-studies/ravensburger-ar-puzzle/` (March 2014)

The SDK

Metaio SDK is an award-winning software for its tracking algorithm and 3D rendering engine. The SDK supports iOS, Android and Windows app development. Moreover, it includes the 3D Unity plug-in. The **Metaio** SDK release contains both the mobile SDK and **Metaio** Cloud plug-in. Further technical details can be found in Table 3.16.

Version	5.3
Android	2.4+
CPU	ARMv7, minimum 800 MHz
Memory	32 MB RAM
GPU	Integrated GPU with full OpenGL ES 2.x support
Source	https://metaio.app.box.com/SDK-PC
Documentation	https://dev.metaio.com/sdk/getting-started/
Support	http://helpdesk.metaio.com/
Licensing	Watermark License
Sample projects	4
Internet connection	Optional

Table 3.16.: Metaio SDK

The SDK is released together with three sample applications. Moreover, one of the sample applications, the Example project exemplifies instant tracking, visual search, custom shading, location based Augmented Reality, QR code reader and more. The Template sample demonstrates user interaction by animating the 3D world object at the tap action upon the virtual object. Even a sample interactive furniture Augmented Reality application is provided. The package also contains an example for the cloud plug-in, an app that requires internet access.

Features

Metaio SDK comes with numerous benefits and some of them are listed below [Met12]:

- Can be deployed on iOS, Android, Windows and Unity platforms.
- Offers advanced tracking for 2D target images and 3D objects.
- Supports bar codes and QR codes scanning.
- Optimizes 3D graphical rendering.
- Enhances 3D visual effects.

[69] `http://www.metaio.com/customers/case-studies/tv-movie/` (March 2014)
[70] `http://www.metaio.com/customers/case-studies/prosieben-galileo-smart/` (March 2014)

- Supports location based tracking, providing that the device is equipped with GPS and compass.
- Provides SLAM instant tracking.
- Uses AREL scripting language for developing Augmented Reality apps that can be deployed to different platforms, thus the platform independent feature.
- Supports 3D markerless tracking based on CAD data.

Integration and Implementation

The SDK **metaio** library project must be included in the workspace. More details can be found in Chapter 5.

3.8. Vuforia

Vuforia is the Augmented Reality platform for mobiles of the Qualcomm developer network. Furthermore, **Vuforia** is the product of the research and development center located in Vienna, Austria [Qua14]. **Vuforia** brings unique and engaging experiences through creative 3D graphics, touch, video and audio features. High fidelity and rich interactivity is promised through fascinating effects [Vuf13a]. License agreement and applications are listed in Table 3.18.

The SDK

The **Vuforia** SDK is platform independent, supports iOS, Android and Unity 3D development. The main focus is on natural features detection, hence markerless tracking. The software is released under Open Source license. A cloud recognition service is included in the SDK. The minimum Android requirement on which the SDK can run is written in Table 3.17.

The SDK package contains 13 sample projects, including virtual buttons implementation, text recognition and multiple targets detection. A cloud recognition sample app is also delivered. In addition to the mentioned projects, two apps are included, "Books" and "Dominoes". "Books" exemplifies the use of the cloud recognition service by providing information about a book after recognizing its cover. "Dominoes" demonstrates touchscreen interaction by allowing the user to dynamically create a dominoes course [Vuf13c].

Version	2.8.7
Android	2.3+
CPU	ARMv6, minimum 500 MHz
Memory	-
GPU	-
Source	https://developer.vuforia.com/resources/sdk/android
Documentation	https://developer.vuforia.com/resources/dev-guide/getting-started-android-native-sdk
Support	https://developer.vuforia.com/forum
Licensing	Open Source License
Sample projects	15
Internet connection	Optional

Table 3.17.: Vuforia SDK

Features

Aside the robust solutions towards lightning conditions, target occlusions and strong angles, other features include [Vuf13b]:

- Text recognition, support for approximately 100000 English words.
- 2D target images and 3D simple objects recognition.
- Up to five simultaneous target images at a time.
- The possibility to define own target images.
- Local database storage up to 100 images per application.
- Up to 1000000 targets cloud recognition.
- Extended tracking by means of which the digital information continues to be displayed even when the target image is no longer visible.
- Virtual buttons accessible by touching the smartphone's screen, thus making use of the touch screen functionality.
- Playing videos on top of target images.
- Shaders for creating compelling effects.
- Optimized rendered graphics.
- Robust to low light and partial visibility.

Integration and Implementation

After downloading the **Vuforia** SDK and extracting its contents, the QCAR environment variable must be set in Eclipse. It is recommended to keep the directory structure the SDK provides thus allowing future upgrades to be independent of the application development. More details can be found in Chapter 5.

Identity	https://www.vuforia.com/ Qualcomm vuforia
Country	Austria
Year	2011
Classification	Markerless Marker-based
Features	-
Platforms	Mobile (Android, iOS) iOS
Content creation	-
Plug-ins	Unity Standard Unity Pro
Products	Vuforia[71]
Applications	Guinness World Records 2013[72] Horrible Hauntings[73] Hot Wheels Power Port[74] Toyota 86 AR[75] Anatomy 4D[76]

Table 3.18.: Vuforia

[71] `https://www.vuforia.com/platform` (March 2014)
[72] `https://www.vuforia.com/case-studies/guinness-world-records-2013` (March 2014)
[73] `https://www.vuforia.com/case-studies/horrible-hauntings` (March 2014)
[74] `https://www.vuforia.com/case-studies/hot-wheels-power-port` (March 2014)
[75] `https://www.vuforia.com/case-studies/toyota-86-ar` (March 2014)
[76] `https://www.vuforia.com/case-studies/anatomy-4d` (March 2014)

4. Evaluation Criteria

In order to evaluate the Augmented Reality frameworks, a number of scenarios are defined. A scenario is a collection of criteria that best characterizes the requirements of a specific context. Therefore, several criteria have been proposed. Some of the analyzed criteria represent tracking challenges that cause recognition problems and are listed under environmental criteria. Target specific criteria are considered to determine which target image characteristics are and which are not supported by the framework. Basic problems often found in Augmented Reality systems that must be overcome are categorized under performance criteria. The fourth criteria category is represented by optional supported features like face tracking and camera flash.

4.1. Criterion

An evaluation criterion is a constraint that affects the efficiency of an Augmented Reality system. Two types of criteria are discussed in the presented research: online and offline criteria. An online criterion can be actively tested. An offline criterion represents a framework specific feature that cannot be tested, rather it is determined if the framework supports it.

An active test records the time needed to detect and recognize the target image. From this point on this amount of time will be referred to as testing time. If the test fails, the testing time will be -1 seconds. The offline criteria is measured in yes/no answers. For example, it will answer the question "Does the framework support the front camera?".

Dominik Rockenschaub has conducted a similar research for marker based Augmented Reality systems in his master thesis entitled "Entwicklung und Anwendung einer systematischen Vorgehensweise zur Analyse Marker basierter Augmented Reality Frameworks fÃ¼r mobile EndgerÃ¤te" [Roc12]. He identified marker based criteria as well as environmental criteria. Dominik defines marker based criteria to be those criteria that can be configured and directly influence the marker. Such a criterion would be the marker pattern or the complexity to produce a marker. For the purpose of this thesis, size, contrast ratio and grayscale are categorized under target criteria.

Environmental criteria are defined as the criteria conditioned by the environment, such as light intensities, mirroring, deformation and change in perspective. Both categories that Dominik identified are a good start for the present thesis. Each category adds criteria specific to a markerless based tracking system such as the material of the target image under target criteria or background clutter under the environmental category.

In addition to the two categories, performance and usability criteria are defined. Each framework description makes a point to state that some basic graphical problem is resolved and the tracker is optimized. These basic issues are collected under performance criteria and it is determined if they were actually invalidated. Furthermore, special features are implemented within each SDK for presenting a more comprehensive product. Not all features are supported by every framework, therefore they are categorized under usability and determined which are endorsed. Each of the four criterion category is presented in more detail in the remaining chapter.

4.2. Environmental Criteria

Environmental criteria represent those constraints found in the immediate neighborhood that influence the recognition of the target image. These criteria are conditioned by the environment and determined by the surroundings. Table 4.1 lists the proposed environmental criteria, all online criteria.

Light intensity	Test different lightning conditions.
Viewpoint	Test strong angle views.
Visibility	Test how much of the target image area must be visible to still recognize it.
Noise	Test how much noise can the target image contain and still be recognized.
Background	Test in clutter background environments.
Distance	Test the minimum and maximum distance to the target image.

Table 4.1.: Environmental criteria

4.2.1. Light Intensity

Extreme light intensities cause issues that the Augmented Reality frameworks must be able to adapt to. A target image might not be recognized or even wrongfully recognized if for example, due to uneven lightning, shadows hide parts of the image. Lightning conditions are not constant, which means that problems arise when dealing with sudden light changes.

Several illumination conditions are tested to find out how tracking is influenced:

- Light intensity levels, from very bright to semi dark. (or dark and bright)
- Shadows, obscuring parts of the target in semi dark.
- Camera flash as a direct light on the target.
- Sudden change in lightning conditions.

- Mirroring, a phenomenon that happens when the target image is printed on a reflexive surface. Mirroring is influenced by external light.

All tests are online tests and a successful test records the testing time.

4.2.2. Viewpoint

Mobile systems expect an increased level of mobility, thus the perspective from which the target image is observed changes often. This criterion tests the framework's recognition capabilities under different angles. Figure 4.1 demonstrates the perspectives taken into consideration for testing.

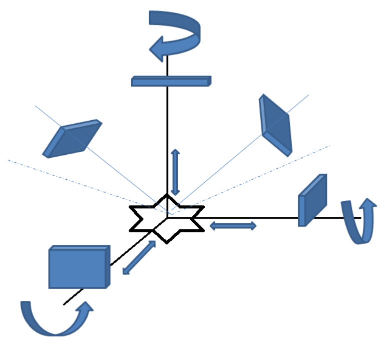


Figure 4.1.: The viewpoints considered for testing

Considering that at the basis of this representation is the target image, the testing device can move freely around it. Following the 6 degrees of freedom, commonly known as 6DOF, the device has the freedom to move into space taking into consideration translation and rotation[Lan13]. By translation it is understood that the device can be moved in the three perpendicular axis, forward/backward, left/right and up/down. Different perspectives are tested to conclude which viewpoints are detectable.

4.2.3. Visibility

Sometimes the target image is not completely visible as the line of sight can be obstructed or blocked by other real world objects. The visibility test will determine how much of the target's area must be visible in order for the framework to recognize it. The tests are performed starting from a 10% visible area and gradually increasing it in steps of 10%.

4.2.4. Noise

Augmented Reality is used in different industries and is natural that over time, the target image presents signs of deterioration. This noise can appear from overexposure, rupture, degrading material or spilled liquids. The tests are performed on different noise levels, starting with a 100% clean target image and decreasing the quality of the target at every step with 10%.

4.2.5. Distance

The distance from the camera to the target image represents another important factor in the recognition process. These tests determine how far or how close the user must be to the target in order for the Augmented Reality framework to recognize it and superimpose the virtual content. The target is printed at a default size and tests start from up close while increasing the distance by 10cm at every next step.

4.2.6. Background

The target image is usually found in a dynamic environment, such that the background is almost never static, making it harder for the framework to recognize the target. Furthermore, how much does a bright or colorful background influence the detection phase of the tracker? These tests are performed to decide in which backgrounds the framework performs well, thus being suitable for more dynamic scenarios.

4.3. Target Criteria

Not all images are suitable to be target images in the Augmented Reality recognition process. While an image can pass as a target image for one framework, it might not be supported by another one. The criteria tested to determine if a target image is accepted by a framework are found in Table 4.2.

Grayscale image	Test if grayscale and color images are both supported.
Contrast ratio	Test the image in different contrasts.
Size	Test the minimum and maximum size of the image at which it can still be recognized.
Aspect ratio	Test the image by changing its height.
Material	Test the image printed on different surfaces, such as photo paper, glass, cotton and so on.

Table 4.2.: Target criteria

4.3.1. Grayscale Image

The target images are printed both in color and grayscale. The tests determine which frameworks also support grayscale target images. The testing time for a color target image can be compared to the time needed to recognize a grayscale image and consequently it can be observed if tracking is affected.

4.3.2. Contrast Ratio

The target images are printed out with different contrast values. The tests determine how high/low the contrast ratio can reach in order for the target image to still be recognized. Examples of such images are given in Chapter 6.

4.3.3. Size

The target images are printed in different sizes, from 5cm height to a full scale A4 page. All tests are performed placing the target images at a fixed distance from the camera, starting from small sizes and progressively increasing the size. These tests determine the minimum and maximum size a target image can have such that they are still recognized by the framework.

4.3.4. Aspect Ratio

The target images are printed with a different height value as the original images. The new height value equals the width, thus printing square target images. The tests determine which frameworks can overcome this constraint. Furthermore, the target images are digitally modified and printed such that they are shrunk either on the vertical or the horizontal.

4.3.5. Material

The target images are printed on different materials. Tracking is influenced by the material on which a target image is printed as it might be more difficult to recognize the target from a glass than a sheet of paper. A number of different materials are tested, including:

- Paper, normal printing paper.
- Glossy photo paper, for the mirroring effect.
- Glass, for bottle surfaces.
- Plasticized paper for restaurant menus.
- Screen, for internet websites' target images.

4.4. Performance Criteria

When dealing with Augmented Reality systems, a number of problems can occur such as constant flickering or blurred view. Performance criteria are exactly those problems that every Augmented Reality system tries to overcome and the frameworks that achieved these are determined. The list of the achievement criteria is provided in Table 4.3. All performance criteria are offline criteria.

Registration	Determine if the real world and the virtual object are accurately aligned with respect to each other.
Flicker	Determine if the virtual content glimmers.
Motion blur	Determine if the virtual content is blurred.
Fast moves	Determine if the virtual content is lost when dealing with fast moves.
Occlusion	Determine if the virtual content is rightfully occluded when necessary.

Table 4.3.: Performance criteria

4.4.1. Registration

Registration is one of the basic problems Augmented Reality systems must deal with. For the virtual content to be integrated in the real scene and create the illusion that it is part of the real world, the real objects and the virtual one must be aligned with respect to each other [Azu97b]. In other words, the real and virtual coordinates must align.

4.4.2. Flicker

Flickering is an effect of misalignment between the real scene and the superimposed virtual object, already known as registration. Other identified causes are camera noise or poor light conditions. By suppressing this effect, the virtual content blends smoothly in the real world.

4.4.3. Motion Blur

Motion blur is an effect which originates with the camera, especially when dealing with mobile systems. As explained by Fisher, Bartz and Strasser [Pro06], fast movement in the real scene causes the light intensity to be temporarily integrated in the image sensor. As a result, the camera images of the scene are blurred.

4.4.4. Fast Moves

As previously stated, mobile systems allow a high degree of mobility. By fast moves it is understood that the target image is tracked while the device is in movement, as for example passing by the target while walking. It is determined if the tracking is lost or perseveres during fast moves.

4.4.5. Occlusion

Occlusion is a problem when an object from the real scene is supposed to be in front of the virtual one, thus occluding it. When occlusion is not correctly handled, the scene is spatially misinterpreted as the virtual object appears to be closer to the viewpoint than the real object. Occlusion is exemplified in Figure 4.2 by metaio.

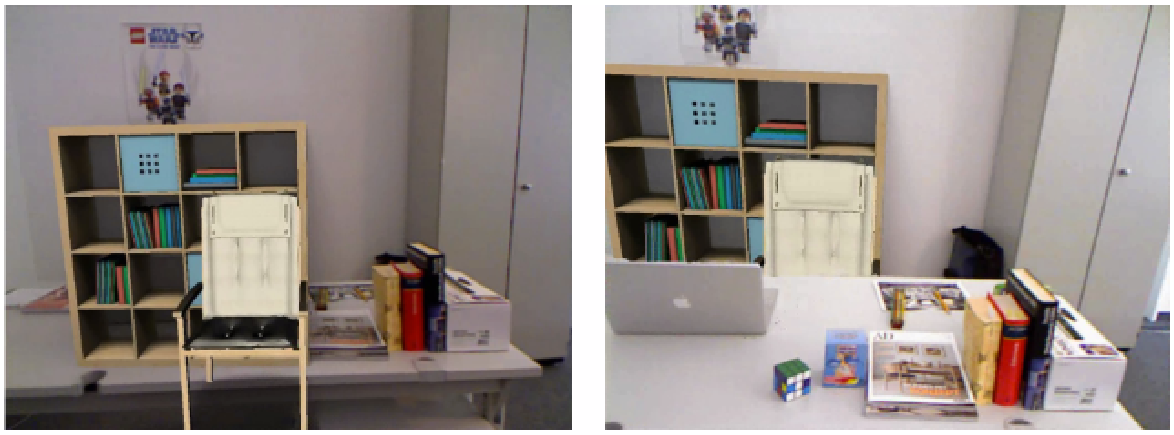


Figure 4.2.: Left image exemplifies the occlusion problem, both the chair and the bookcase are in front of the desk. Right image exemplifies occlusion handling.[77]

4.5. Usability Criteria

This criterion category illustrates optional features that a framework can provide and thus offer an advantage over the other frameworks. Most of these features refer to the usability of the software, nevertheless the last two criteria determine if the frameworks offer support for the embedded camera. The proposed features are listed in Table 4.4.

[77] Source:
`http://techcrunch.com/2011/10/27/metaio-adds-gravity-to-their-augmented-reality-platform/`
(April 2014)

Multi-targets	Test if more than one image target can be tracked in the same time.
Face tracking	Test if face tracking is supported.
Text detection	Test if text recognition is supported.
Extended tracking	Determine if the virtual content is still augmented when the target is no longer in the field of view.
Front camera	Determine if the front camera is supported.
Flash	Determine if camera flash is supported.

Table 4.4.: Usability criteria

4.5.1. Multi-targets

Multi-targets tracking helps stabilizing the pose estimation leading to a more precise tracking. For example, a second target image will help indicate the position of the virtual object relative to the real world. Not all frameworks support multi-targets tracking. The tests record the testing time for those frameworks that do support multiple targets.

4.5.2. Face Tracking

Face tracking is a feature that recognizes and tracks faces. By detecting face features like mouth, eyes, nose and their position, a virtual model can be superimposed on the face (Figure 4.3). This criterion is of interest in mCommerce applications like virtually trying on glasses before ordering them online.

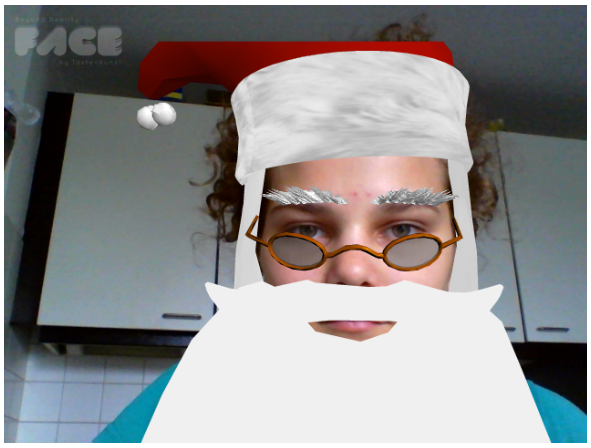


Figure 4.3.: Face tracking with Santa Claus model

4.5.3. Text Detection

Magazines, restaurant menus or even bus bench commercials use words to promote different products. By detecting and recognizing text, the real scene can be therefore further augmented. Frameworks that support this feature can easily be used to create translation apps.

4.5.4. Extended Tracking

Extended tracking allows the virtual content to persist in the real world whether the target is visible or not. After the target image is detected the first time, extended tracking is activated as seen in Figure 4.4. By pointing away the device, the virtual content remains superimposed in the same position with respect to the real world as illustrated in Figure 4.5. As not all frameworks support extended tracking, it is determined which ones favor it.

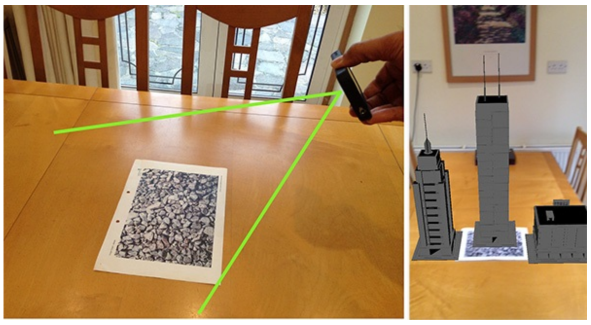


Figure 4.4.: Extended tracking while target image is visible[78]

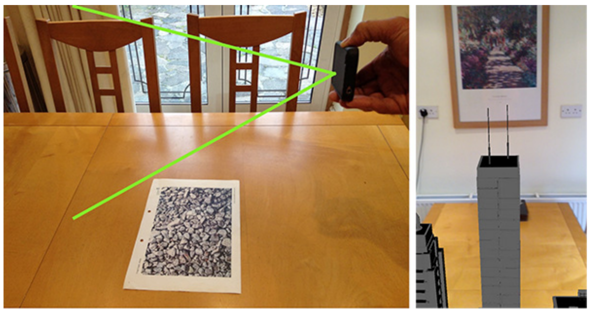


Figure 4.5.: Extended tracking while target image is not visible[78]

[78] Source: `https://developer.vuforia.com/resources/dev-guide/extended-tracking` (April 2014)

4.5.5. Front Camera

Most of the smartphones on the market are equipped with a rear camera. Nevertheless, some applications need a front camera in order to run properly. The front camera does not need to be actively tested, it is enough to determine if the framework supports using it.

4.5.6. Flash

Camera flash is actively tested under light conditions under the environmental criteria category. The usability criterion refers to the framework's ability to use a camera flash. The framework either supports it or not.

4.6. Scenarios

As already defined, a scenario is a collection of criteria given a specific context. A criterion can be found in more than one scenario, having a higher importance in ones than in others. In this way a criterion can have different weights in various scenarios depending on its relevance in each particular scenario. The weights are between 1 and 5, 1 being least important and 5 very important.

4.6.1. Interior Design App

Consider an indoors app for visualizing large objects like furniture, home furnishings, appliances. A target image would be placed in the spot where the real object would be and the virtual object is displayed on top of the target. A couple of constraints that must be considered include:

- how far or close the user can be from the target.
- having different viewpoints for a better outlook of the virtual object.
- occluding the virtual object by real objects if necessary.
- a correct alignment between the real scene and the virtual object.
- using more than one target for a better placement of the virtual object, at the right place and in the right scale.
- seeing the virtual object even when the target is not visible anymore.

The most important criteria are the viewpoint, as the user must be able to walk around the 3D object and see it from all angles, and registration, the alignment of the 3D object in the real scene so that for example a couch would not be seated on top of a real chair.

The next on the importance scale are the distance to the target image, occlusion and extended tracking such that a better perspective can be offered of how the virtual object fits in the scene. The further away the user is from the target, the better s/he can see how the 3D object blends in. Occlusion is meant to realistically show a virtual chair behind a real desk while extended tracking would allow the user to move around the desk and still see the chair even when the target is not visible anymore.

The size of the target image and the possibility of tracking more than one target are on the next level. The size is indirectly connected to the distance, bigger the size, more distance between the target and the user. Multiple targets are helpful to registration, thus the virtual object is shown in the right scale. These two criteria are seen as aid criteria to distance and registration.

Normally, as it is considered that this scenario takes place indoors, a uniform bright light source is implied. Nevertheless, shadows are bound to happen, which places the light intensity criterion on level 2 together with visibility. It is also assumed that the target will be placed in a visible position.

The weights of each criterion are given in Table 4.5 below.

Environmental criteria	
Light intensity	2
Viewpoint	5
Visibility	2
Distance	4
Target criteria	
Size	3
Performance criteria	
Registration	5
Occlusion	4
Usability criteria	
Multi-targets	3
Extended tracking	4

Table 4.5.: The list of evaluated criteria for Interior design scenario

A similar scenario can be seen in architecture. Imagine an app that would augmented a model of a building or a park or a bridge when a printed design plan is recognized. The same criteria would hold as well in this scenario.

4.6.2. Magazine App

Assume the scenario of a magazine app. An image in the magazine represents a target image for the magazine app. By hovering over with a smartphone the virtual content is revealed to the user. More constraints are considered together with:

- gray colored images.

- strong contrasts images.
- small size images.
- various patterns.
- images printed either on normal paper or glossy paper.
- recognize images given that they are not completely visible.

Most of the target criteria are considered in this scenario. On the first level of importance is the pattern criterion as the framework must be able to recognize all types of images, some that can be considered difficult to detect. On the same level is light intensity situated, considering that a direct light or a semidark environment will affect the detection of the target image.

Next come the grayscale images and visibility degrees. Not all images in a magazine are colored, even less in newspaper ads. Suppose someone is trying to sell a car and pays for an Augmented Reality ad in a black and white newspaper. An interested person, using a capable app would be able to see the model of the car. Another often occurrence, especially with brand new magazines, is when flipping pages, they are not completely visible as they are hidden by the backbone of the book. Therefore, images that are close to the spine of the book are covered and not 100% visible.

Magazines get old, paper tears, colors wash out, which leads to the next criterion, noise. On the same level are the printed material, light conditions and the support for multiple targets. Magazines can be printed both on normal paper or glossy paper. Glossy paper can be tricky as in combination with light it can cause the mirroring effect. Additionally, strong angles might interfere with detection. As for multiple targets, there could be more than one target image per page.

How small can a target image be and still be recognized? The size matters, as well as if an image's aspect ratio is modified to fit onto a page. In addition, text detection might be a useful feature if supported by the app.

The contrast ratio is not an often problem when dealing with magazine pictures, nevertheless it might happen that a batch of magazines are printed with a high contrast. Other random situations include a constant flickering, which would just be annoying, or a misalignment between the virtual content and the real scene, which does not lead to such big problems as in the previous example. Lastly, the distance between the user and the target should not be very small or recognition will prove to be difficult. Each criterion is associated a weight in Table 4.6.

Another scenario for the above criteria can be a restaurant menu. By hovering over the images in the menu, the food would be augmented before the user. The multi-targets criterion will ensure a real scale augmentation, thus knowing how much food really is on the plate.

Environmental criteria	
Light intensity	5
Viewpoint	3
Visibility	4
Noise	3
Distance	1
Target criteria	
Grayscale image	4
Contrast ratio	1
Size	2
Aspect ratio	2
Pattern	5
Material	3
Performance criteria	
Registration	1
Flicker	1
Usability criteria	
Multi targets	3
Text detection	2

Table 4.6.: The list of evaluated criteria for Magazine scenario

4.6.3. Bus Shelter App

Suppose a company comes up with a new marketing idea and uses Augmented Reality for posters creatively located on the side panels of bus shelters. A user can be either at the bus stop looking at the poster or notice it from inside a moving bus and try to discover its message. An outdoors scenario focuses on the environmental criteria along with:

- sudden light changes such as the sun hiding in the clouds.
- how far away the user can be from the target.
- degradation of the poster in time due to rain, wind, sun exposure and so on.
- a dynamic background such as people walking behind it or cars driving by.
- how much of the poster is visible from where the user sits.
- fast moves as the user passes by in the bus.
- once the poster is recognized, the virtual content is persistent even though the user is in a moving bus.

Light intensity, viewpoint and noise are the most important criteria for this scenario. Outdoors, the light intensities are not constant, they vary over time, sometimes sudden changes take place when the sun appears from behind the clouds. The user can be at different angles from the poster without the possibility to find a new position. This results into viewpoint handling.

As expected in an outside scenario, the target image degrades in time due to meteorological conditions, such as rain, wind, sun exposure, or even doodling.

In the event that a person sees the poster but cannot stop, either because s/he is in a hurry or already inside the bus just seconds before it leaves, fast moves and extended tracking are considered. Once activated, tracking should not be interrupted because of a sudden movement or the loss of the target image. On the same level are visibility and background. In an outdoor scenario, the background is rarely static, meaning that people walking by might represent a tracking challenge.

The level of medium importance lists material and distance. Even if the poster is printed on normal paper, being placed under a window screen can lead to the mirroring effect. The distance between the user and the poster must reach a maximum at which the poster is still recognizable.

The size of the poster comes up in the next level as it is related to distance. How big can the poster be and still be detected from a close distance? It is not desired for the user to have to cross the street in order to catch the whole image on the screen of his phone. What about the case when the poster does not fit the side panels and the aspect ratio is modified? How easy it is then for the target to be recognized?

The flash feature can be quite useful considering the dynamic light conditions, such as night times or cloudy days. Text detection is another useful feature when the poster supports augmented text.

The criteria for an outdoors scenario and their weights are listed in Table 4.7 below.

Environmental criteria	
Light intensity	5
Viewpoint	5
Visibility	4
Noise	5
Background	4
Distance	3
Target criteria	
Size	2
Aspect ratio	2
Material	3
Performance criteria	
Fast moves	4
Usability criteria	
Text detection	1
Extended tracking	4
Flash	1

Table 4.7.: The list of evaluated criteria for Bus Shelter scenario

Another example would be the treasure hunting game where posters can be found all over the city. The app will recognize the posters and provide clues to the next location such as videos, small animated 3D objects and so on. The criteria proposed fit to the treasure hunting scenario too.

4.6.4. Supermarket Promotions App

Some supermarkets today use Augmented Reality for promoting their products. Passing by a shelf with various products, the customer can point the camera to the boxes, select a product and initiate a game. For example burst some bubbles, such that when a predefined number of busted bubbles is reached, the customer wins a discount for the chosen product. A couple of features must be supported and some requirements must be met, such as:

- support for more than one target image per frame, such that the customer can choose which product he wants to play for.
- a blurred image would make it hard for the customer to play the game.
- the target image can be on a cereal box or a label on a wine bottle.
- flash might be needed if there is not enough light in the store for the app to recognize the target.
- the label can be torn or the image on the cereal box can be ripped.

The ability of recognizing more than one product at a time is of highest importance. For the supermarket workers, it would bring extra headache having to place product on separate shelves because they initiate promotional games. As important as multiple targets is, visibility is also mandatory. Products can be partly obscured by other products when stacked on shelves.

The viewpoint comes on the next level of importance. Consider the angle from which a product must be "scanned" when placed on a top shelf. The material also influences the fast recognition, considering that not only cereal boxes can be introduced in such a promotional campaign. And as the campaign uses games to draw costumers, games that involve interactivity with the superimposed virtual content, the motion blur effect must be avoided.

Trying to fit a target image on a wine bottle can lead to altering the aspect ratio of that target. Or even worse, reduce the image size in such a way that it cannot be detected anymore. Even more, some labels are printed black and white, for reducing manufacture costs. Does the framework still recognize the label?

Seeing as the focus of the campaign is on products, deterioration of boxes for example, on the transport way or while stacking them of shelves, is an often occurrence. Would it help if the customer would come closer to the product? Distance must also be considered as a factor.

Supermarkets are usually well lighted. One situation that can nevertheless can occur would be if the light bulb is somehow behind the customer casting a shadow on the shelves. The flash feature is thus also considered. Additionally, the text detection feature might be helpful for simple messages the supermarket manager left for the customers, to guide them to the augmented products.

The complete list of Supermarket Promotions criteria is given in Table 4.8 below.

Environmental criteria	
Light intensity	1
Viewpoint	4
Visibility	5
Noise	2
Distance	2
Target criteria	
Grayscale image	3
Size	3
Aspect ratio	3
Material	4
Performance criteria	
Motion blur	4
Usability criteria	
Multi-targets	5
Text detection	1
Flash	1

Table 4.8.: The list of evaluated criteria for Supermarket Promotions scenario

As an alternative to the game, a small video commercial could start playing at the recognition of a specific product. Thus, extended tracking would be a required feature. Furthermore, the motion blur would weigh less on the importance scale.

4.6.5. Tourist Translator App

An app exists today for translating text from English into Spanish and several other languages. Imagine you are visiting a foreign country, you do not know the language and need help to get around. This app can be used to translate signs, window ads, menus. An important feature that the Augmented Reality framework must support is text detection, as well as:

- the contrast between the text and the background where it is written.
- different materials on which the target is printed, either glossy paper, metal plates, glass and others.
- handle flickering as a constant flicker of the translated word would make it difficult to read.
- how far away the user can be from the target.
- contamination when dealing with outside signs.

The most important feature in the Tourist Translator scenario is the text detection criterion. A low contrast between the color of the word and the color of the background on which it is printed can increase the difficulty. As for the translated word, a constant flicker effect would make it really hard to read.

The light intensity presents recognition challenges both indoors and outdoors, either as sudden changes in illumination or shadows. Furthermore, the whole word needs to be visible in order to be translated. That means the viewpoint is on the same level of importance as visibility.

Level 3 lists the distance to the printed word and size of the said word. Contamination of the printed material as well as the material itself on which the word is printed are on level 2. The material can be from normal paper, glossy paper to metal and glass, or even fabric in the case of t-shirts which display messages. Street signs or window signs can be subject of deterioration (sun exposure, rain, erosion).

Some of the printed words are found in dynamic backgrounds, which should also be supported. The flash can sometimes be useful in poor light situations such as twilight or a dim romantic restaurant. Table 4.9 below lists the proposed criteria and their determined weights.

Environmental criteria	
Light intensity	4
Viewpoint	4
Visibility	4
Noise	2
Background	1
Distance	3
Target criteria	
Contrast ratio	5
Size	3
Material	2
Performance criteria	
Flicker	5
Usability criteria	
Text detection	5
Flash	1

Table 4.9.: The list of evaluated criteria for Tourist Translator scenario

Another suitable situation would be reading a book and learning a new language. For this case, the viewpoint would be lower ranked on the importance scale as normally the book is right in front of the user. Thus, the viewpoint is straight forward.

4.6.6. mCommerce app

A number of applications already exist for virtually trying on products such as glasses, hats and order them online. Take sunglasses for example. By using such an app, the customer can browse through the catalogue, choose a pair of glasses, switch between available colors and the sunglasses are projected on the image of the face looking back from the phone. The most important feature that must be supported is face tracking, together with:

- a static background for an easier face detection
- a constant light source or the color of the skin keeps changing
- fast moves lead to losing the tracking
- the possibility of switching to the front camera

Additional to face tracking, a constant light source is needed for a better tracking. By often changes in light intensity, the skin of the face displays different colors [JY95]. As a result it is harder to detect the face or can even lead to lose the tracking. Face tracking is performed by locating features like eyes, nose and mouth, in addition to the skin color. That said, it is important for all these features to be clearly visible. The viewpoint can be corrected until all features are easily detectable. A requirement of such an app is the use of the front camera of a smartphone.

Fast moves determine a large displacement of the face in a sequence of images, thus the search for a face from image to image becomes much more harder [JY95]. Another factor that influences motion in an image is the size of the face, therefore the distance criterion is considered. The flicker effect of the virtual glasses, in addition to being annoying, can disturb the customer from properly observing the product. Therefore, the customer is most likely to stop using the app and cancel his/her order.

By using such an app, it is assumed that the customer finds himself at home as to avoid going to the store personally. With that in mind, it is also expected that the background is a static one which will hardly interfere with the performance of the framework. The weights follow in the Table 4.10.

Environmental criteria	
Light intensity	5
Viewpoint	4
Visibility	4
Background	1
Distance	3
Performance criteria	
Flicker	2
Fast moves	3
Usability criteria	
Face tracking	5
Front camera	4

Table 4.10.: The list of evaluated criteria for mCommerce scenario

5. Test App Implementation

This chapter is divided into three sections. An introduction into the development environment as well as an overview about the hardware capabilities of the testing device, the implementation of the test app and finally the integration of the selected frameworks. The use cases are enumerated and the database model described in section 5.2. The last section presents the integration steps for each framework and is intended for future Augmented Reality Android developers.

5.1. Development environment

Android is the most spread mobile platform on the market, an open source software for smartphones. Most smartphones have an Android operating system installed. Google is also a big provider of commercial and free apps and games. Furthermore, free tools are available for developing applications such that anyone can become a developer.

The majority of Android apps are implemented in Java using the Android Developer Tools, the ADT plugin for Eclipse. This app has been developed in Eclipse IDE provided with a built-in ADT. Mobile developers can take advantage of the vast documentation put at their disposal on the official site for Android developers, http://developer.android.com/ (May 2014).

A fewer number of apps have parts implemented in native code to possibly optimize performance or access C/C++ basic functionalities and libraries. For this purpose Android NDK (Native Development Kit) has been released. Android NDK offers the tools to build C/C++ code into libraries which in turn is loaded in Java over the JNI Java Native Interface. The native functions within the libraries are therefore mapped to Java methods, thus being able to be called.

The app has been installed and tested on a Samsung Nexus phone. The technical details are given in Table 5.1. For developing Augmented Reality mobile apps, the Android emulator is in most cases not compatible. Most of the time, the smartphone's SD card needs to be accessed and files copied on it. Emulator testing is also not advised when dealing with apps that use the camera. The camera's hardware capabilities are provided by a smartphone.

Model	Samsung Galaxy Nexus I9250[79]
Production year	2012
Sensors	Accelerometer Ambient light sensor Proximity sensor Digital compass Barometer Rotation sensor
Camera	Rear camera: 5 Megapixel camera Video 1080p@24fps Autofocus LED flash Front camera: 1.3 Megapixel camera Video 720p@30fps

Table 5.1.: Galaxy Nexus specifications

5.2. Implementation

The presented implementation is an app which integrates a number of Augmented Reality libraries for the purpose of testing them under different conditions.

5.2.1. Use cases

When the app starts, the Home view will be displayed on the main screen as seen in Figure 5.1. The app runs on three pages or screens. The left screen is known as the Test page. This screen displays the Camera views for the different frameworks in order to test them. On the right screen, also known as the Results page, are shown the Graphs views. Sliding left and right will switch between the three pages.

[79] Source: `http://www.gsmarena.com/samsung_galaxy_nexus_i9250-4219.php (May 2014)`

The application has three main views on the middle screen, the Main page:

- Frameworks view presenting the frameworks either alphabetically ordered or by score provided that a scenario is selected.
- Criteria view showing the criteria grouped in the four predefined main categories.
- Scenarios view revealing the scenarios as already defined in Chapter 4.

Additionally to the three main views, a Settings view is also implemented.

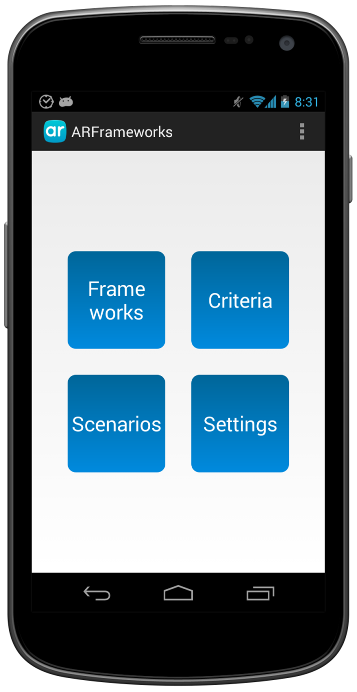


Figure 5.1.: Screenshot of the Home view

Frameworks View

The seven frameworks that are selected for evaluation are integrated in the Test app. For a specific scenario the frameworks are ordered by score, from highest to lowest. When a scenario is not selected the frameworks are ordered alphabetically by default as in Figure 5.2.

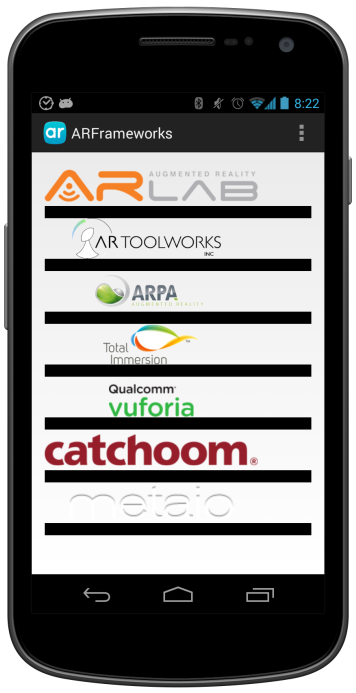


Figure 5.2.: Screenshot of the Frameworks view in default mode

The app does not allow other frameworks to be added to the list. This is a reasonable assumption, as each new framework would require backend support, integration in the project which involves coding. Furthermore, a framework cannot be deleted from the app considering that the library is already attached to the project. Nevertheless, it is possible to set a framework inactive and choose for inactive frameworks to be removed from the view and comparison graphs. Additional information is provided together with the framework's website.

Criteria View

The criteria are listed by category in alphabetical order as shown in Figure 5.3. By selecting a scenario, only the criteria important for that specific scenario are displayed, as well as their importance. At this point the criteria are ordered by weight.

CRUD operations are supported for the Criteria view. Therefore the app allows the addition of new criterion categories. Furthermore, new criteria can be added to a category. A criterion can be renamed or its description can be updated, as well as set offline. Clearly, a criterion can also be deleted. In the case of deletion of a criterion category, all the criteria found in that category are also deleted.

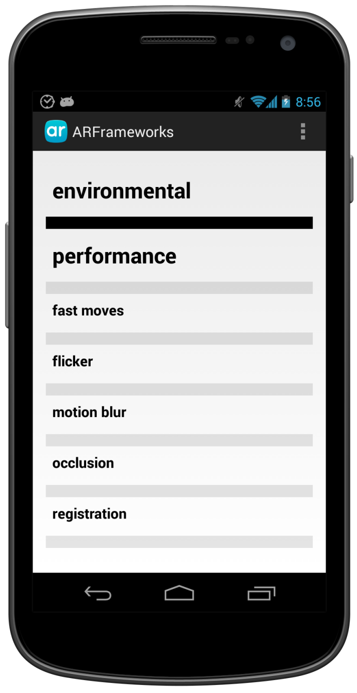


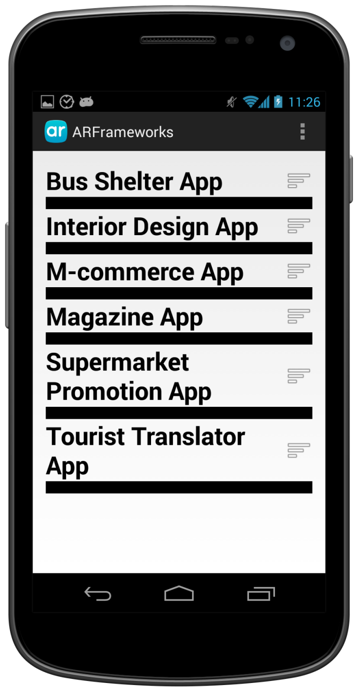
Figure 5.3.: Screenshot of the Criteria view in default mode

A criterion can, in its turn, have specifications. For example, the criterion visibility is found under the environmental criterion category. Testing visibility requires testing in different steps, starting with a 20% visible area and increasing at every step. This steps are saved as criteria specifications for the visibility criterion. Steps can be either added and removed from a criterion which can be seen as a global operation, or set to active/inactive inside a scenario.

The criteria's weight is added to the score of the framework then and only then when all its active specifics have tested successfully. It might happen that a scenario needs a framework that supports low visibility tracking, as it is the case of the Supermarket Promotions scenario. This would require the recognition of the target image from the lowest step, a 20% visibility to the 100%. On the other hand, visibility is not as important to the Interior design scenario, which is why the lowest step count starts with a higher visible area.

Scenarios View

When choosing the Scenarios view, the predefined six scenarios are listed (Figure 5.4). Each scenario has associated the list of evaluation criteria for a specific scenario. This list can be updated at any time, meaning that new criteria can be added, others can be removed. In addition, the importance of each criterion is changeable. It is important that they are easily accessible and changeable. By opening a scenario to access its contents, the Criteria view is actually displayed with the scenario filter activated.



Bus Shelter App
Interior Design App
M-commerce App
Magazine App
Supermarket Promotion App
Tourist Translator App

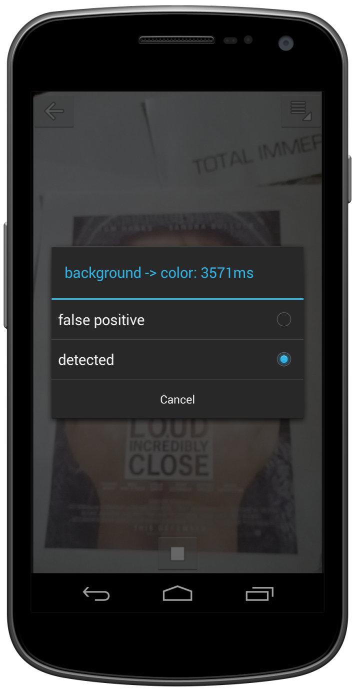
Figure 5.4.: Screenshot of the Scenarios view

The CRUD operations are also supported. New scenarios can be defined as well as updated. By adding a new scenario, it is implied that all frameworks will be tested on it. When a scenario is removed, the criteria weights of the corresponding evaluation criteria are also cleared. Additionally, the scores given to the frameworks for their achievements within the scenario are erased.

Test Page

A sliding action from the Main page will trigger a dialog. There are mainly two different cases, has a framework been selected or not? The only possibility for a framework to be chosen before sliding to the Test page is when the view in the Main page is the Frameworks view. A framework is selected by sliding it right, thus revealing the Test page. The question is if the test to be performed will be online and resulting in opening the framework in Camera view or offline which leads to selecting a criterion from the predefined offline criteria. The possibility exists that a scenario has already been chosen in the Frameworks view. When all criteria that define the scenario are online, the Camera view will open, respectively a form will be displayed for offline criteria. The question to choose between online testing and supported features arises when both criterion types define the scenario.

The Test page is accessible from the other two views as well with restrictions on the possible criteria to be evaluated. In both cases, a dialog will ask for a framework to be selected before starting testing. When sliding from the Scenarios view, only the criteria defining it are tested. It is allowed to test either one criterion or one or more criterion categories. By default, all criteria are testable when sliding from the Frameworks view.


background -> color: 3571ms

false positive ○

detected ●

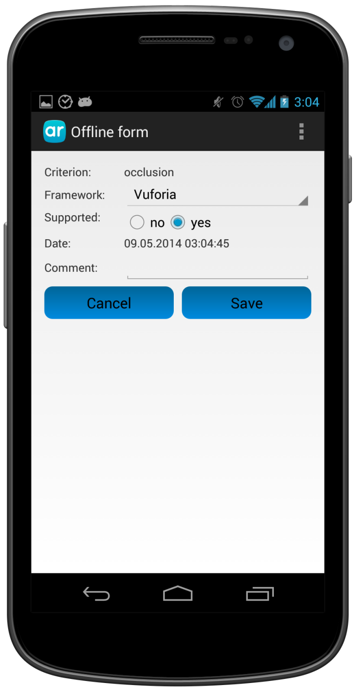
Figure 5.5.: Screenshot of the Camera view

The Settings view also supports sliding into the Test page. A test settings page is displayed offering different options that affect testing in general. Here it can be decided if a framework will stop tracking once the image target is recognized and the result saved in the database or tracking is resumed.

The Test page displays two views: Camera view and Offline view. The Camera view is used by each framework activity to activate the camera, initialize and start the tracker. Three buttons are overlaid on the Camera view as it can be seen in Figure 5.5:

- A Start button at the bottom of the page to control the beginning of the tracking action.
- A Criteria button at the top right corner of the view to switch between available criteria.
- A Return button to exit the Camera view.

The Offline view is a simple form to determine which features are supported. As it can be seen from Figure 5.6, both the framework and criterion can be changed, thus avoiding the constant back and forth between the Test page and the Main page. Only offline criteria are available at this point.



Criterion:	occlusion
Framework:	Vuforia
Supported:	no [✓ yes]
Date:	09.05.2014 03:04:45
Comment:	

Figure 5.6.: Screenshot of the Offline view

Results Page

The Result page is accessible from Frameworks view, Criteria view and Settings view. The results are displayed in a graph for online criteria while offline criteria are shown in tables. When sliding in from the Frameworks view, the Graph view is shown by default. As it is depicted in Figure 5.7, the graphs are really bar charts having on the vertical the testing times and on the horizontal the criteria categories. This is the default graph once a framework is chosen and it shows the testing times for all online criteria.

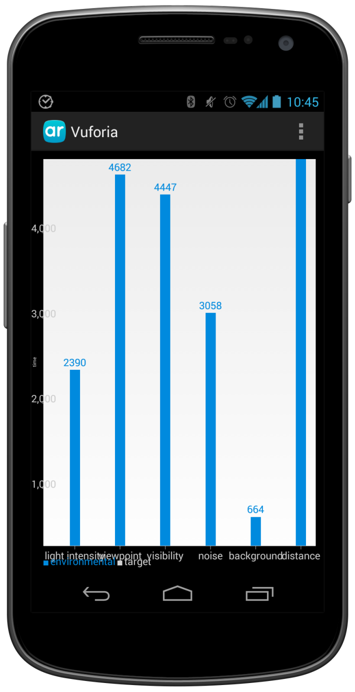


Figure 5.7.: Screenshot of the Graph view. Default graph for the Vuforia framework

The values shown in the graphs can be changed. In the case that a framework has been selected, the general case, a criterion category can be selected and thus show the test times for the specified criteria. As acceptable options, a single criterion can also be chosen and its criterion specifics detailed.

All frameworks are compared against a criterion when sliding in from Criteria view. The testing times are displayed as shown in Figure 5.8.

Considering that a criterion can be tested more than once on the same framework, a choice is given on which testing time to be displayed. The default choice is "Select the best timing", therefore the result with the quickest time is shown. Other options include:

- Show the last test, even if it failed.
- Average on the positive results, adding up all testing times of the successful tests and divide it by the number of successful tests.
- Maximum time value, which translates that for each failed test, a predefined MAX value is added as a testing time and the total amount of time is divided by the number of successful and failed tests.

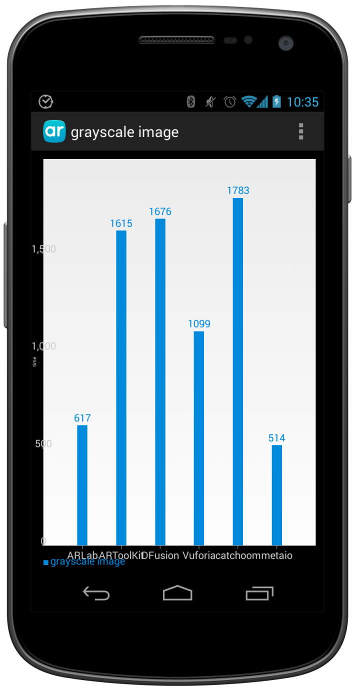


Figure 5.8.: Screenshot of the Graph view. Default graph for the grayscale criterion

The Settings view supports sliding into the Results page. A test settings page is shown providing options for displaying the results. Here it can be set if inactive frameworks should or should not still display their results together with the active frameworks in the Results view. As a specific view setting, in the case of a Graph view, the MAX value can be reset. The default value is 3000ms.

5.2.2. Database

Android uses the SQLite database. By the help of an SQLiteOpenHelper, quick SQL queries are performed on the database. An additional utility class contains more advanced queries, double joins and select in select.

For this app, six tables have been created. The primary three tables are:

- frameworks storing the eight frameworks
- criteria storing four predefined criterion categories, with a total of 27 criteria, not counting the specifics
- scenarios storing the six scenarios defined in this paper

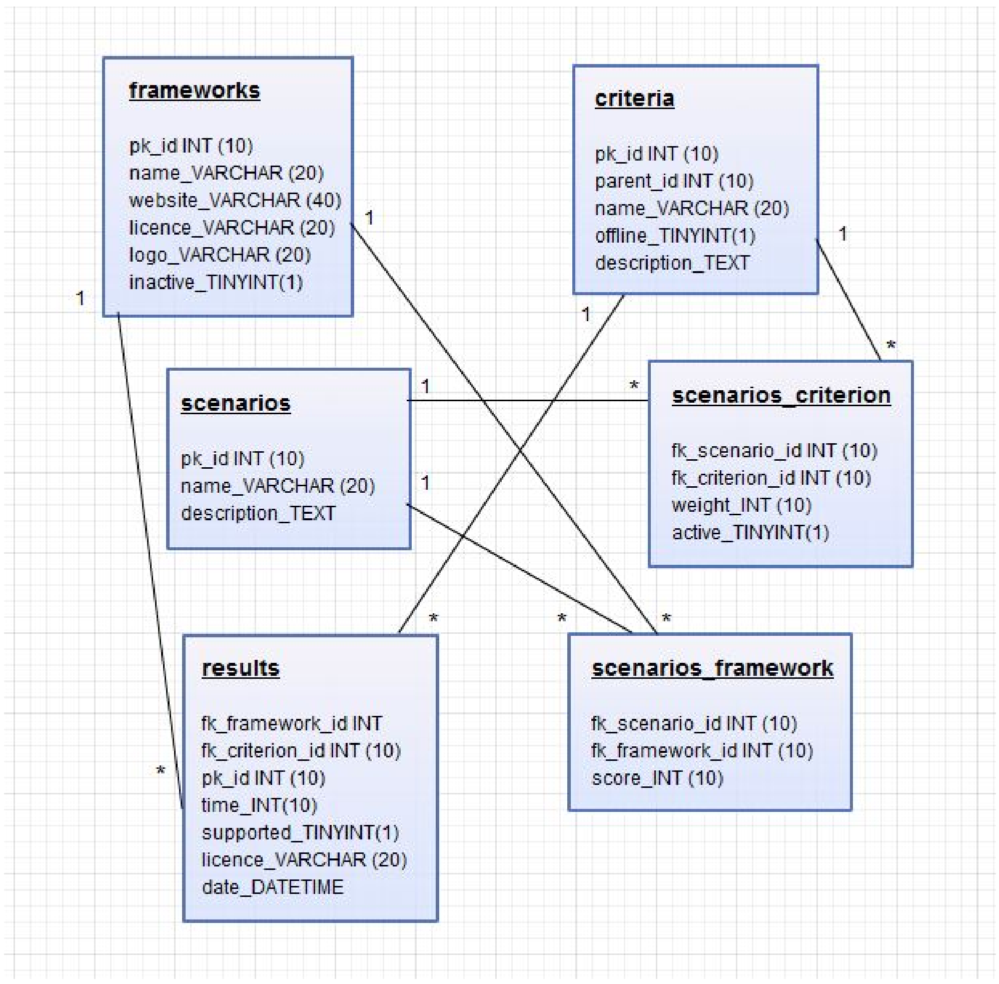


Figure 5.9.: The ARFrameworks.db database model

The results table stores all tests performed, online and offline. More than one time has a criterion been tested on a framework. An offline criterion is usually set just once. Additionally, the results table is accessed for inserting new test results, or retrieving information for creating graphs. As it can be noticed from the database model in Figure 5.9 the scenarios do not depend on the results table. Furthermore the results are used to update the frameworks' scores when a criterion is added or removed from a scenario. Lastly, the many-to-many relationship tables are scenario_criterion and scenario_framework.

A criterion insertion always assumes the new criterion is online and ready to be tested. The addition of a criterion specific to a criterion triggers the search for all scenarios that are defined by that criterion and subtract its weight from the frameworks' scores. The logic behind is that the new criterion specific did not have time to be tested. The criterion specific is stored as inactive by default. By globally setting the active/inactive property of a specific criterion, every instance of it in all scenarios is reset.

A test that fails is saved in the database with a testing time equal to -1, and additionally sets the supported field to false. When a new test succeeds, the database is queried to determine if a previous test has succeeded on the same constraint and chosen framework. If no other positive tests are found, meaning it is the first time to succeed, the score of the framework is updated in the case of criterion testing. Otherwise, if a criterion specific tests successfully, its active siblings within the scenario are found. When all specifics are supported, only then is the criterion successful.

A similar action takes place when a criterion specific is added to a scenario. By default it is set to inactive and once activated, if it has been tested and failed, its active siblings are looked up. In the case they all have positive results, the weight of the criterion is subtracted from all frameworks.

5.3. Frameworks Integration

Each of the eight evaluated frameworks follow different integration steps. As a common step, all frameworks must be either referenced to or added to the **libs** folder. Each framework recognizes a different target image. For this reason, the figures in this section show the Camera view for each framework presenting the image targets.

5.3.1. ARLab

ARLab is delivered in EADMatching.jar together with the 4 native source libEAD libraries. This library demonstrates image matching, therefore images are added to the assets folder. There is also the possibility to load an image from local resources, a given url or .dat files. The AndroidManifest.xml must be modified to set the package to "com.arlab" or the library will not work.

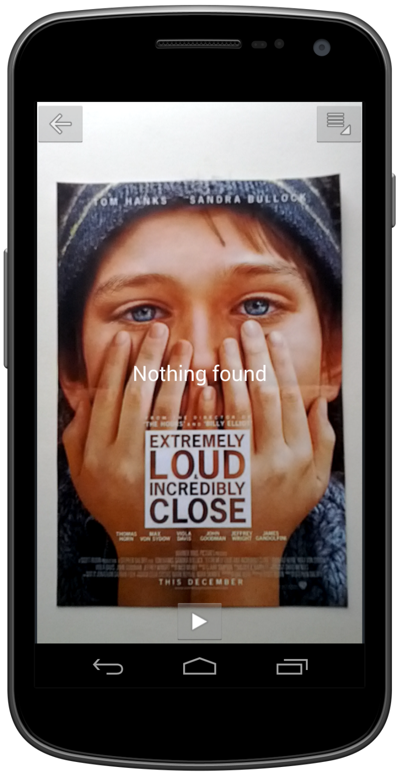


Figure 5.10.: Screenshot ARLab's Camera view

5.3.2. Arpa Solutions

ARPA Solutions is delivered in libarpanftandroid.jar together with 6 other native dependencies. A **raw** folder is added to the **res** folder containing the target image, the texture and the license file. The license file is received after applying for a test license, providing the name of the app and the main package. Without a valid license, an Incorrect license message is displayed informing that the license is either incorrect or cannot be found.

Additional, a **jni** folder needs to be added to the application folder. The Objects.cpp, Android.mk and Application.mk files from the sample project **jni** folder are to be copied here. The Objects.cpp file must be modified to link the name of each native method to the call from the Activity class. Android NDK is required to build libObjects.so. The ARPA Solution User Guide is a good place to start [Sol14].

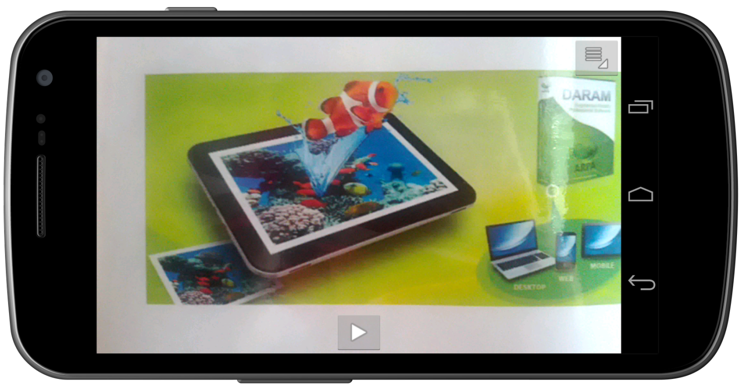


Figure 5.11.: Screenshot Arpa's Camera view

5.3.3. ARToolKit

ARToolKit needs the ARBaseLib included in the workspace and a reference to it set in the project.properties file. In addition to the Activity class, an ARToolKit renderer is used. As already noted, a number of sample projects are offered which can be used as example for setting up a project. The target image project is an NFT (Natural Feature Tracking) project and has native code that must be modified and build. Additionally, the ARToolKit folder which is included in the SDK must be copied one folder above the workspace or inside the workspace if the Android.mk Artoolkit directory path is reset.

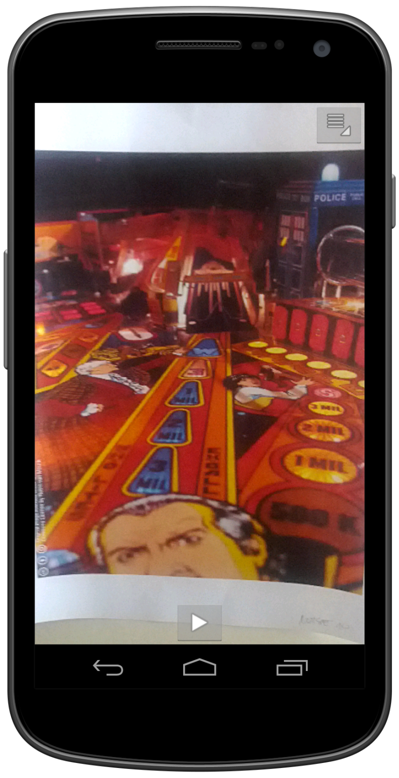


Figure 5.12.: Screenshot ARToolKit's Camera view

5.3.4. Catchoom

ARToolKit needs the ARBaseLib included in the workspace and a reference to it set in the project.properties file. In addition to the Activity class, an ARToolKit renderer is used. As already noted, a number of sample projects are offered which can be used as example for setting up a project. The target image project is an NFT (Natural Feature Tracking) project and holds native code that must be modified and build. Additionally, the ARToolKit folder which is included in the SDK must be copied into a folder above the workspace or inside the workspace if the Android.mk Artoolkit directory path is reset.

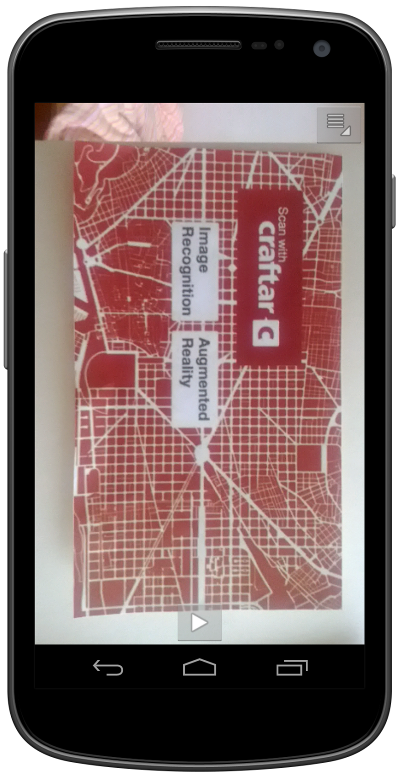


Figure 5.13.: Screenshot Catchoom's Camera view

5.3.5. D'Fusion

In order to use this the framework provided by Total Immersion, a D'Fusion project must be created or imported. A test key is generated when a D'Fusion is exported. This test key must be set to fit the requirements of the app to be developed. That means that the target is selected as "mobile" and the platform chosen is "android". Additionally, the "Application ID" must be specified as the package name of the Android app.

The project will be exported into an empty folder indicated by the user. The contents of this folder will be copied into the **assets** folder within the Android's project folders. If a **libs** folder does not already exist, it must be created and the dfusionmobilesdk.jar added to it. The additional libtiAndroidAR2.so native library must be added to both the **armeabi** and **armeabi-v7** folders.

As a last step, the dfusionmobilesdk.jar must be added to the build path of the project. More about the steps and configurations are detailed in the documentation which is delivered together with the SDK. The above mentioned steps can be found in the Annex.

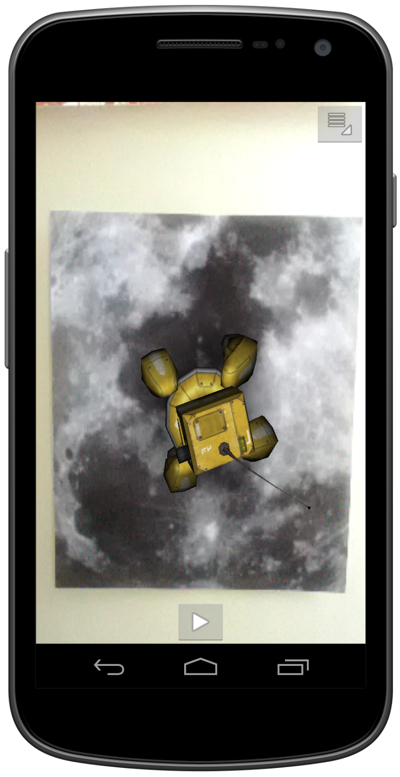


Figure 5.14.: Screenshot D'Fusion's Camera view

5.3.6. Metaio

Metaio needs the metaioSDK library included in the workspace and a reference set in the project.properties file. It is advised to check if any inconsistencies appear between the libraries the metaioSDK imports and the Android app. The android-support-v4.jar included in the metaioSDK is an old version of the support library.

The library requires a valid signature. For this purpose, an account must be made on the Metaio page, http://my.metaio.com/. Under "My apps" the app can be registered for free by inputting the app's name and application ID. A valid signature is then generated in a matter of minutes. This signature will then be added to the **strings.xml** resource file under the name "metaioSDKSignature".

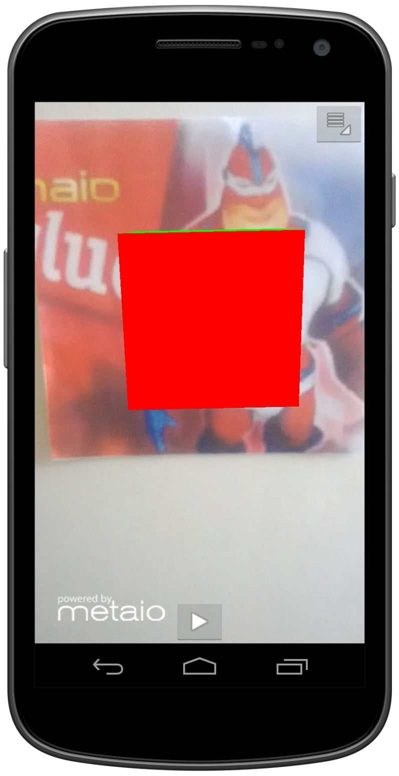


Figure 5.15.: Screenshot Metaio's Camera view

5.3.7. Vuforia

Vuforia framework also takes advantage of native code and requires the native C++ to be build. The same rule applies, the main .cpp must be changed such that the method names all link to the Activity class defined in Java code. In the present implementation, the ImageTargets.cpp file is the main file. The "." (dot) separating folders in the package name definition are replaced by _ (underscore) and each method name starts with Java_. As a result, com.arframeworks.frameworks.VuforiaFramework.getOpenGlEsVersionNative() Java definition will become Java_com_arframeworks_frameworks_VuforiaFramework_getOpenGlEsVersionNative.

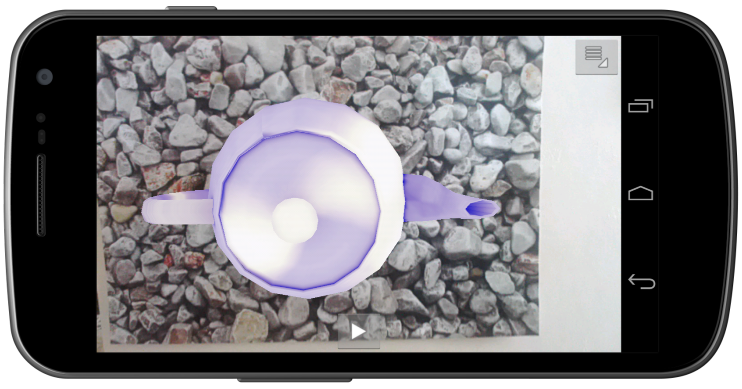


Figure 5.16.: Screenshot Vuforia's Camera view

6. Analysis of the Evaluation Results

This chapter presents the evaluation design as a test sequence and the conditions in which each criterion has been tested. The evaluation of each criterion is pointed out and the results per frameworks are evaluated. Section 3 presents the best fitted framework for each of the defined scenarios in Chapter 4 and how it was determined. Each framework displays its average testing time per constraint in Section 4. The chapter concludes with a summary of the results.

6.1. Evaluation Design

A detailed analysis is performed for each framework based on the predefined online criteria. Each framework uses a particular target image, each of them set to a default size. Some criteria are tested on special modified target images, nevertheless the default target image is used in most cases. For each criterion, different test conditions are defined.

Test Conditions

The default test parameters and conditions are given in Table 6.1. The first column represents the default distance between the test device and the target image. The second column determines the landscape/portrait format in which the framework is tested. Third column provides the target images default size and last column depicts the default target images used for each framework evaluation.

The default test parameters do not apply to the viewpoint and distance environmental criteria, as well as neither to the size target criterion. The viewpoint criterion requires both portrait and landscape formats. The distance criterion needs variable distances while the size criterion requires a variable size of the target image. For the purpose of testing some criteria within good conditions, Photoshop has been used to digitally alter the target images. More details about the test implementations per criterion are shown below in Table 6.2.

Evaluation Details

Six frameworks have been tested on 11 criteria. 57 tests have been executed per framework, summing up to a total of 627 tests for all frameworks and criteria. One test sequence needs minimum 30 seconds without the preparation time and followup operations. A total of 150 target images have been printed to cover the noise, aspect ratio, contrast ratio, color/grayscale,

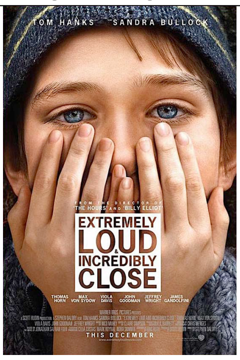
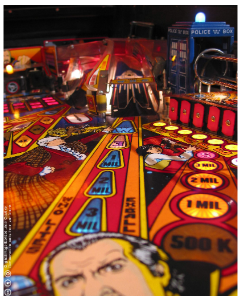
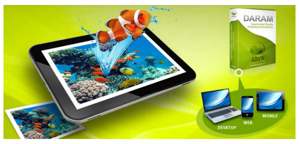
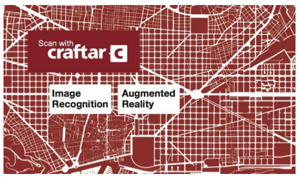
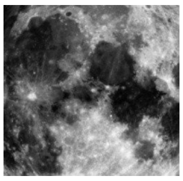
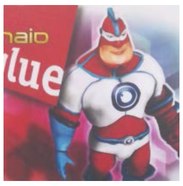
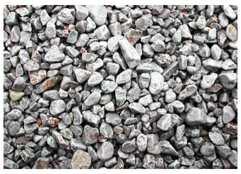
Framework	Distance	Format	Size	Target image
ARLab	40cm	portrait	19x29 (551x822)	 TOM HANKS SANDRA BULLOCK EXTREMELY LOUD INCREDIBLY CLOSE THIS DECEMBER
ARToolKit	40cm	landscape	19x24 (1637x2048)	
ARPA			24x11 (675x310)	
Catchoom	40cm	portrait	19x11 (555x320)	
D'Fusion	40cm	both	9x8 (250x243)	
Metaio	40cm	both	7x7 (200x200)	
Vuforia	40cm	both	19x13 (1500x1000)	

Table 6.1.: Default test paramaters

Background	The default size target images are tested on a high contrast background, dark colors versus bright colors.
Distance	The distance is tested on the target images starting from a 10cm distance between the target image and the testing device and gradually increasing the distance by 10cm.
Light intensity	Four cases are considered: 1. Darkness tested in a room at sunrise using only natural light. 2. Direct light from a lamp, up to 50% of the target image being exposed to light. 3. Sudden change in light intensity by closing and opening the window shades every 10 seconds. 4. Computer screen glare by tracking an image on the screen.
Noise	The target image is digitally altered to introduce noise from default 0% to 10%, %30, %50, %70, %90 and 150%.
Viewpoint	Five different viewpoints are tested on a default size image target: 90 ° frontal, 45 ° tilted right, 45 ° tilted left, 45 ° tilted up and 45 ° tilted down.
Visibility	Visibility is tested on the target images starting with a visible area of 20% and increased the area at every step with 10% until the target was detected.
Aspect ratio	The target image is digitally altered to change the aspect ratio either on horizontal or vertical by shrinking either the height or the width to one third (1/3), half (1/2) and two thirds (2/3).
Contrast ratio	The target image is digitally altered to change the contrast ratio in steps of 50: -50, 0 (by default), 50 and 100.
Grayscale image	The default target images are printed out in grayscale and tested for its corresponding framework.
Material	Four cases are considered: 1. Standard printing paper. 2. Target image placed behind a glass window. 3. Target image plasticized to simulate a plastic printed target. 4. Glossy photo printing paper.
Size	The target images are printed out in different sizes keeping the aspect ratio. The height is set to 5cm, 10cm, 15cm and 20cm.

Table 6.2.: The tests executed per criterion

material and size requirements. The target images are printed on custom A4 paper using a colored ink jet printer.

A criterion is tested at most five times in a row before declared not detected. A maximum of 30 seconds per test is enough to decide if the framework can successfully detect the target image under the specific criterion. A minimum of three tests per criterion is stored in the database.

6.2. Evaluation and Analysis

In order to preserve the same testing conditions such as sunset light, a criterion is tested by each framework before moving on to the next criterion. Furthermore, the testing environment changes between criteria and by testing one criterion at a time for every framework, the environment is minimally altered during the testing.

The execution and results of the online criteria are described in this section. For each criterion, it is concluded in a table if the criterion tests succeeded or not on the evaluated framework. The cells left open imply "no". Additionally, for each criterion the graph containing the testing times for the evaluated criterion over all frameworks is shown.

As each tested criterion has specifics and each specific has been tested more than once, the testing times represent the average of its specifics average times. The order of the frameworks is not visible on the x axis of the graphs. Therefore, the order is consistent in all graphs as follows: ARLab, ARToolKit, D'Fusion, Vuforia, catchoom and metaio. In the Annex, each criterion is further detailed on its specifics and the testing times represent the best scored time by a framework on a specific.

ARPA Solutions framework requires an active license, which was procured but from some technical reasons, the connection could not be made. Therefore it has been removed from the final project.

6.2.1. Background

The evaluation of the background criterion has been performed in two background test sequences, dark versus bright contrasts. This reproduction is meant to simulate high contrast backgrounds, as it would be the case for apps like the Tourist Translator or an mCommerce app. As it can be seen in Table 6.3 most of frameworks overcome a bright contrast background, granted, with more difficulty than a darker background, as proven by the testing times in Figure 6.1.

Background	ARLab	ARToolKit	D'Fusion	Vuforia	catchoom	metaio
Bright colors	yes	yes	yes	yes	yes	yes
Dark colors	yes	yes	yes	yes	yes	yes

Table 6.3.: The background specifics successfully tested on each framework

As it can be observed in Figure 6.1, D'Fusion supports both bright and dark backgrounds in a constant manner. D'Fusion and Vuforia have similar times dealing with dark contrasts while Vuforia performs slightly better in the case of brighter backgrounds. Vuforia is the only framework that has stored a better time on bright backgrounds than a darker one. As both graphs can attest to it, metaio is the leader of the background criterion.

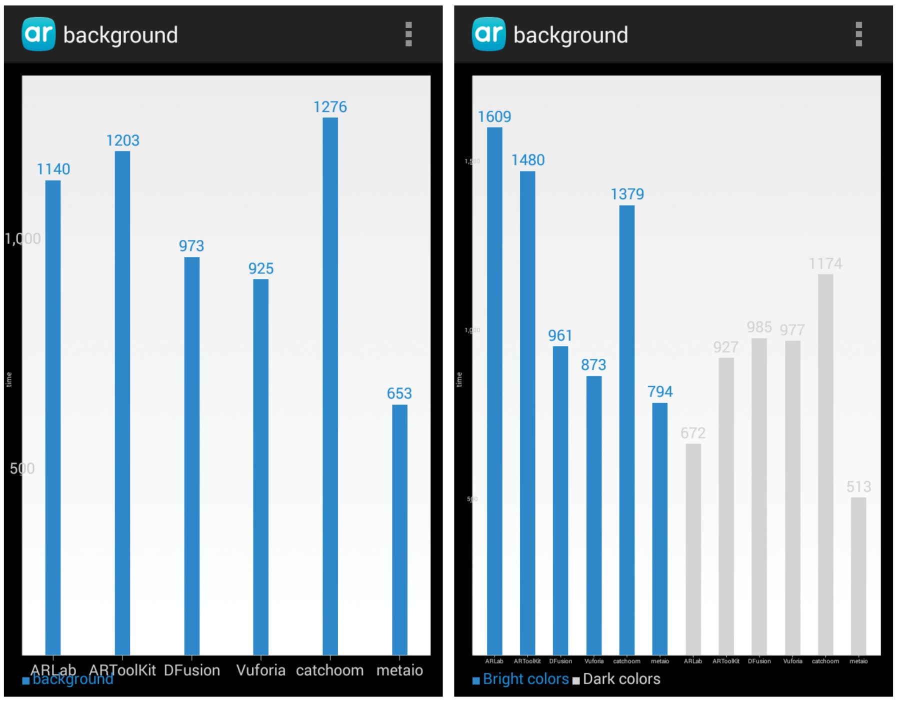


Figure 6.1.: The arithmetic mean of the average times of the background specifics on the left and the average times per specific on the right

6.2.2. Distance

The evaluation of the distance criterion has been performed by starting testing up close, at a 10cm distance from the image target. The distance is increased at every step 10 cm until the framework cannot recognize the image target anymore. As it is observed in Table 6.4, the Metaio framework reaches 2 meters and 40cm.

The testing times shown in Figure 6.2 portray Vuforia as the fastest framework when it comes to the distance between the testing device and the target image. Metaio should be on the second place, if not on first place, considering that its average testing time increased because of the extra effort required to detect the target from 240cm away.Catchoom is a little slower

Distance	ARLab	ARToolKit	D'Fusion	Vuforia	Catchoom	metaio
10cm	yes	yes				
20cm	yes	yes		yes	yes	yes
30cm	yes	yes	yes	yes	yes	yes
40cm	yes	yes	yes	yes	yes	yes
50cm	yes	yes	yes	yes	yes	yes
60cm	yes		yes	yes	yes	yes
70cm	yes		yes	yes	yes	yes
80cm	yes		yes	yes	yes	yes
90cm			yes	yes	yes	yes
150cm				yes	yes	yes
210cm				yes		yes
240cm						yes

Table 6.4.: The number of tests performed per framework

that D'Fusion. However, it can recognize the target image from 10cm closer or farther away than D'Fusion. The worst timing is registered by the ARLab framework.

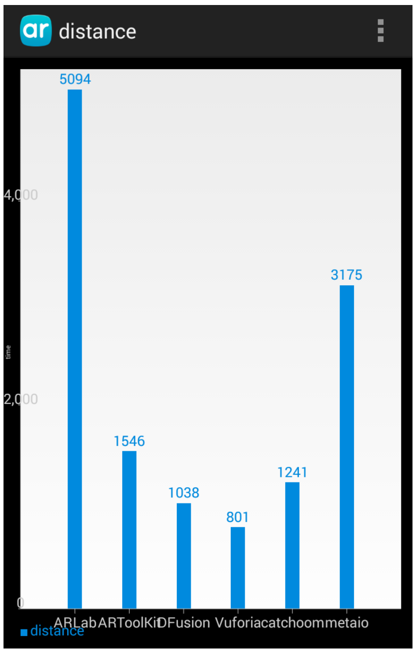


Figure 6.2.: The average testing time for each framework evaluated on the distance criterion

As it can be seen in Figure 6.3, metaio needs over 10 seconds to detect the target image at 230 and 240cm away. Up to 200cm metaio takes under a second for a successful augmentation.

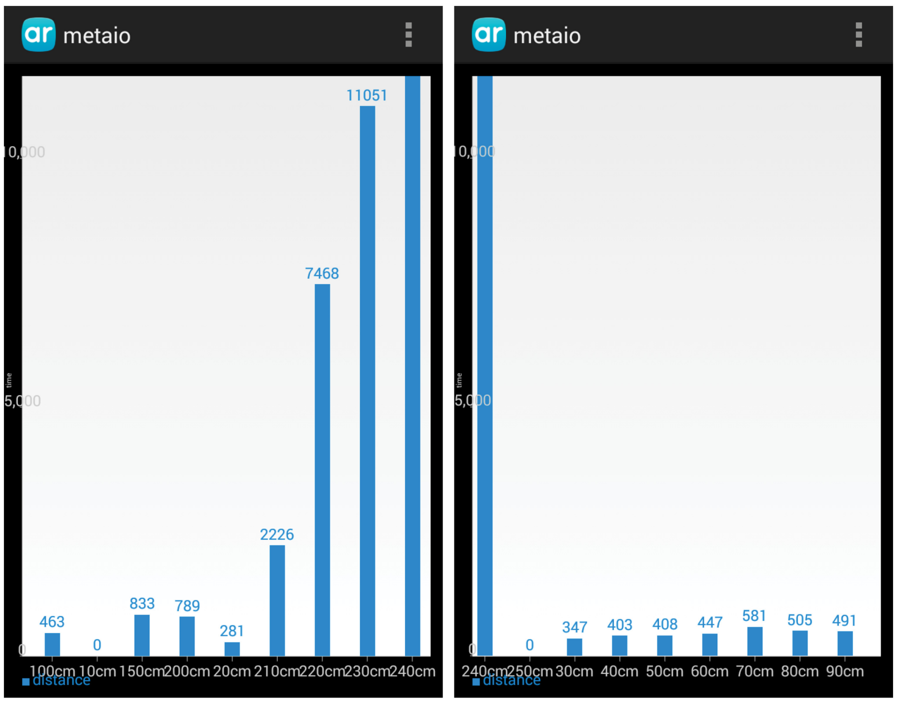


Figure 6.3.: The average times metaio took to detect the target image from various distances

6.2.3. Light Intensity

The evaluation of the light intensity criterion has been executed in special lightning conditions. The darkness criterion specific has been tested inside a room, at natural light at sunrise. The second test is also performed inside a room, by overexposing the target image to the direct light of a desk lamp. The third simulated event happens by closing and opening the window shades. This experiment is meant to replicate sudden changes in light intensity. The mirroring effect is reproduced by tracking the target images on a computer screen.

Light intensity	ARLab	ARToolKit	D'Fusion	Vuforia	catchoom	metaio
Darkness	yes	yes	yes	yes	yes	yes
Direct light	yes	yes	yes	yes	yes	yes
Sudden change	yes	yes	yes	yes	yes	yes
Computer screen	yes	yes	yes	yes	yes	yes

Table 6.5.: The number of tests performed per framework

Light conditions are supported overall as observed from Table 6.5. As Figure 6.4 shows, metaio has the lowest average testing times. Vuforia has an average testing time under a second, as well as metaio. The worst timing is registered by ARToolKit while ARLab and catchoom are steady under a second and a half.

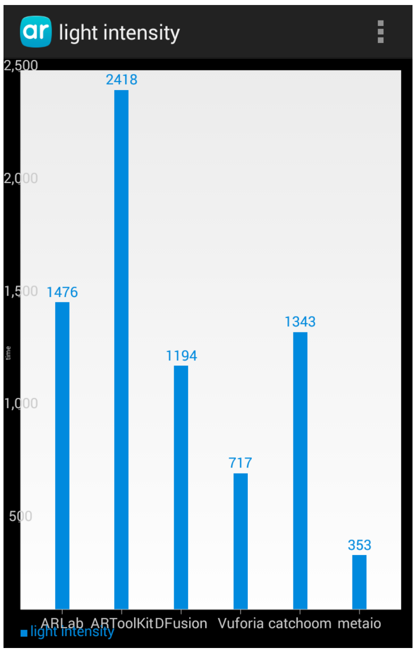


Figure 6.4.: The average testing time for each framework evaluated on the light intensity criterion

Figure 6.5 implies that screen glare is acceptably supported by all frameworks, including a slower ARToolKit. The case that raises most problems is the semidarkness scenario. Metaio presents itself well in all cases followed by Vuforia, while ARToolKit scores the highest testing time under sudden change.

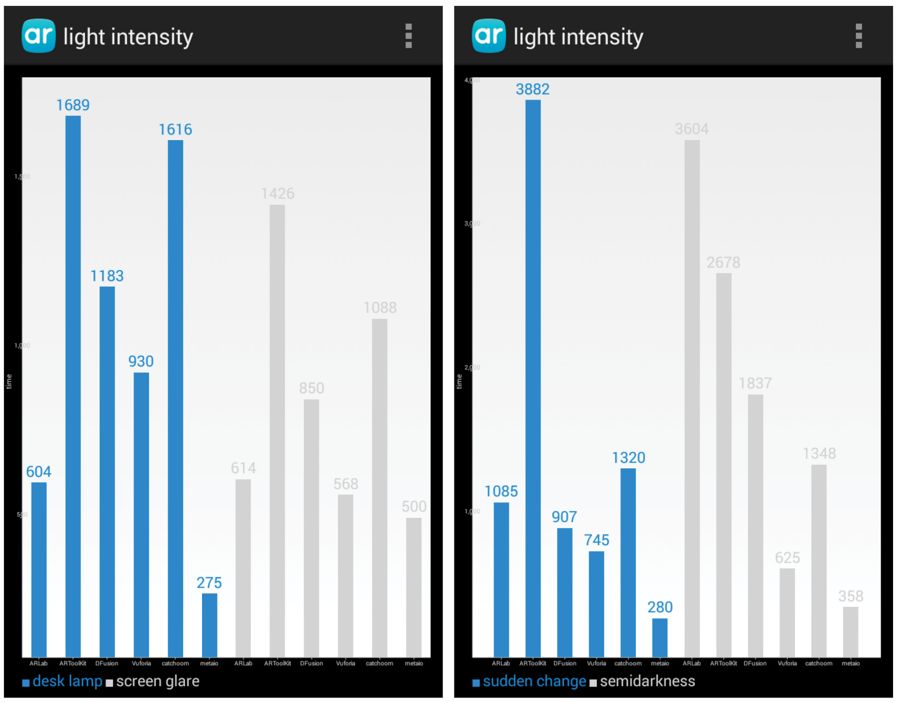


Figure 6.5.: On the left are the average times for the desk lamp and screen glare specifics. On the right are the testing times from the semidarkness and sudden change tests.

6.2.4. Noise

The evaluation of the noise criterion has been performed by digitally altering the target images and adding different noise levels. The test sequence includes six noisy target images. The noise is incrementally added starting from a 10% noisy target image in steps of 20% up to 90%, as well as a 150% and a 200% noisy target, as demonstrated in Figure 6.6.

Noise	ARLab	ARToolKit	D'Fusion	Vuforia	catchoom	metaio
10%	yes	yes	yes	yes	yes	yes
30%	yes	yes	yes	yes	yes	yes
50%	yes		yes	yes	yes	yes
70%	yes		yes	yes	yes	yes
90%	yes			yes	yes	
150%				yes		
200%				yes		

Table 6.6.: The percentage of digitally created noise on a target image that still is recognized

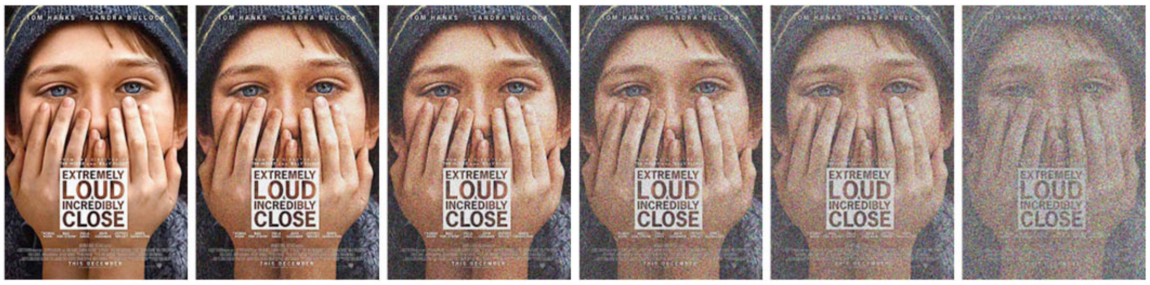


Figure 6.6.: The six noisy versions of the ARLab target image from left to right: 10%, 30%, 50%, 70%, 90% and 150% noise

As it can be seen in Table 6.6 and confirmed by Figure 6.7, Vuforia and metaio have very similar times when dealing with noise. However, Vuforia detects a 200% level noise target image while metaio goes as high as 70%.

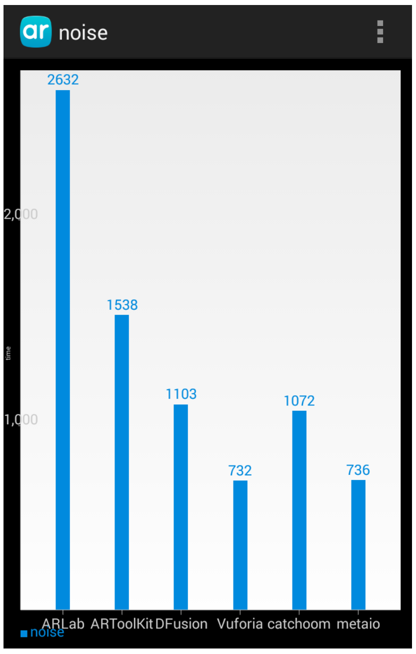


Figure 6.7.: The average testing time for each framework evaluated on the noise criterion

The fastest times are still recorded by metaio while ARLab is the slowest detector. As Figure 6.7 depicts, four out of the six frameworks have a detection time over 1 second. Nevertheless, Vuforia is still the most impressive while metaio outperforms D'Fusion and catchoom.

ARLab requires per average testing time, more than one second per test, going up to almost five seconds when targeting a 70% noise level image as shown by the values in Figure 6.8. Vuforia barely passes the 1 second mark with an average of 1101ms at the 200% noise level.

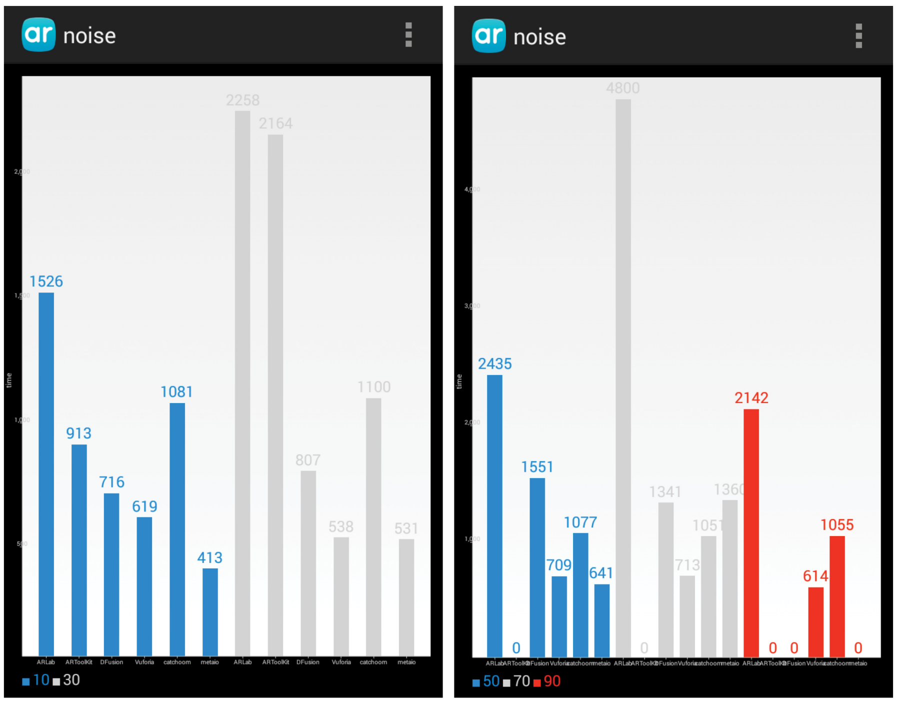


Figure 6.8.: The average testing times for 10 and 30% noise level on the left to 50, 70 and 90% noisy target images on the right

6.2.5. Viewpoint

The evaluation of the viewpoint criterion has been performed in four different perspectives, as demonstrated in Figure 6.9. The viewpoint changes 45 ° left, right, up and down from a 90 ° frontal view.

Viewpoint	ARLab	ARToolKit	D'Fusion	Vuforia	catchoom	metaio
45 ° left	yes		yes	yes	yes	yes
45 ° right	yes		yes	yes	yes	yes
45 ° up	yes	yes	yes	yes	yes	yes
45 ° down	yes	yes	yes	yes	yes	yes

Table 6.7.: The 45 ° angles supported by the frameworks

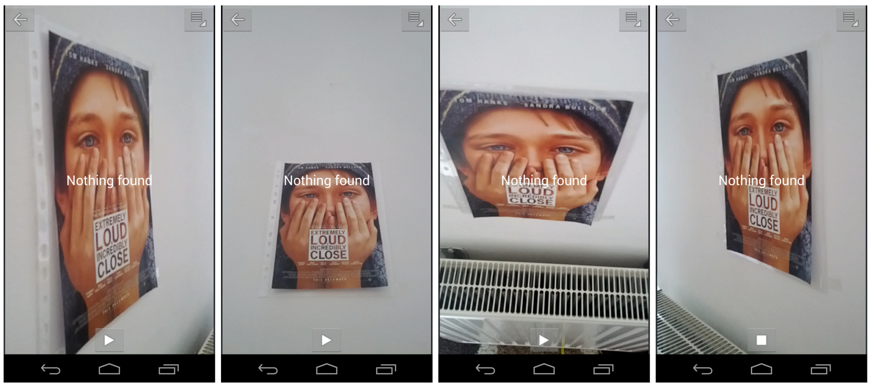


Figure 6.9.: The four viewpoint perspectives that are tested

As it can be seen in Table 6.7, most of the frameworks detect 45° angles. The testing times are shown in Figure 6.10. It is recommended for stronger angles to be tested. ARToolkit provides good testing times, nevertheless it should be considered that it recognizes only two out of the four perspectives.

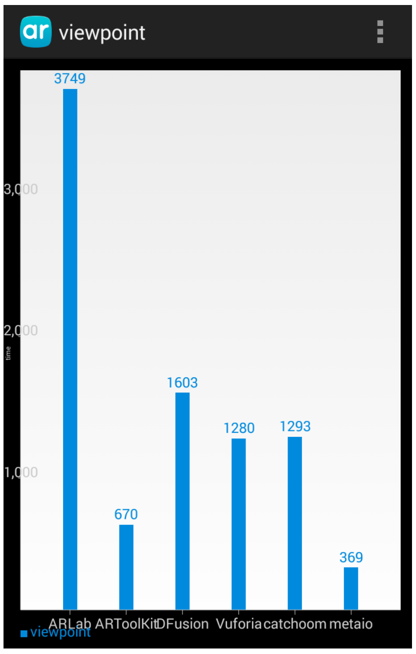


Figure 6.10.: The average testing time for each framework evaluated on the viewpoint criterion

ARLab has difficulties detecting the target image from a 45° upward angle. As it can be seen in Figure 6.11, the other perspectives are easier to detect. Three out of the six frameworks detect the target image from a 45 ° right angle under half a second.

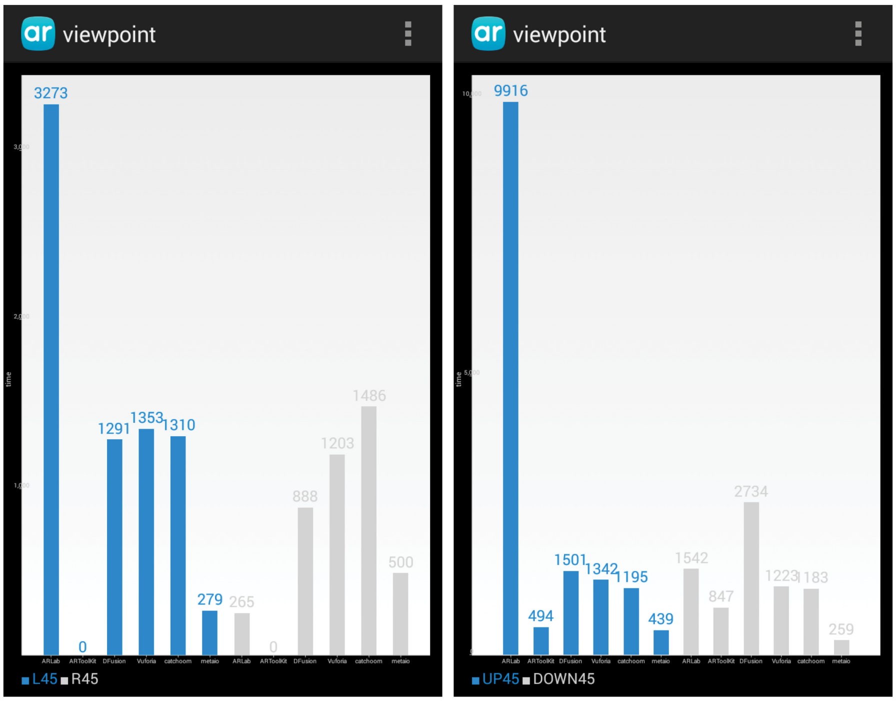


Figure 6.11.: The average testing times for 45 ° left and right angles on the left and 45 ° up and down angles on the right

6.2.6. Visibility

Visibility is tested by uncovering 10% of the visible area at every step. A white sheet of paper is used as demonstrated in Figure 6.12 to cover the target image and uncover it in incremental steps. When the target is recognized and the view augmented, the step increasing stops.

Visibility	ARLab	ARToolKit	D'Fusion	Vuforia	catchoom	metaio
10%				yes		
20%				yes	yes	
30%			yes	yes	yes	
40%	yes		yes	yes	yes	yes
80%	yes	yes	yes	yes	yes	yes
90%	yes	yes	yes	yes	yes	yes

Table 6.8.: The test result having a percent of the target image visible

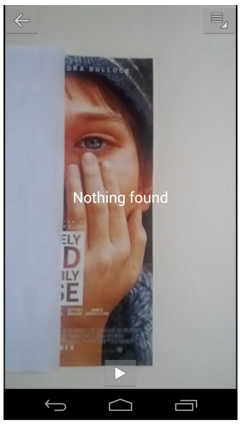


Figure 6.12.: The target image has a 40% visible area

As it can be seen in Table 6.8, Vuforia can detect a target image with very little visible area. Metaio was able to detect the target at 40% with the condition that the white sheet of paper is not visible in the camera frame. Once the white paper is visible, the tracking is lost. The testing times are shown in Figure 6.13.

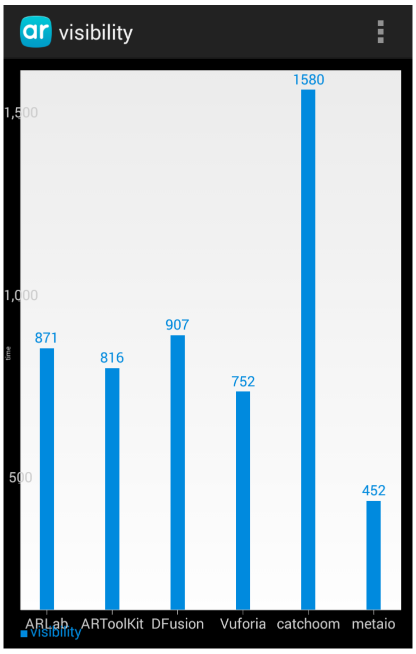


Figure 6.13.: The average testing time for each framework evaluated on the visibility criterion

Based on these averages, metaio is almost three times faster than cathcoom. Catchoom has the lowes performance with an average close to a second and a half. The other four frameworks have registered similar timings, all under the one second mark.

Detection at 10% and 20% require a longer processing time. Vuforia detects the target at 10% visibility somewhere between one and two seconds, being the only framework that scores this performance. For more than 20% visibility Figure 6.14 illustrates that the average time is under a second. Catchoom detects the target image, having only 20% uncovered, in little over two seconds and faster given more visibility. ARLab and metaio need at least 40% of visible area before starting detection while ARToolKit cannot detect anything under 80% visibility.

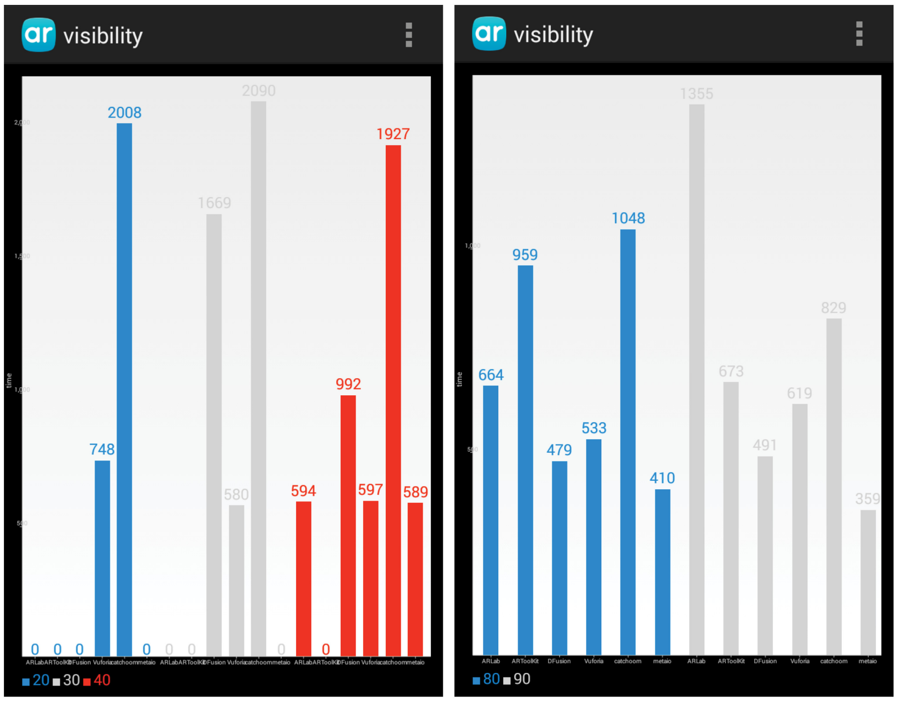


Figure 6.14.: The average testing times for 20, 30 and 40% visible area on the left and 80 and 90% visibility on the right

6.2.7. Aspect Ratio

The evaluation of the aspect ratio requires the target images to be digitally modified. Thus, six tests are performed on the target images illustrated in Figure 6.15 and Figure 6.16. The first row depicts a vertical shrinking by setting up the value of the height to its third (1/3) in one case, a half (1/2) and two thirds (2/3). The second row shows the same operations done on the horizontal ending up with three images whose width has been reduced to one third, a half and two thirds.

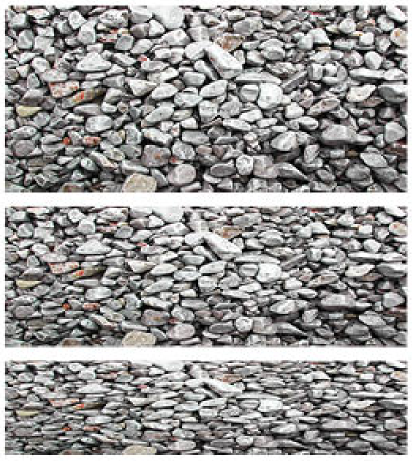


Figure 6.15.: The three changes in aspect ratio done on the horizontal

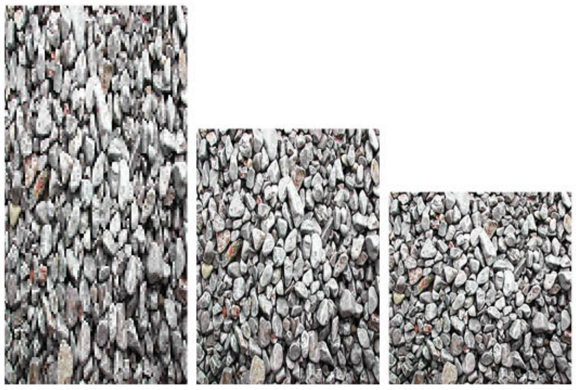


Figure 6.16.: The three changes in aspect ratio done on the vertical

As it can be seen in Table 6.9 metaio is the only framework that supports all six aspect ratio changes. ARLab, on the other hand, detects nothing. Vuforia is the closest framework to metaio. Nevertheless, metaio still has the best times and covers all constraints.

Aspect ratio	ARLab	ARToolKit	D'Fusion	Vuforia	catchoom	metaio
Horizontal 1/3			yes			yes
Horizontal 1/2			yes	yes	yes	yes
Horizontal 2/3		yes	yes	yes	yes	yes
Vertical 1/3						yes
Vertical 1/2			yes		yes	yes
Vertical 2/3			yes	yes	yes	yes

Table 6.9.: The tests that succeeded for each framework

Figure 6.17 presents ARToolKit, D'Fusion and catchoom between a second and a half and two seconds. Vuforia stays close to metaio while ARLab is nonexistent.

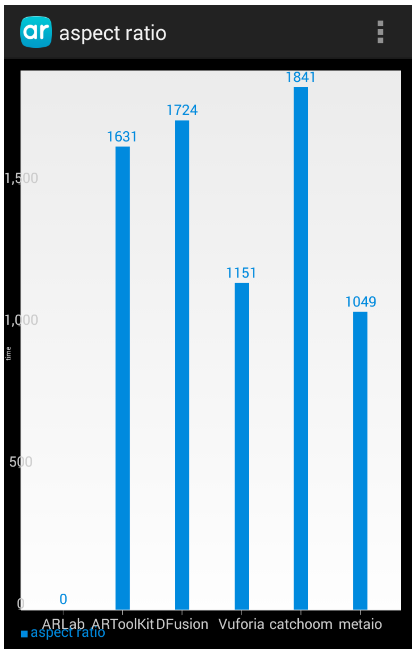


Figure 6.17.: The average testing time for each framework evaluated on the aspect ratio criterion

Figure 6.18 proves that Metaio scores the quickest times. However, D'Fusion is quicker on the one third horizontal. D'Fusion also has the highest average time on the half vertical, over three seconds, where coincidentally metaio scores the fastest time.

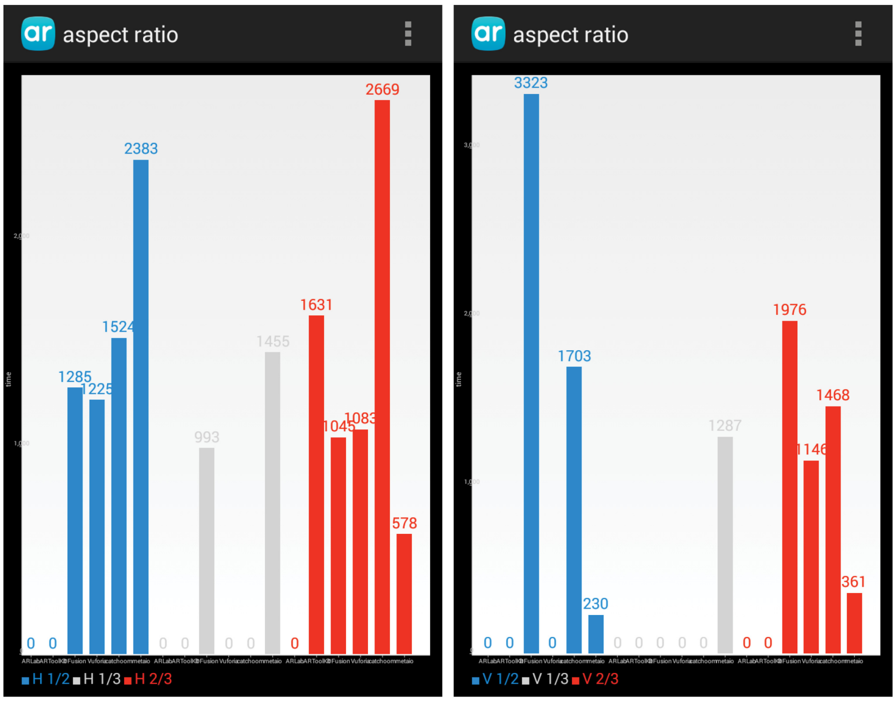


Figure 6.18.: On the left are the average test times for the horizontal shrinking and on the right, the vertical ones

6.2.8. Contrast Ratio

The evaluation of the contrast ratio requires for the target images to be digitally modified. Thus, four tests are performed on the target images illustrated in Figure 6.19. The first image has the contrast value set to -50 and at each step the contrast value is incremented by 50. Thus the second image has contrast 0 as the default image, and the next two images have a contrast value of 50, respectivly 100.

Contrast ratio	ARLab	ARToolKit	D'Fusion	Vuforia	catchoom	metaio
-50	yes	yes	yes	yes	yes	yes
0	yes	yes	yes	yes	yes	yes
50	yes	yes	yes	yes	yes	yes
100	yes	yes	yes	yes	yes	yes

Table 6.10.: All contrast ratio tests passed

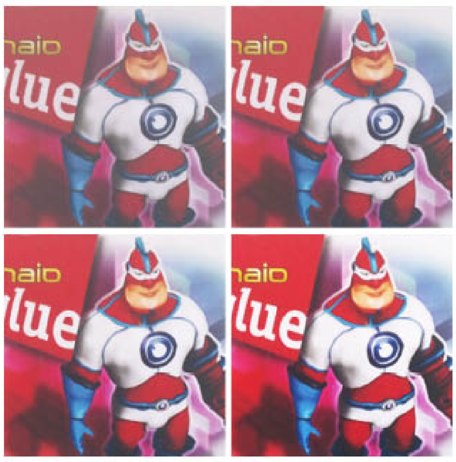


Figure 6.19.: The four digitally modified Metaio's target images with contrast values in the range -50 to 100

Table 6.10 states that all frameworks passed the contrast ratio test, nevertheless not without difficulties as Figure 6.20 concludes. ARLab tested poorly and metaio holds the fastest testing times.

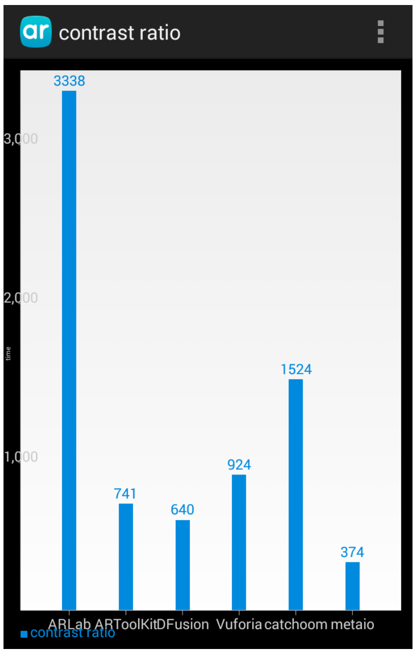


Figure 6.20.: The average testing time for each framework evaluated on the contrast ratio criterion

Three other frameworks test on average under one second together with metaio: ARToolKit, D'Fusion and Vuforia as depicted in Figure 6.20. Catchoom has an average testing time of one and a half seconds while ARLab needs more than half in order to detect target images of different contrast levels.

Figure 6.21 shows that Metaio scores the quickest times at every criterion specific, very closely together for a -50 and a +50 contrast over the original. A +100 contrast takes metaio slightly longer to detect the target. ARToolKit is not far behind metaio, while D'Fusion and Vuforia are once again in the same range.

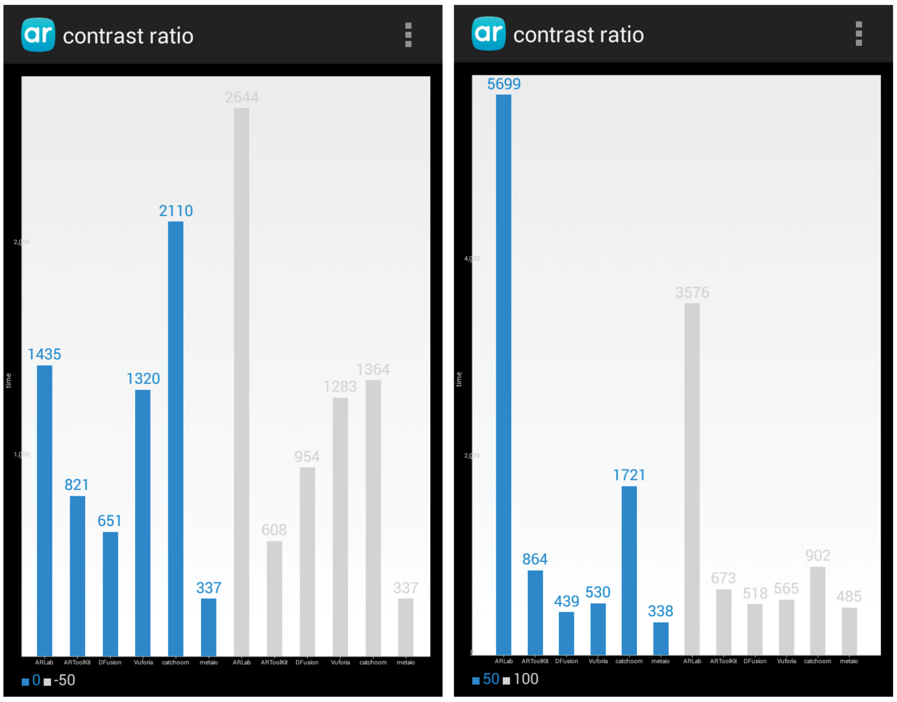


Figure 6.21.: The average testing times for the default target image and -50 contrast level are on the left and +50 and +100 contrast levels on the right

6.2.9. Grayscale

The evaluation of the grayscale criterion has been performed on a printed default size target image in grayscale, as seen in Figure 6.22. All frameworks can detect a grayscale image. It is suggested to test both the grayscale version and the original colored target image in order to compare the timing needed for them to be recognized.

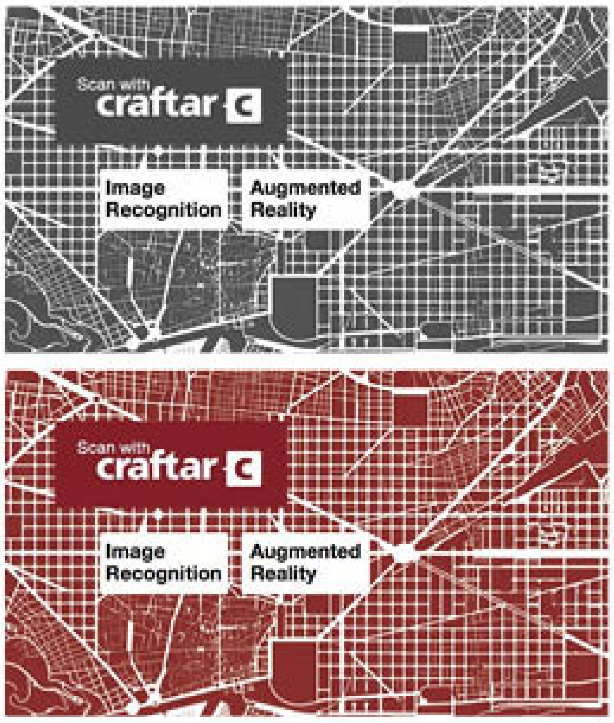


Figure 6.22.: The grayscale version of the Catchoom's target image is on top

All frameworks recognize the grayscale version of their original target, as corroborated by Table 6.11.

Grayscale	ARLab	ARToolKit	Catchoom	D'Fusion	Metaio	Vuforia
Gray	yes	yes	yes	yes	yes	yes
Colored	yes	yes	yes	yes	yes	yes

Table 6.11.: All frameworks support grayscale detection

The testing times are shown in Figure 6.23, metaio having the lowest testing time while three other frameworks tested in under a second. Catchoom and ARLab have recorded times a little over a second. As it can be seen, colored or gray, the detection time is roughly the same. A slight difference can be observed at ARLab.

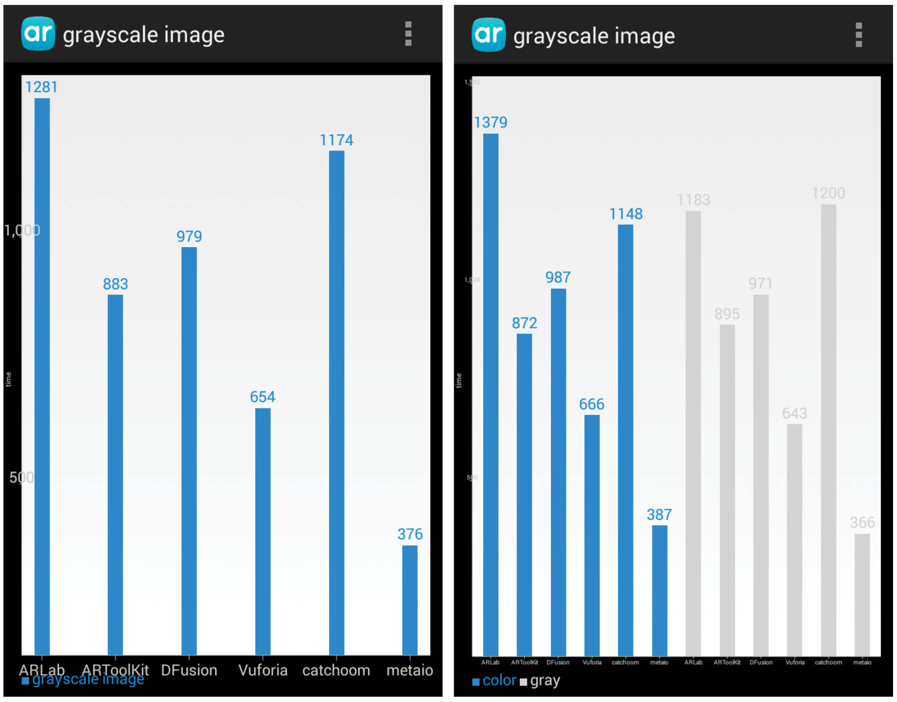


Figure 6.23.: On the left are the average testing times for each framework, on the right both the gray and color target images' average testing times

6.2.10. Material

The evaluation of the material criterion has been performed in three different stages. The first case depicts the target image behind a glass window, as it would be in the Bus Shelter App, where the poster is placed in between the glass windows of the side panels. The second case simulates a restaurant menu scenario where the menu is plasticized to avoid its deterioration over time and spilled drinks. The third target image is printed on a glossy photo printing paper as it might be used in the Supermarket Promotion scenario for displaying promotional ads or as packaging of different promotional products.

Material	ARLab	ARToolKit	D'Fusion	Vuforia	catchoom	metaio
Glass	yes		yes	yes	yes	yes
Plasticized paper	yes	yes	yes	yes	yes	yes
Glossy photo paper	yes	yes	yes	yes	yes	yes

Table 6.12.: The frameworks that support glass, plastic and glass printouts

As it can be seen in Table 6.12 all of the above materials are supported by every framework, but not without difficulties. The testing times are shown in Figure 6.24. ARToolKit was not tested on glass.

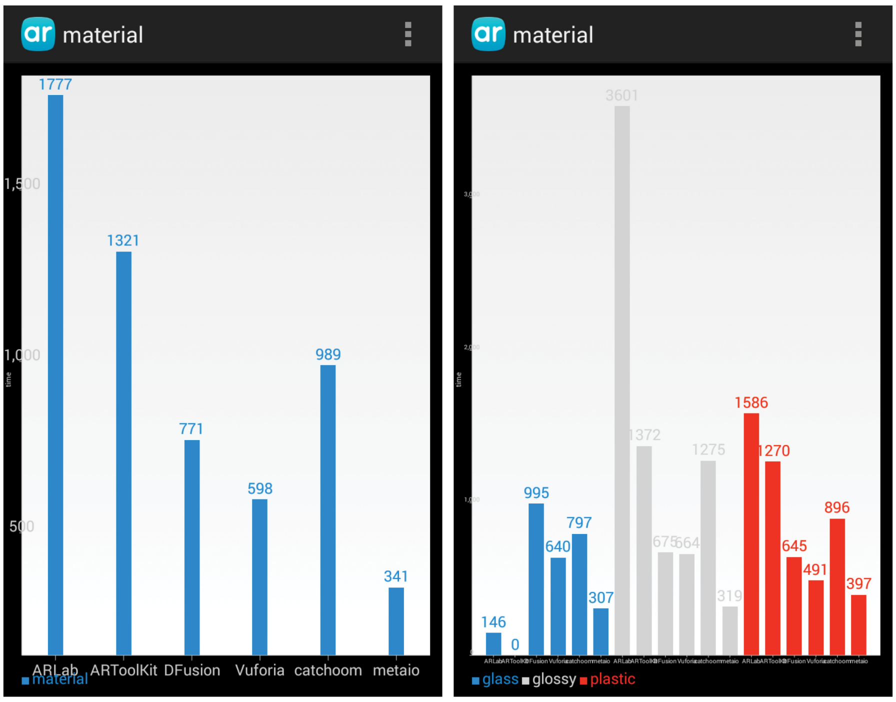


Figure 6.24.: On the left are the average times of the material criterion and on the right, a detailed graph of the specifics' results

ARLab has the best time on glass detection and the worst on plastic. Metaio leads the glossy and paper specifics while D'Fusion and Vuforia tolerate them, showing a worse performance on glass material. ARToolKit keeps itself constant around the one second marker.

6.2.11. Size

The evaluation of the size criterion has been executed on four different target images. Each of the six tested target images have been reduced to 5cm, 10cm, 15cm and 20cm height as exemplified in Figure 6.25. The distance between the testing device and the target image has been constant at a default of 30cm.

As it can be seen in Table 6.13 ARLab only recognizes 10cm high target images and D'Fusion does not detect 20cm high images while ARToolKit only recognizes 20cm high targets. The testing times are shown in Figure 6.26.

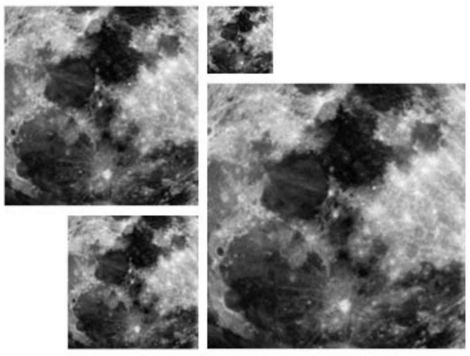


Figure 6.25.: The four different sizes used for testing, exemplified on the D'Fusion's target image

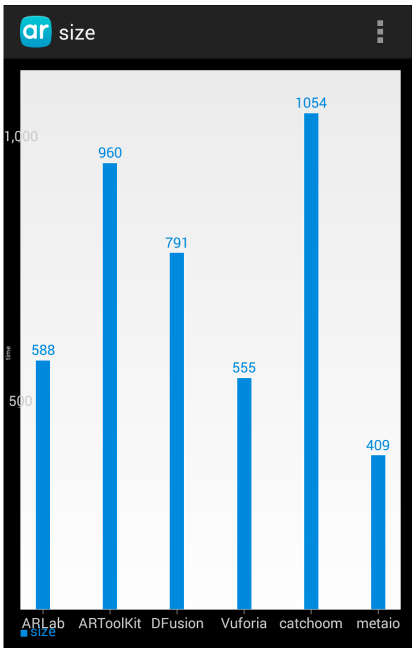


Figure 6.26.: The average testing time for each framework evaluated on the size criterion

Metaio has the fastest time, as illustrated in Figure 6.26. Close behind are ARLab and Vuforia, both a little over half a second. ARLab only supports a 10cm high marker, while Vuforia supports all sizes alongisde catchoom and metaio as determined by Table 6.13.

The frameworks that detect different sizes, keep the average testing times through out the tests constant as proved by Figure 6.27. Catchoom tests the highest average time and most of the various target sizes are detected under one second. Catchoom needs over a second when dealing with bigger sizes.

Size	ARLab	ARToolKit	D'Fusion	Vuforia	catchoom	metaio
5cm			yes	yes	yes	yes
10cm	yes		yes	yes	yes	yes
15cm			yes	yes	yes	yes
20cm		yes		yes	yes	yes

Table 6.13.: The case of successful results considering the size constraints

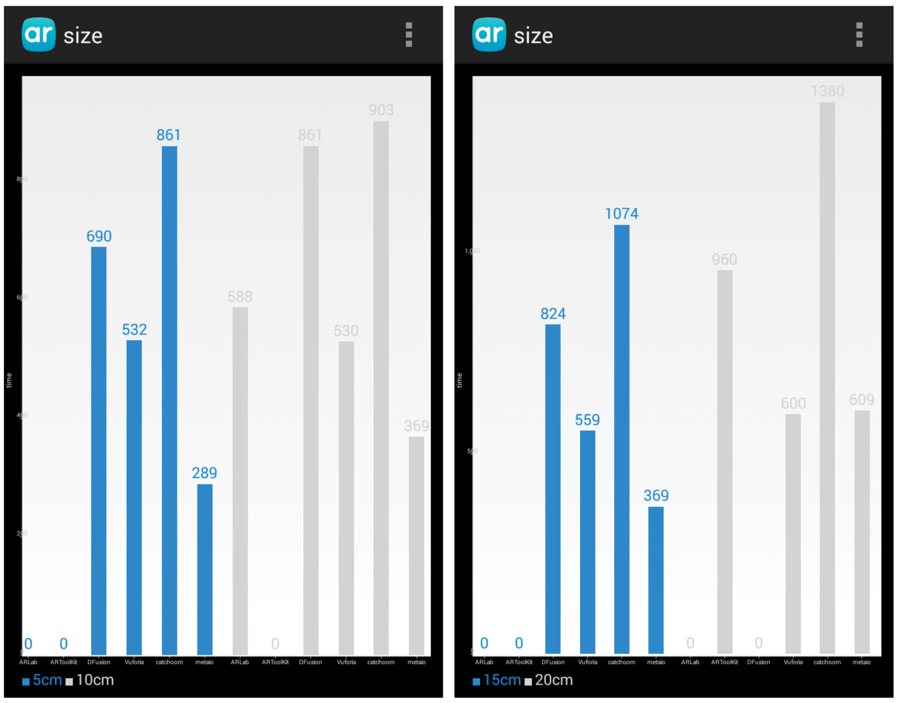


Figure 6.27.: The average testing times for 5 and 10cm high target images are on the left and 15 and 20cm on the right

6.3. Scenarios

The best fitted framework for each of the scenarios defined in Chapter 4 are given in the contents of this section. Additional to the online criteria, for each framework it has been determined which features are supported and which are not in the tables below.

Table 6.14 presents the performance criteria, as observed during the active testing of the frameworks. It has been decided that none of the frameworks support occlusion. The ARLab overlaying text on top of the target image during testing is not relevant to the performance criteria, thus they could not be determined. For adding the weight of the criteria that are supported, a simple question must be answered: "Should the framework be awarded these points for this criterion?". Thus, a minimum flicker will result in No.

Performance	Registration	Flicker	Motion blur	Fast moves	Occlusion
ARLab					
ARToolKit	not supported	min	no	partially supported	not supported
D'Fusion	supported	no	no	supported	not supported
Vuforia	supported	min	no	supported	not supported
Catchoom	supported	no	yes	supported	not supported
Metaio	supported	min	no	not supported	not supported

Table 6.14.: Performance criteria support for each framework

Multi-targets, face tracking and text detection criteria have not been actively tested. Additional features must be integrated within each framework for these criteria to be tested. It has been decided not to pursue the online evaluation of the mentioned criteria as the app reached over 150MB. Most of the frameworks detailed the features they support in the product information on their websites and these are concluded in Table 6.15. Not all criteria has been found in the research, thus the cells in the table are left blank.

Usability	Multi targets	Face tracking	Text detection	Extended tracking	Front camera	Flash camera
ARLab	yes				yes	
ARToolKit	yes				yes	
D'Fusion		yes		yes	yes	
Vuforia	yes	no	yes	yes	yes	yes
Catchoom	yes	no	no			no
Metaio	yes	yes	yes	yes	yes	

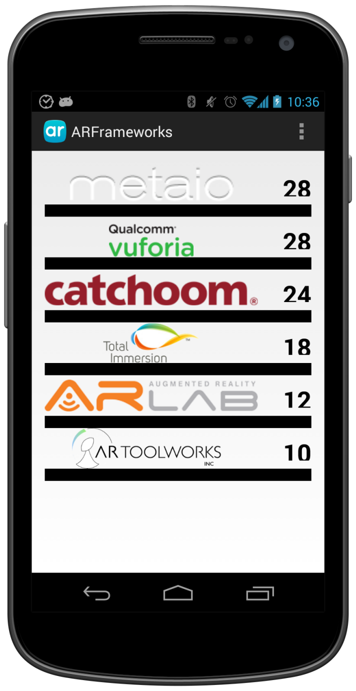
Table 6.15.: Usability criteria support for each framework

6.3.1. Interior Design App

Considering the Interior Design scenario and the constraints defined in Chapter 4, section 4.6.1 Interior Design App, the evaluated criteria determine the classification in Figure 6.28.

For the environmental criteria, the following criterion specifics are considered relevant:

- Viewpoint: all possible viewpoints (45° left, 45° right, 45° up, 45° down)
- Distance: minimum 90/100cm
- Light intensity: semidarkness
- Visibility: 80/90%

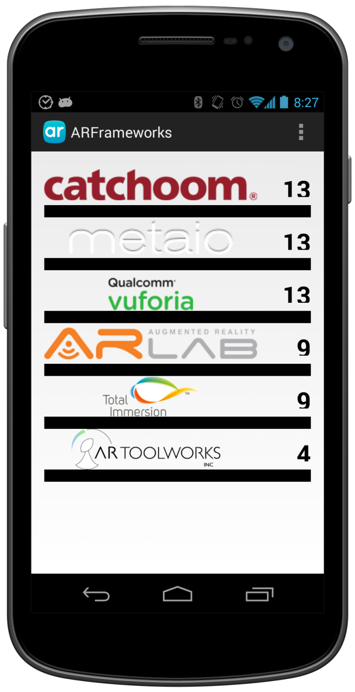


metaio	28
Qualcomm vuforia	28
catchoom	24
Total Immersion	18
AR LAB	12
AR TOOLWORKS	10

Figure 6.28.: Framework classification for the Interior Design App

The virtual object must be seen from all possible angles in order for the user of the app to have an accurate understanding of its design/shape. A minimum distance of 90/100cm is required from the smartphone to the target image such that the virtual object is observable at a reasonable scale. Granted, this distance is proportional to the height of the user.

Light intensity does not present an important constraint in this specific scenario, as it is assumed that the area where the target image is placed is in the open, visible, well illuminated space. Furthermore, considering the open space, the visible area of the target image is almost always at 100%. Nevertheless, a minimum of 80/90% of the target's area must be visible as there might be object in the line of sight that obstruct the target, such as other present furniture pieces like an armchair or a flower bowl. The environmental criteria classify the frameworks as in Figure 6.29.



catchoom	13
metaio	13
Qualcomm vuforia	13
AUGMENTED REALITY ARLAB	9
Total Immersion	9
AR TOOLWORKS INC	4

Figure 6.29.: Environmental criteria classification for the Interior Design App

The only target criterion evaluated within the Interior Design scenario is the size of the target image. It has been determined that an optimal size, considering the minimum distance, would be 20cm. Thus, ARToolKit, catchoom, metaio and Vuforia gain 3 extra points. By adding the performance and usability criteria, the final classification is shown in Figure 6.28.

6.3.2. Magazine App

Considering the Magazine scenario and the constraints defined in Chapter 4, section 4.6.2 Magazine App, the evaluated criteria determine the classification in Figure 6.30.

The following environmental criterion specifics are considered relevant:

- Light intensity: desk lamp, screen glare, semidarkness
- Visibility: 40%

- Viewpoint: an upward perspective (45 ° up)
- Noise: up to 50/70%
- Distance: 10/20cm

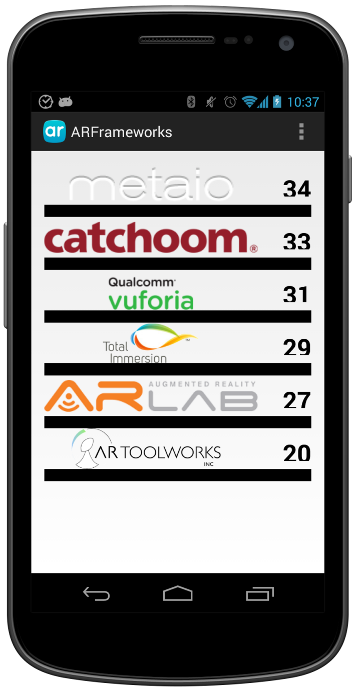
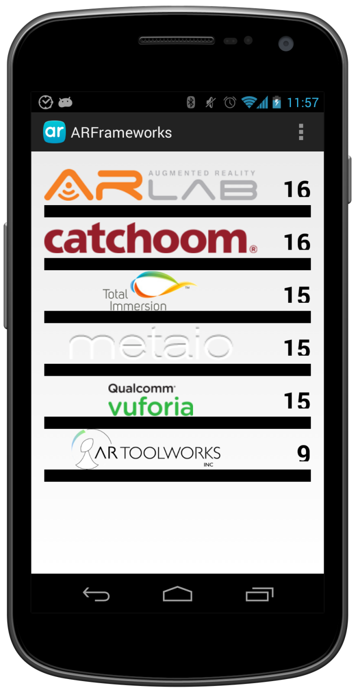


Figure 6.30.: Framework classification for the Magazine App

Different light conditions can have a great impact on a target image. Take for example a direct light from a desk lamp, or a semidark environment. The magazine can be online distributed, which makes screen glare a real problem. A minimum of 40% of the target must be visible to detect the target. Such a small visible area can be the result of paper deterioration or ripped out pages. Most of the time when a target image inside a magazine is being tracked, the user holds the smartphone in front of the magazine at an upward viewpoint.

An image inside a magazine can suffer deterioration over time, such as spilled coffee, food splatter or paper washed out colors. A noise up to 50/70% has been considered. The distance between the smartphone and the magazine page is between 10 to 20cm. The effect of the evaluated environmental criteria can be seen in Figure 6.31



ARLAB AUGMENTED REALITY	16
catchoom	16
Total Immersion	15
metaio	15
Qualcomm vuforia	15
ARTOOLWORKS INC	9

Figure 6.31.: Environmental criteria classification for the Magazine App

All target criteria are considered within the Magazine scenario as follows and depicted in Figure 6.32:

- Grayscale image
- Material: glossy paper
- Size: 5, 10 and 15cm
- Aspect ratio: 1/2 horizontal or vertical, 2/3 horizontal or vertical
- Contrast ratio: -50, 50 and 100

Not all images inside a magazine are color as well as not always are the magazines printed on custom paper. Some magazines are printed on glossy paper or contain a number of glossy photo paper. Considering an A4 format, the images can be printed somewhere in between 5 and 15cm. An image might me modified from its original version to fit in the magazine's pages, shrunk either horizontally or vertically. The contrast of the images might differ from one batch of magazines to another. Thus, all three contrast levels are taken into consideration. The results of the evaluated criteria are visible in Figure 6.32. The final classification of frameworks is found in Figure 6.30, also included are the scores of the offline criteria.

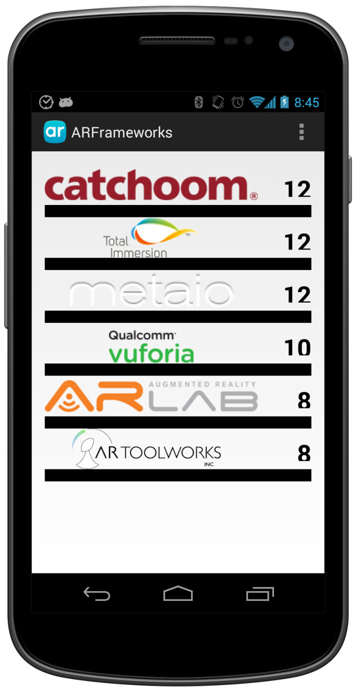


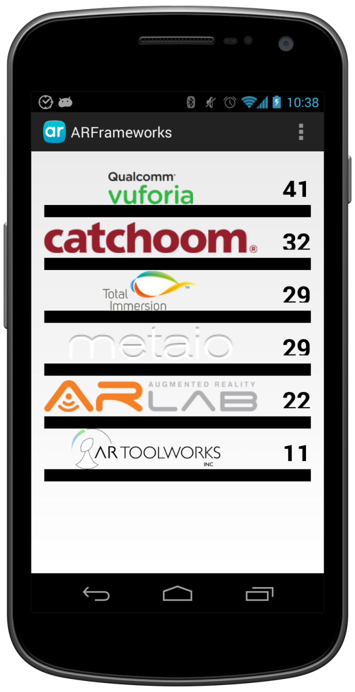
Figure 6.32.: Target criteria classification for the Magazine App

6.3.3. Bus Shelter App

Considering the Bus Shelter scenario and the constraints defined in Chapter 4, section 4.6.3 Bus Shelter App, the evaluated criteria determine the classification in Figure 6.33.

The following environmental criterion specifics are considered relevant in the context of the Bus Shelter scenario:

- Light intensity: sudden change, screen glare
- Viewpoint: 45° left, 45° right, 45° down
- Noise: 90%
- Visibility: 30%
- Background: bright versus dark contrasts
- Distance: from 50cm to 200cm



Qualcomm vuforia	41
catchoom	32
Total Immersion	29
metaio	29
ARLAB AUGMENTED REALITY	22
AR TOOLWORKS INC	11

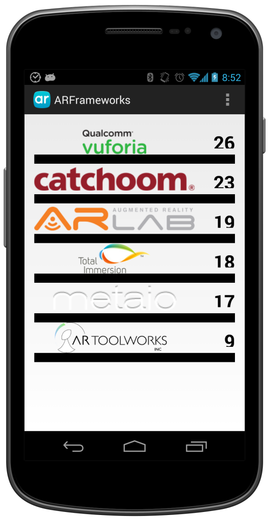
Figure 6.33.: Framework classification for the Bus Shelter App

The Bus Shelter scenario takes place out in the open. Therefore, sudden changes in the light conditions are expected. Being exposed to the environment and sometimes, vandalism, a maximum of 90% noise is considered. Additionally, if the app is used from inside the bus, a window glare, thus mirroring, is a possibility.

The user can stand at many angles from the poster, including a downward perspective while in the bus. Moreover, a minimum of 30% of the poster must be visible in order for it to be detected. A poster on a dark contrast might influence its tracking. The distance between the user and the poster should be in the range anywhere between half a meter and two meters. The effect of the evaluated environmental criteria can be seen in Figure 6.34

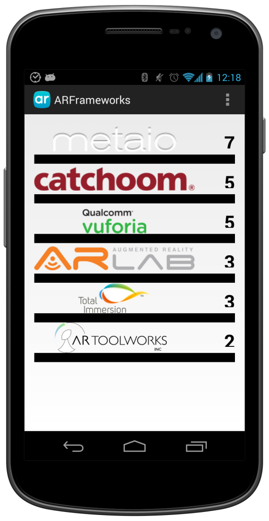
Three target criteria are considered as follows:

- Material: glass, glossy paper
- Aspect ratio: 1/2 horizontal or vertical, 1/3 horizontal or vertical
- Size: a minimum of 20cm



Qualcomm vuforia 26
catchoom 23
ARLAB 19
Total Immersion 18
metaio 17
AR TOOLWORKS 9

Figure 6.34.: Environmental criteria classification for the Bus Shelter App



metaio 7
catchoom 5
Qualcomm vuforia 5
ARLAB 3
Total Immersion 3
AR TOOLWORKS 2

Figure 6.35.: Target criteria classification for the Bus Shelter App

The poster is placed behind the side windows of a bus shelter. Additionally, the poster can be either printed on custom paper or glossy paper. Moreover, the poster can be tracked from inside of a bus, thus double glass constraint. The possibility exists that the poster is reduced to fit a particular side of the side window, either vertically or horizontally. Furthermore, a minimum size of 20cm is required for such a poster in order to be detected. The results of the evaluated target criteria are summed up in Figure 6.35. The final classification is given in Figure 6.33 above.

6.3.4. Supermarket Promotions App

Considering the Supermarket Promotions scenario and the constraints defined in Chapter 4, section 4.6.4 Supermarket Promotions App, the evaluated criteria determine the classification in Figure 6.36.

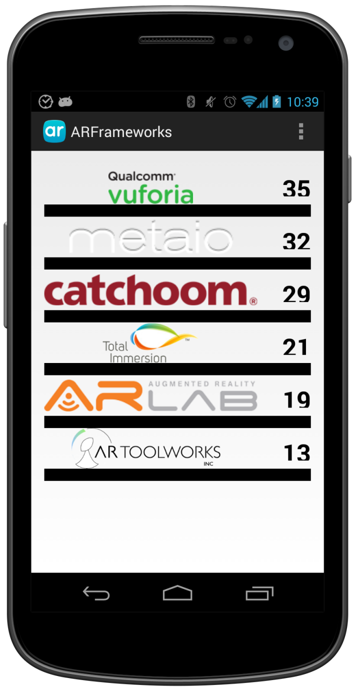


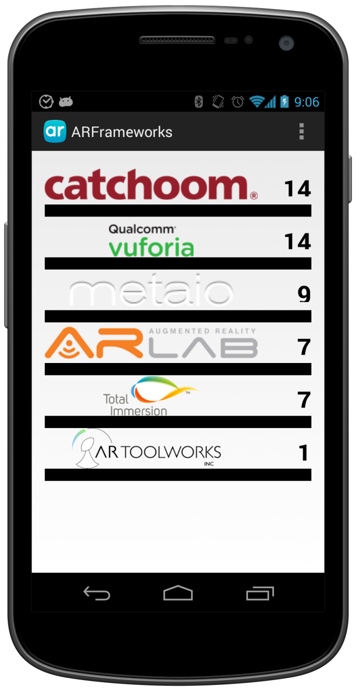
Figure 6.36.: Framework classification for the Supermarket Promotions App

For the environmental criteria, the following criterion specifics are considered relevant:

- Visibility: 20/30%

- Viewpoint: all possible viewpoints (45 ° left, 45 ° right, 45 ° up, 45 ° down)
- Noise: 70%
- Distance: between 40 and 100cm
- Light intensity: semidarkness

A product can be partially occluded by another product, thus the visible packaging must suffice for the detection. Moreover, the package can be situated somewhere up high, thus the extra need for an app that supports strong angles. Not to mention the possibility that the package itself is damaged, making it even more difficult for the app to recognize it. The distance between the customer and the product should be somewhere in between 40 to 100cm. As usually supermarkets are well illuminated, the only light conditions that might impact the detection is semidarkness. Thus considering that an isle is not as lighted as others. The effect of the evaluated environmental criteria can be seen in Figure 6.37



catchoom	14
Qualcomm vuforia	14
metaio	9
ARLAB	7
Total Immersion	7
AR TOOLWORKS	1

Figure 6.37.: Environmental criteria classification for the Supermarket Promotions App

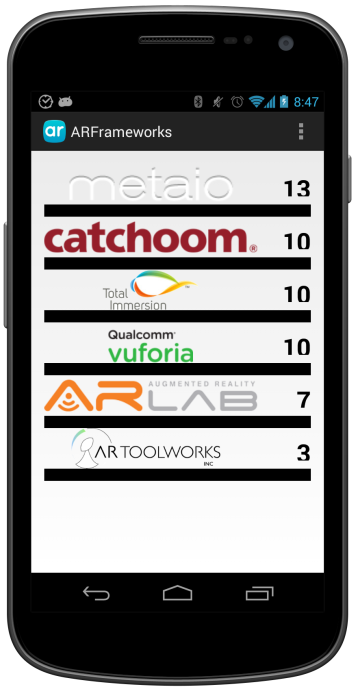
All target criteria are considered within the Supermarket Promotions scenario as follows, and presented in Figure 6.38:

- Material: glossy paper, plastic, glass

- Grayscale image
- Size: 5 and 10cm
- Aspect ratio: 1/2 horizontal or vertical, 1/3 horizontal or vertical

Special Augmented Reality promotions can be written within target images through out the store on special glossy paper. Additionally, the packaging can be made out of plastic or even glass, as wine bottles, for example. Not all packaging must be colored, black and white is always an option. 5, 10 cm is considered to be the default size of any printed Augmented Reality target onto any product. Furthermore, in order for it to fit on different products, the aspect ratio can be modified accordingly. Taking the wine bottle example, a square target image would be reduced vertically to fit the bottle rather in circumference than on height.

The results of the evaluated criteria are visible in Figure 6.38. The final classification of frameworks is found in Figure 6.36, also including the scores of the offline criteria.



metaio	13
catchoom	10
Total Immersion	10
Qualcomm vuforia	10
ARLAB	7
AR TOOLWORKS	3

Figure 6.38.: Target criteria classification for the Supermarket Promotions App

6.3.5. Tourist Translator App

Considering the Tourist Translator scenario and the constraints defined in Chapter 4, section 4.6.5 Tourist Translator App, the evaluated criteria determine the classification in Figure 6.39.

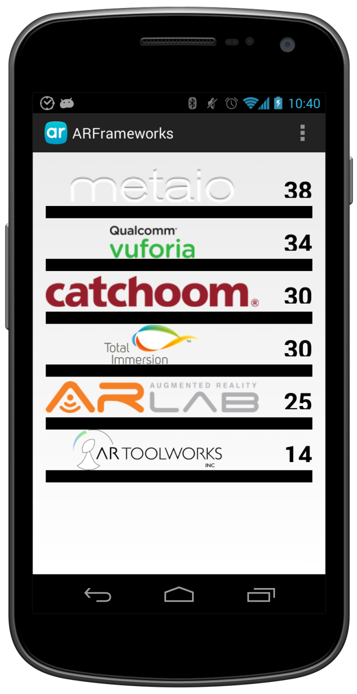
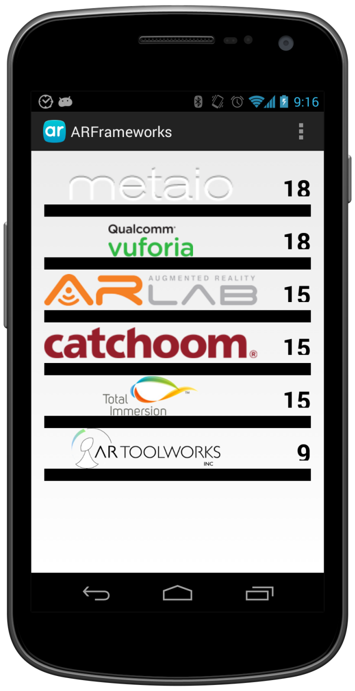


Figure 6.39.: Framework classification for the Tourist Translator App

All environmental criteria are considered relevant in the context of the Tourist Translator scenario:

- Light intensity: sudden change, screen glare
- Viewpoint: 45 ° left, 45 ° right, 45 ° down
- Visibility: 90%
- Distance: from 20 to 200cm
- Noise: 70%
- Background: bright versus dark contrasts

Considering that the Tourist Translator app is mostly used in the open space, it is sensitive to sudden light changes, as well as window glare. This is the case when a sign inside a shop is being tracked and translated. All viewpoints apply, everything surrounding the user can be a potential target, from a flysheet to a street sign. Text detection needs a 100% visibility, or a minimum of 90%. The factor noise should be supported up to a 70%. The background of the word might have an impact on its detection if the text blends into it. The effect of the evaluated environmental criteria can be seen in Figure 6.40



metaio	18
Qualcomm vuforia	18
ARLAB	15
catchoom	15
Total Immersion	15
AR TOOLWORKS	9

Figure 6.40.: Environmental criteria classification for the Tourist Translator App

Three target criteria are considered as follows:

- Contrast ratio: -50
- Size: 10cm
- Material: glass, glossy paper

The difficulty of detecting the printed text is increased when the contrast between the word and the background on which is printed is low. An average size of 10cm is considered for signs around a city which can be printed out on different materials, including glossy paper and glass. The results of the evaluated target criteria are summed up in Figure 6.41. The final classification is given in Figure 6.39 above.

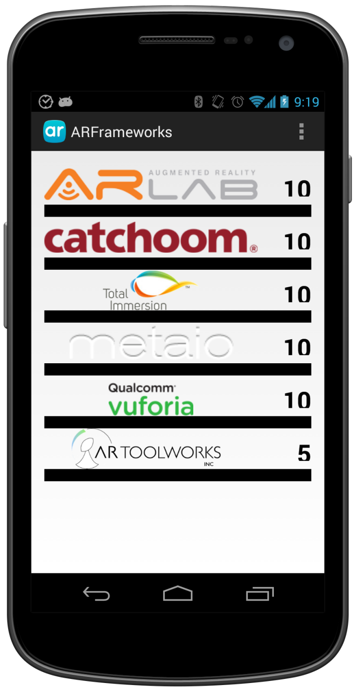


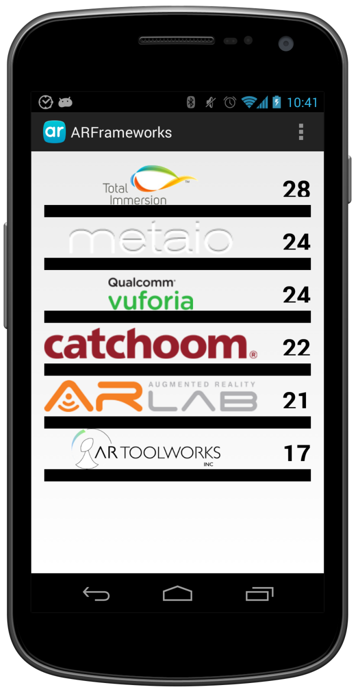
Figure 6.41.: Target criteria classification for the Tourist Translator App

6.3.6. mCommerce App

Considering the mCommerce scenario and the constraints defined in Chapter 4, section 4.6.6 mCommerce App, the evaluated criteria determine the classification in Figure 6.42.

From the online criteria, only the following environmental criteria are considered:

- Light intensity: semidarkness, desk lamp
- Viewpoint: all possible viewpoints (45 ° left, 45 ° right, 45 ° up, 45 ° down)
- Visibility: 80/90%
- Distance: minimum 20cm
- Background: bright versus dark contrast



Total Immersion	28
metaio	24
Qualcomm vuforia	24
catchoom	22
ARLAB Augmented Reality	21
AR TOOLWORKS INC	17

Figure 6.42.: Framework classification for the mCommerce App

This app is intended for home use, thus the customer would be in a closed space where semidarkness or the light from a desk lamp influence the light intensity environmental criterion. All possible viewpoints must be supported for the customer to move the smartphone around and determine if the product he is testing fits the expectations. A visibility of 80, 90% is required for a good detection/tracking process. It is assumed that the customer holds the smartphone, adjusted into his direction and line of sight. Thus, the minimum length for him to use the app is around 20cm. A bright versus dark background might slow down the detection.

The evaluated environmental criteria classify the frameworks as in Figure 6.43. By adding the performance and usability criteria, the final classification is shown in Figure 6.42.

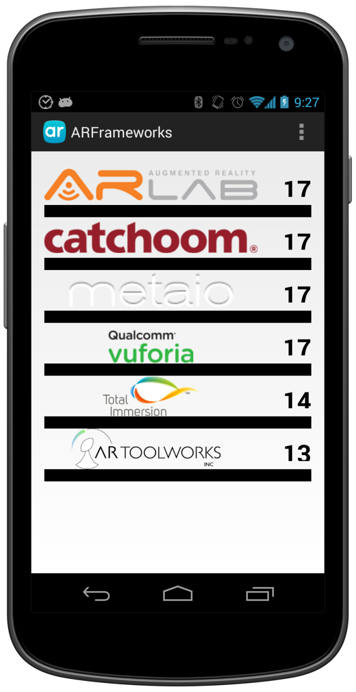


Figure 6.43.: Environmental criteria classification for the mCommerce App

6.4. Frameworks

The results for each framework are given below. Multiple tests have been performed over a period of time and the results in the presented graphs represent the average testing time per criterion. The graphs depict on the left side the environmental criteria category and on the right the target category. Each of the online tested criteria has specifics. For each specific the mean is computed and added to the sum that is being averaged over the criterion.

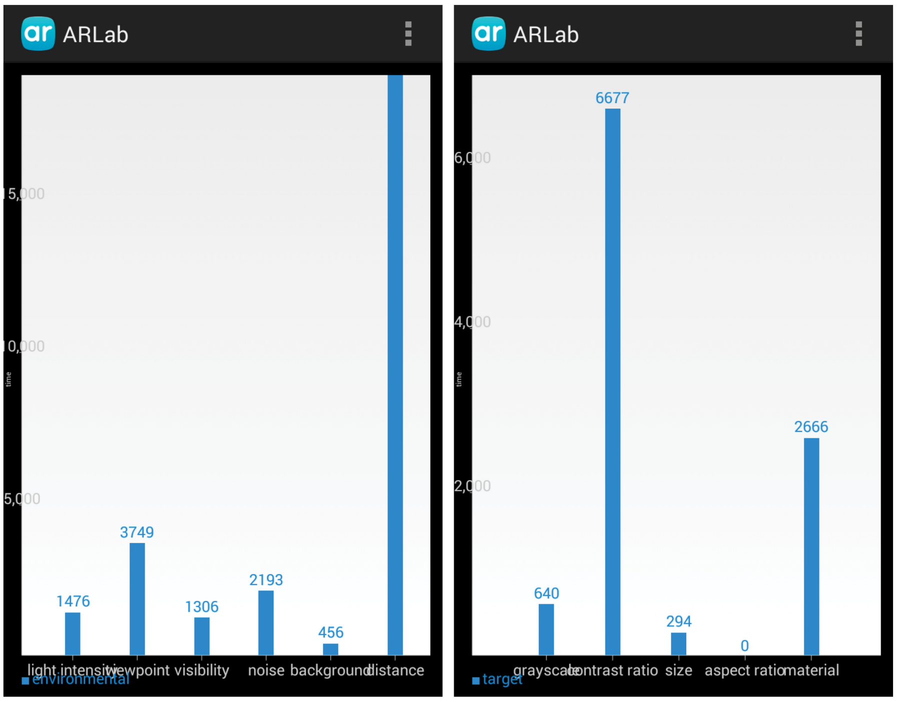


Figure 6.44.: ARLab results

ARLab

As it can be observed from Figure 6.44, ARLab has its fastest testing time detecting 10cm target images. Under one second are also found grayscale targets as well as different printed materials. The lowest times are stored under the environmental distance criterion.

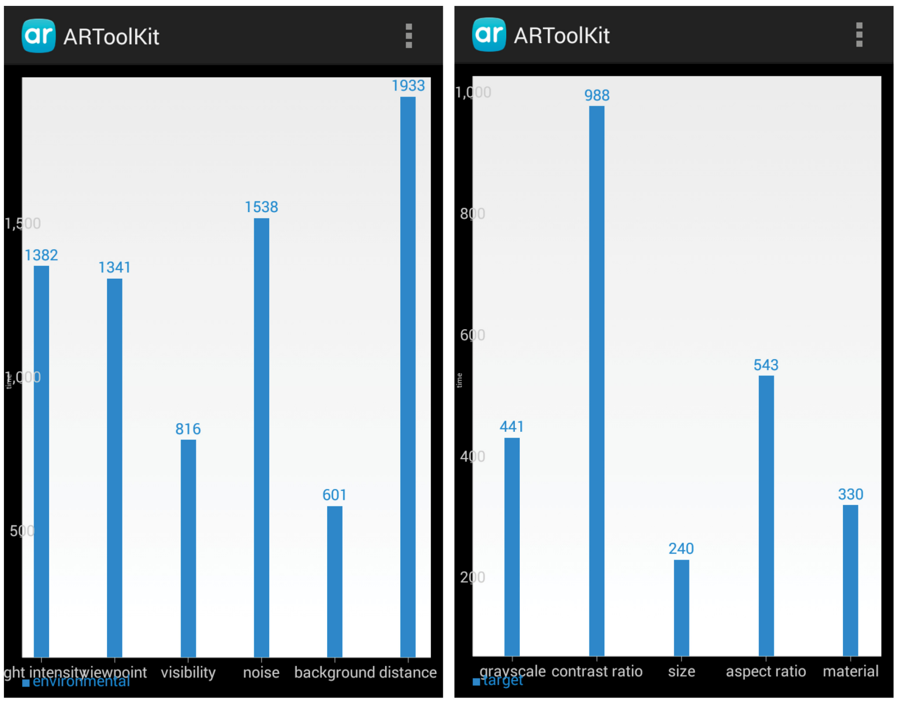


Figure 6.45.: ARToolKit results

ARToolKit

The ARToolKit has several testing times under one second, as depicted in Figure 6.45. The fastest time is detected on 20cm target images. Greyscale images and different printed materials also score under half a second. The worst testing time is recorded under distance.

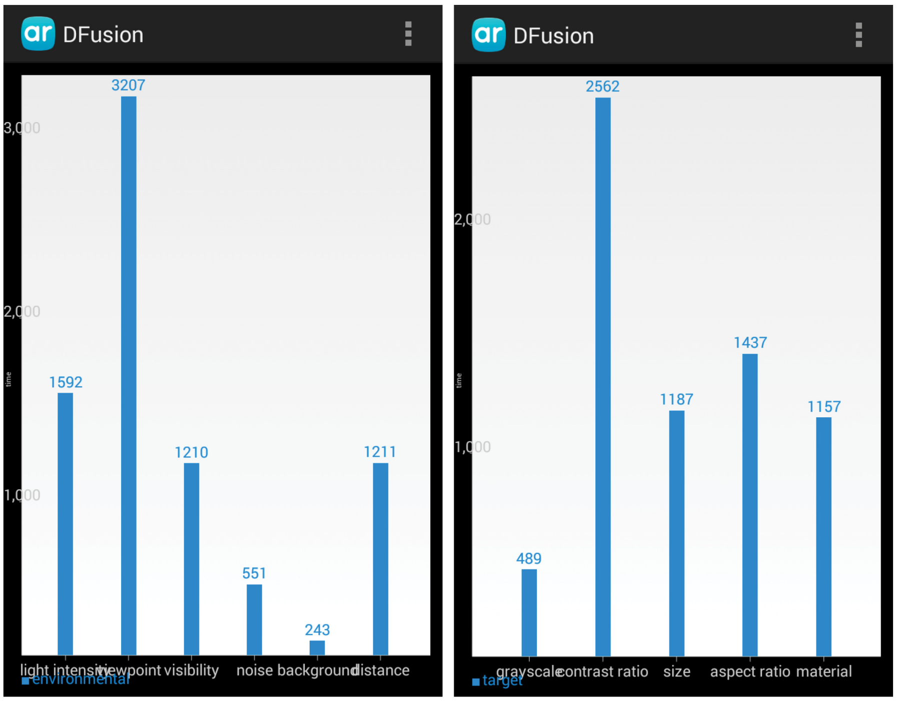


Figure 6.46.: D'Fusion results

D'Fusion

D'Fusion reveals its fastest time under the background criterion. An equal time is spent to detect target images both on bright and dark backgrounds. Under a second can be found the greyscale target image and the noise criterion. All noises from 10% to 70% are overcome. The longest testing time is stored for viewpoint criteria, all supported.

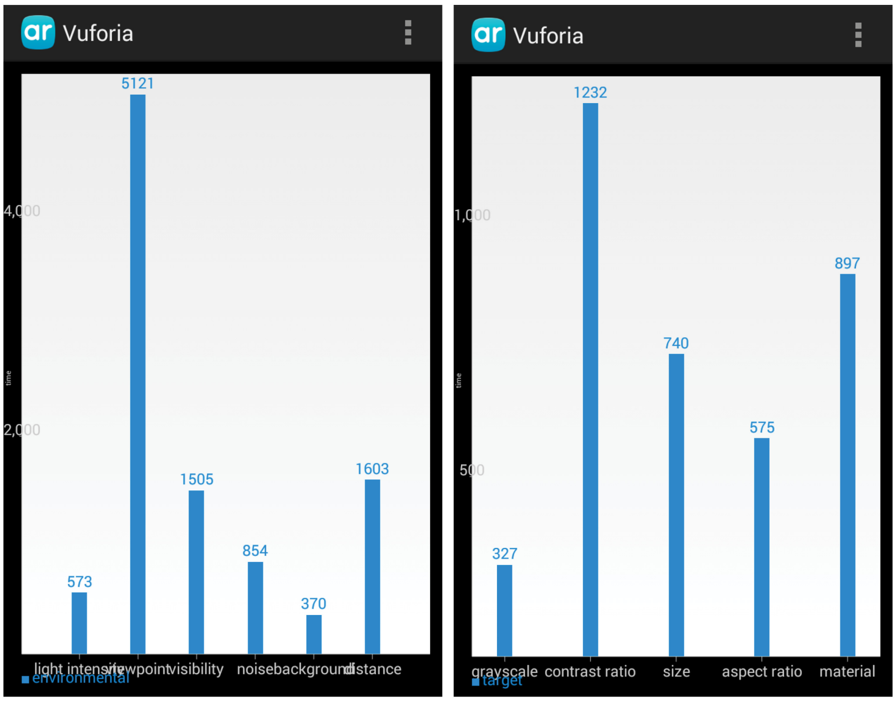


Figure 6.47.: Vuforia results

Vuforia

Greyscale target images and contrast backgrounds are found under half a second. Very good timings are registered by light intensity and printed material as well. Different viewpoints seem to take the a little longer to detect, all between one and two seconds.

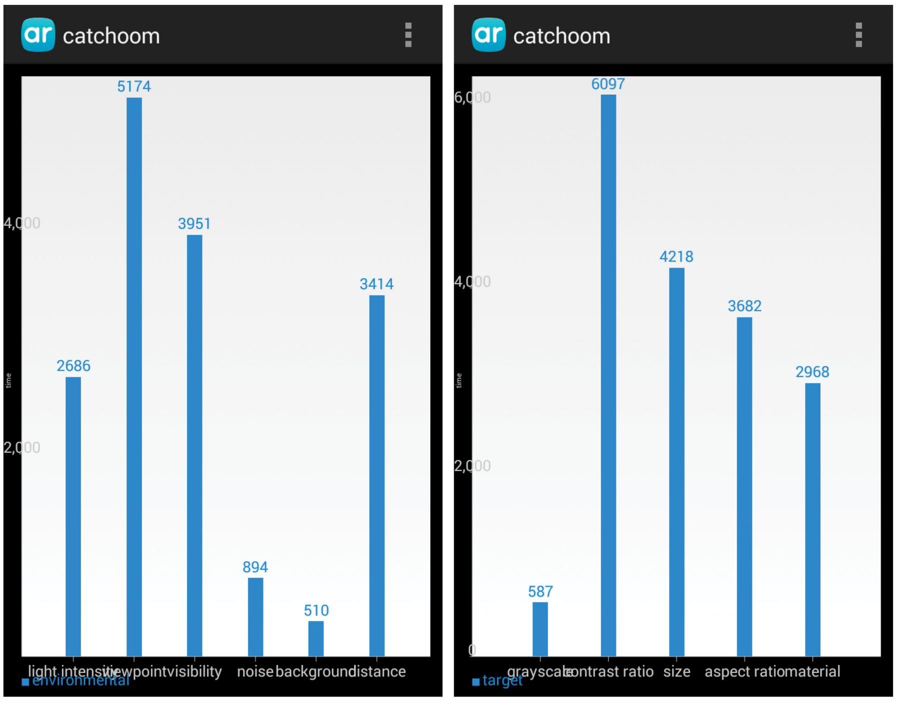


Figure 6.48.: Catchoom results

Catchoom

Catchoom supports greyscale images and contrast backgrounds a little over half a second. The only two other criteria under a second are noise and different printed material. All noise till 90% included are detected. All other criteria are over two seconds and the worst saved testing time is found under the contrast ratio criterion.

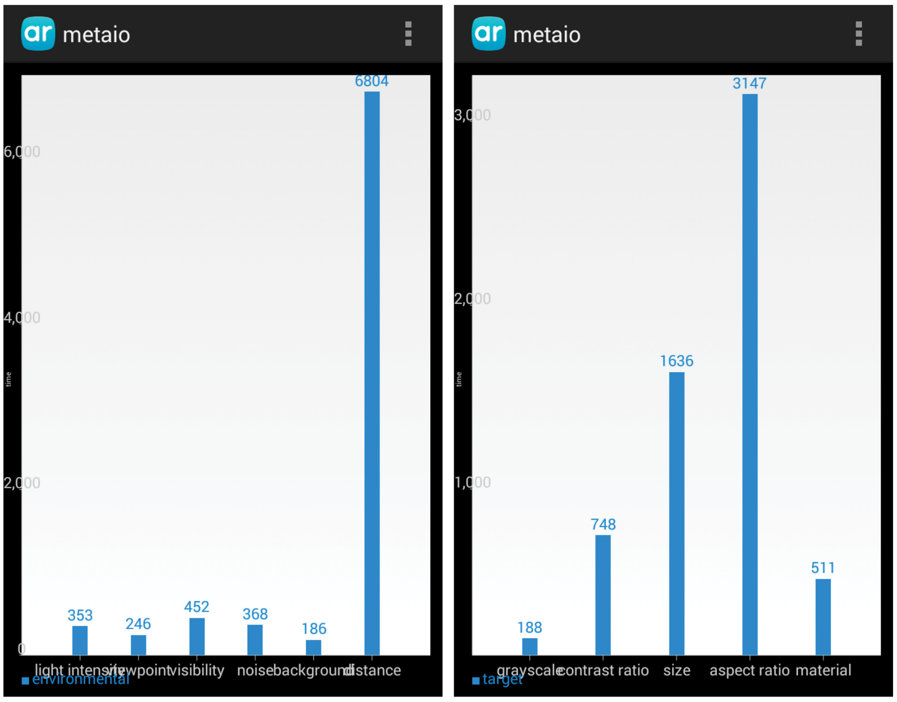


Figure 6.49.: Metaio results

Metaio

Metaio detects most of the criteria under a second. Nevertheless, cases exist when it is outperformed. Contrast backgrounds and grayscale targets are the fastest to recognize, followed closely by different types of light intensity and noise. Noise levels up to including 70% are detected. The aspect ratio constraint needs over 3 seconds on average to detect a digitally modified target. The highest testing time is recorded by distance as the framework needs over 21 seconds to detect the target image from a distance of 240cm.

6.5. Summary of the results

A number of different constraints have been tested on the evaluated frameworks. These constraints raise difficulties for some frameworks while others overcome them easily. Some observations noted during testing and a summary of the findings can be found in what follows. For example, it has been observed that target images are easier to be recognized in darker backgrounds by most of the frameworks than on brighter contrasts.

Scenarios

Each framework has its weaknesses and strengths. Although metaio has very good testing times, it often falls behind because it does not support fast moves. The tracking is lost or most of the times, the virtual content is displaced. Catchoom performs well under environmental criteria but does not adapt well to changes performed on the target image.

ARLab is not as strong as catchoom when it comes to target criteria and even more, it could not be determined for which of the performance criteria it could be rewarded. The literature also did not provide enough information for determining which of the usability criteria are supported. ARToolKit has the worst overall performance. Not to mention the difficulty to test it, the framework crashed very often.

Looking only at the rankings given to the frameworks in the different scenarios, Vuforia scores better than D'Fusion. The only case D'Fusion is on the first place is in the mCommerce scenario. The reason is that D'Fusion supports face tracking, an important requirement for the specific scenario, while Vuforia does not. Through out the chapter it was pointed out that the two are always in a competition, more often than once having very close testing times. Both frameworks are very well documented and easy to use.

Findings

Testing the frameworks on the default size target from various distances brought some interesting findings. Up close, 10cm away from the target image, the target is in most cases not completely visible. However, ARToolKit and ARLab recognized the targets. This is surprising because ARToolKit needs a 80% visible area of the target, while Vuforia only requires 10% and still could not recognize the target up close. From these findings it is concluded that the size of the target image and the distance to it are not strongly correlated.

The light intensity tests revealed a weak ARToolKit tracker when dealing with sudden change in light conditions or a semidark environment. Each of the considered four sequences for testing light intensities instigate environmental issues. In each case four out of the six frameworks show signs of extra work.

Another surprising result was registered by Vuforia which can detect a target image with a 200% level of noise. Most of the frameworks can still overcome a 70%, some even 90%, but 200% is an achievement.

A 40% visible area of the target is a reasonable constraint for almost every framework. Nevertheless, some frameworks work with even less visibility. Metaio produced an interesting outcome: if the entire target image is visible within a camera frame, but a percentage of it is obscured, metaio will not recognize the target. If however, only the visible area is within the camera frame and the obscured part not, metaio will recognize the target from a 40% visible area up.

Some of the frameworks come with the advice not to modify the target's aspect ratio. For this purpose it was tested what would be the effect of "shrinking" the target image. As a result, it is concluded that metaio is the only framework robust to various aspect ratios and D'Fusion exhibits high adaptability.

Angles of 45 ° are generally outrun. However, issues are raised for ARToolKit. Four different target sizes have also been tested and it was determined that not all frameworks support different size varieties. Tests with various contrast ratios and grayscale targets have been successfully passed.

7. Conclusion

Each Augmented Reality tool has its week and strong points. It cannot be concluded that one framework is better than another. The results presented in Chapter 6 prove just that. In some circumstances, given a set of constraints, a framework can outperform others. Take the example of the Bus Shelter scenario, by adding up the weights, considering the difficulty of some criteria, Vuforia is the obvious choice. If, on the other hand, a framework is searched for an indoor scenario, as for example the Magazine app, metaio would be on top of the list.

For the implementation part, it has been particularly hard in the case of some frameworks to determine the time from the point the tracker started and until the target image is recognized. Especially problematic are the native source projects, being in some cases to difficult to follow the thread back to the tracker. For this reason the times were not considered in the scoring process of the frameworks.

Future work

For future work, multi-targets, face tracking and text detection can be tested. The technology will keep evolving, great steps are being taken today towards the recognition of an object and thus making target images obsolete. Until then, markerless tracking is still the most popular and fastest way to augment the world around us.

The Test app allows the possibility for new criteria to added to the mix as well as the definition of new scenarios. The code could be modified and other frameworks could be added in the future. The app itself can be further developed to become a powerful tool of evaluation. During the research, implementation and testing phases it was clear that the surface has been barely scratched. Other actions can augment the app, like:

- switch between the rear and front cameras
- use flash
- switch between markerless tracking, face tracking and text detection
- switch between target image databases
- set camera preferences

A more advanced scoring system could be developed to also consider the best timings. Moreover, technical criteria could be introduced in the reply to how much CPU power is needed to run the tracker or how fast the battery is consumed? Does the framework suspend other apps? How much memory is used?

A. Annex

A.1. Licensing Models

Open Source Licenses

Open Source Licenses represent the GNU General Public License v2 or v3 terms for downloading the product for non-commercial use. Applications under the GPL license can be freely distributed with the condition that the source code is also freely available. If the source code is not freely available, a commercial license must be acquired instead.

Commercial Licenses

Commercial licenses must be purchased for closed source applications. The app developed using the licensed software can be marketed without any restrictions. More options are available:

1. **App License** allows the development of one unique app without any time constraint.
2. **Developer License** allows a company to develop more than one app during the license period. The license can be annually renewed.
3. **Academic License** allows academic institutions and both staff and students to use the software for teaching and research purposes only. The software is offered at deeply discounted prices or even free depending on the provider company.
4. **Demo License** allows the testing of the software for free on predefined sample projects.
5. **Watermark License** allows the free use of the software for creating new apps while displaying the software's watermark on the end product.

A.2. D'Fusion Configurations

The next screenshots are taken from D'Fusion's documentation [Imm12]. The steps and configurations to create a D'Fusion Studio project are detailed below. Additionally, the steps to deploy the project into an Android application are provided by the documentation.

5 Export and Test key Generation

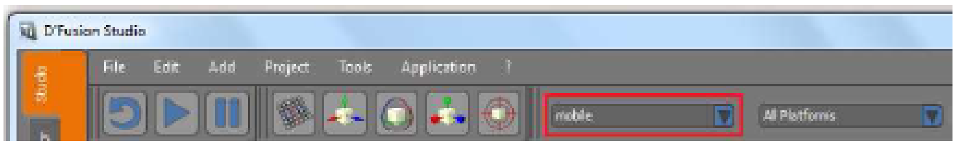
5.1 Export with D'Fusion Studio

A Test Software Key is automatically generated when your D'Fusion Studio project is exported. This Test Software Key will allow you to test your AR scenario when deploying your mobile application.

This section only explains how to export your D'Fusion Studio project. Deployment is explained in section 6.

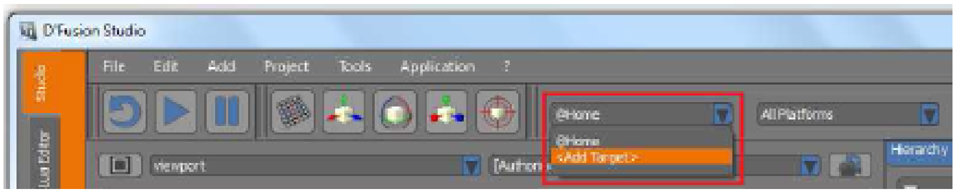
5.1.1 Choose the correct Target/Platform

First of all, make sure that you're working with the "mobile" target selected:



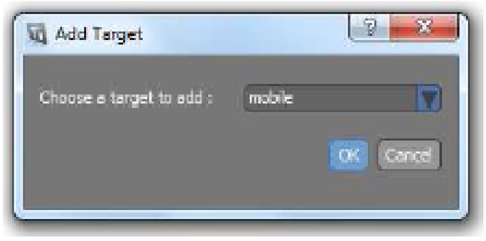
mobile | All Platforms

If this is not the case and there is no "mobile" target registered in your D'Fusion Studio project yet, just add it by choosing "<Add Target>" in the left-hand combo-box:



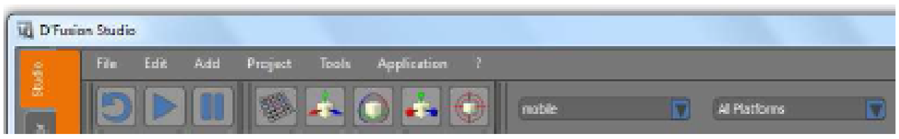
@Home | All Platforms
@Home
<Add Target>

In the window that pops up, choose "mobile" (default selection). You can copy the settings from another existing target configuration, but it's not mandatory:

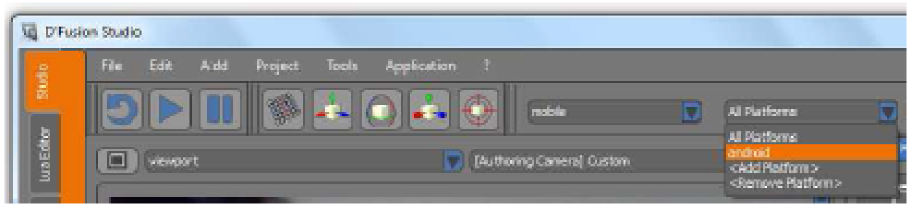


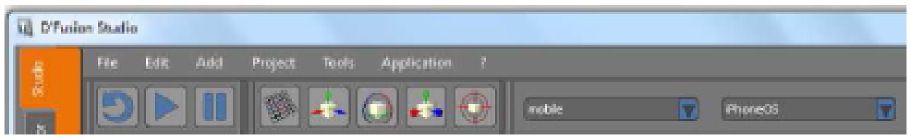
Choose a target to add : mobile

Then, in the second combo-box, select the correct platform you're working on – either "android" or "iPhoneOS"; or possibly choose the "All Platforms" wildcard token:

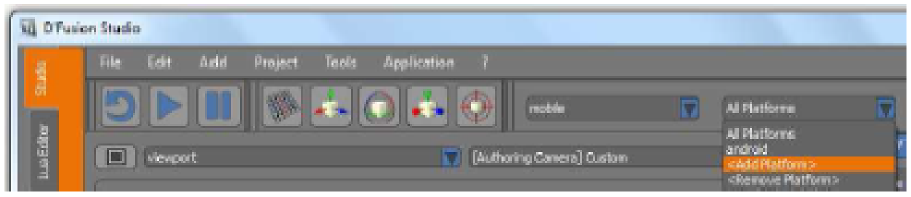


mobile | All Platforms

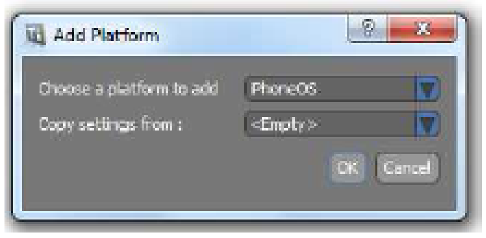




If the platform you actually need to use is not mentioned there (yet), just add it by choosing "<Add Platform>" command in the right-hand combo-box:



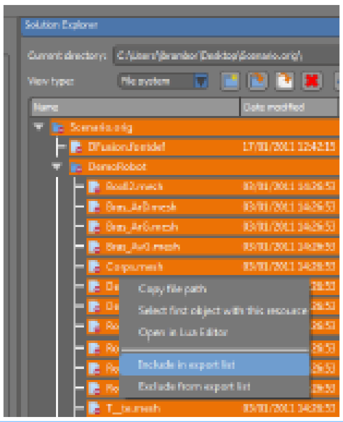
Then, in the dialog that pops up, choose the platform you want to add. You can copy settings form any existing mobile configuration, but it's not mandatory:



Choose a platform to add: iPhoneOS
Copy settings from : <Empty>

5.1.2 Include files for export

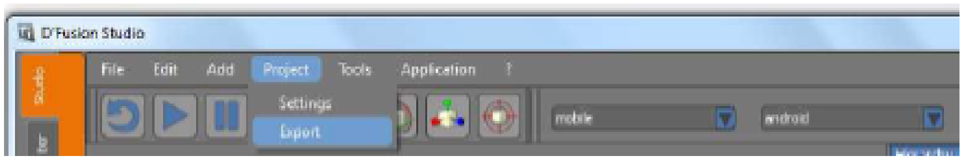
Include files for export (if it's not already the case). To do this, select your project's files that you want to export, make a right-click – a context menu will appear – then, select the "Include in export list" command, as:



Note that the "Include in export list" is an important step: only files selected for export for the actual Target/Platform will be processed and covered by the Test Software Key.

5.1.3 Export your project

To export your project, select "Export" under the "Project" menu:

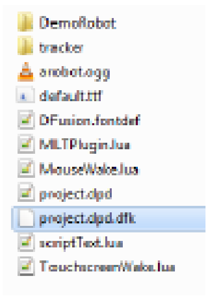


Make sure you specify the correct "**Application ID**":

- For an Apple application, the application ID appears in the "target info" window of your XCode project. Please refer to section 6.1.4.1 "Getting the Application ID (AppID)" for more information.
- For an Android application, the application ID is the package name mentioned in your application's "AndroidManifest.xml" file. See section 6.2.4 "Configure the Manifest file".

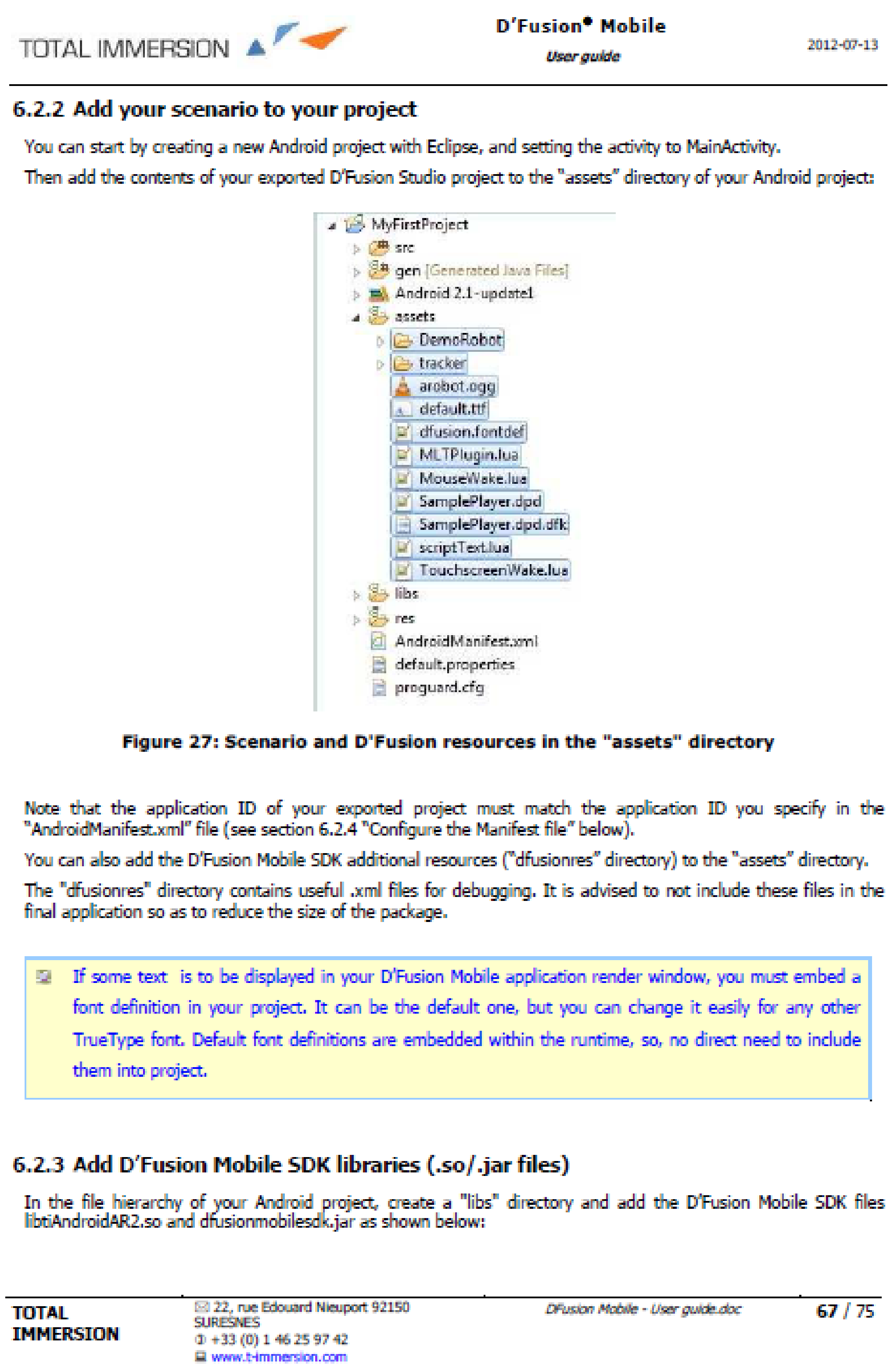
You also need to specify a destination (empty) folder for the export. This will be your "exported project folder".

When the project is exported, a Test Software Key is automatically generate. You can check the presence of this Test Software Key ("*.dpd.dfk" file) in your "exported project folder", among all your exported resources:



```
DemoRobot
tracker
arobot.ogg
default.ttf
DFusion.fontdef
MLTPlugin.lua
MouseWake.lua
project.dpd
project.dpd.dfk
scriptText.lua
TouchscreenWake.lua
```

Now, you can embed this exported scenario into your mobile project (Android only in this example).



6.2.2 Add your scenario to your project

You can start by creating a new Android project with Eclipse, and setting the activity to MainActivity.

Then add the contents of your exported D'Fusion Studio project to the "assets" directory of your Android project:

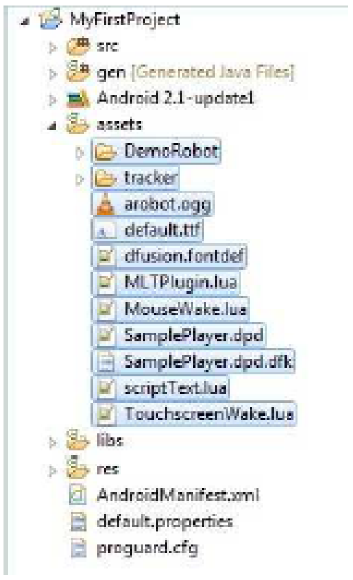


Figure 27: Scenario and D'Fusion resources in the "assets" directory

Note that the application ID of your exported project must match the application ID you specify in the "AndroidManifest.xml" file (see section 6.2.4 "Configure the Manifest file" below).

You can also add the D'Fusion Mobile SDK additional resources ("dfusionres" directory) to the "assets" directory.

The "dfusionres" directory contains useful .xml files for debugging. It is advised to not include these files in the final application so as to reduce the size of the package.

If some text is to be displayed in your D'Fusion Mobile application render window, you must embed a font definition in your project. It can be the default one, but you can change it easily for any other TrueType font. Default font definitions are embedded within the runtime, so, no direct need to include them into project.

6.2.3 Add D'Fusion Mobile SDK libraries (.so/.jar files)

In the file hierarchy of your Android project, create a "libs" directory and add the D'Fusion Mobile SDK files libtiAndroidAR2.so and dfusionmobilesdk.jar as shown below:

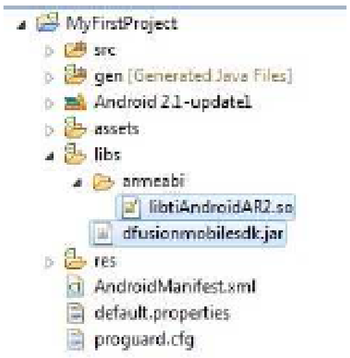


Figure 28: libtiAndroidAR2.so and dfusionmobilesdk.jar folder organization

In Eclipse, go into the "Package Explorer" and right-click on dfusionmobilesdk.jar: select "Build Path" > "Add to Build Path".

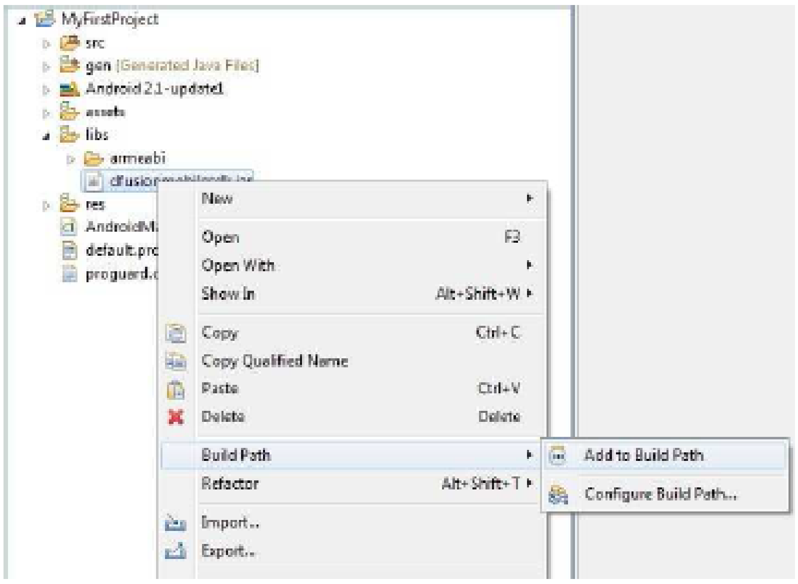
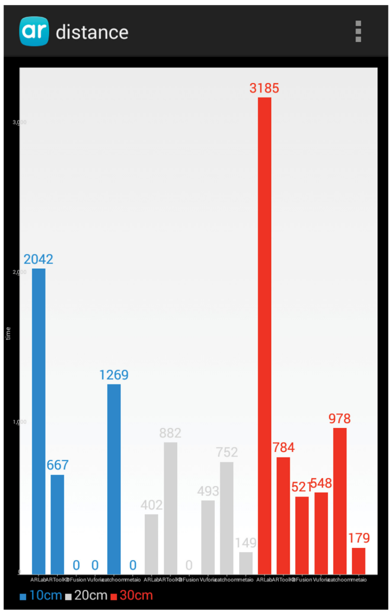
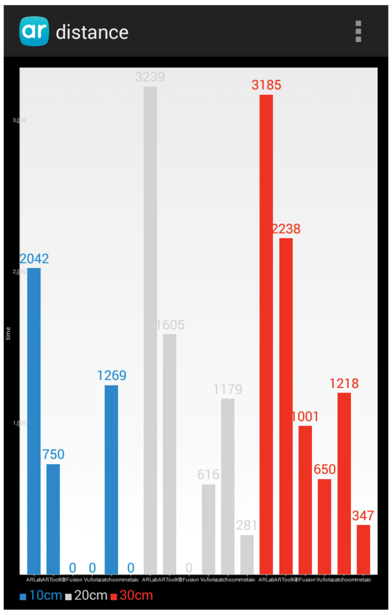


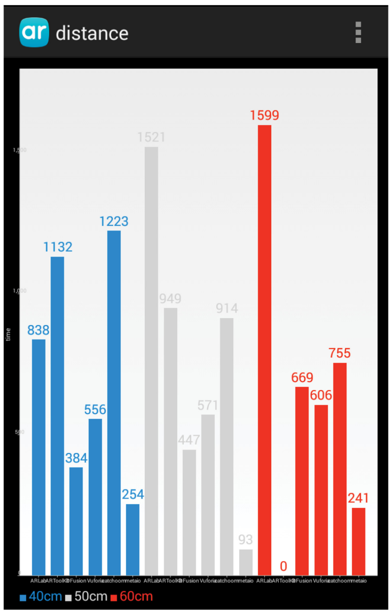
Figure 29: Add dfusionmobilesdk.jar to the build path

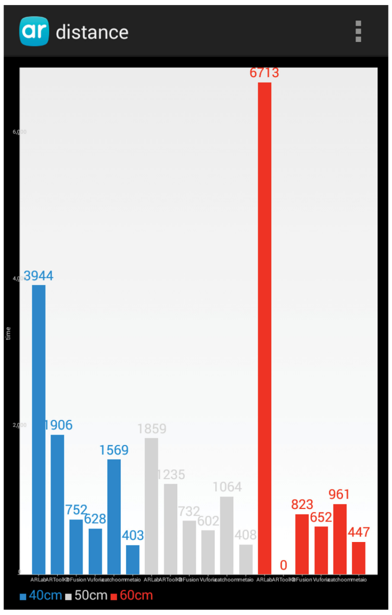
A.3. Detailed Results

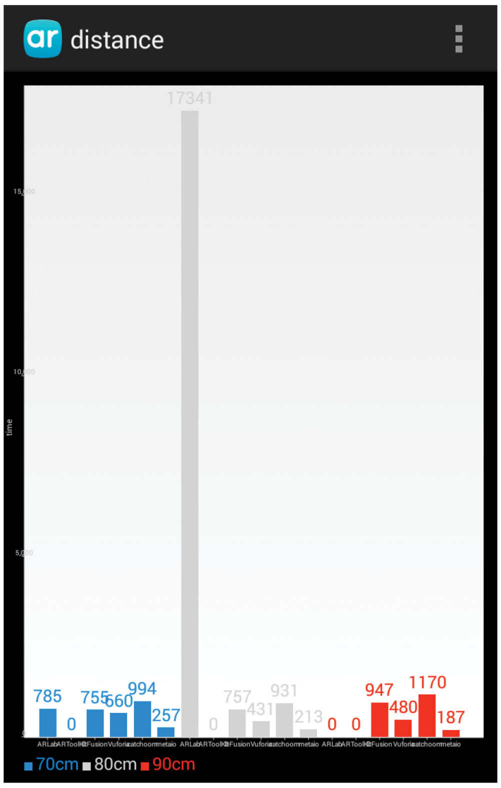
The following graphs represent the best testing times per criterion registered by each framework. Exception from the rule are the distance criterion graphs which display on the left side the average testing times to be compared to the best times on the right.

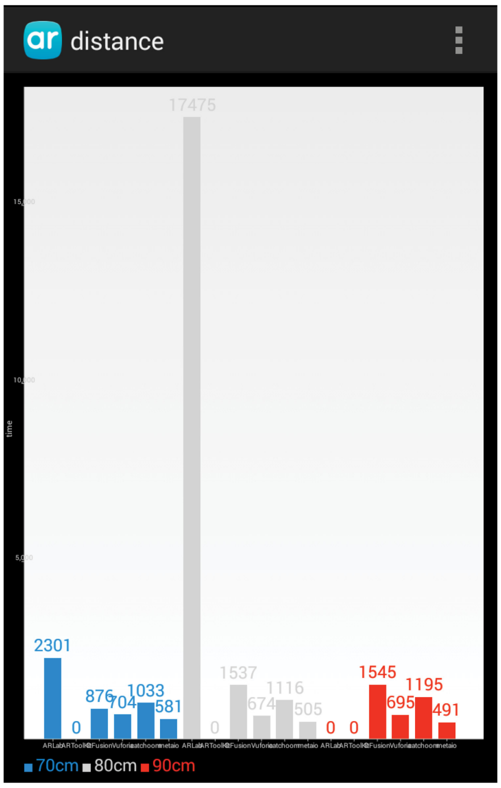


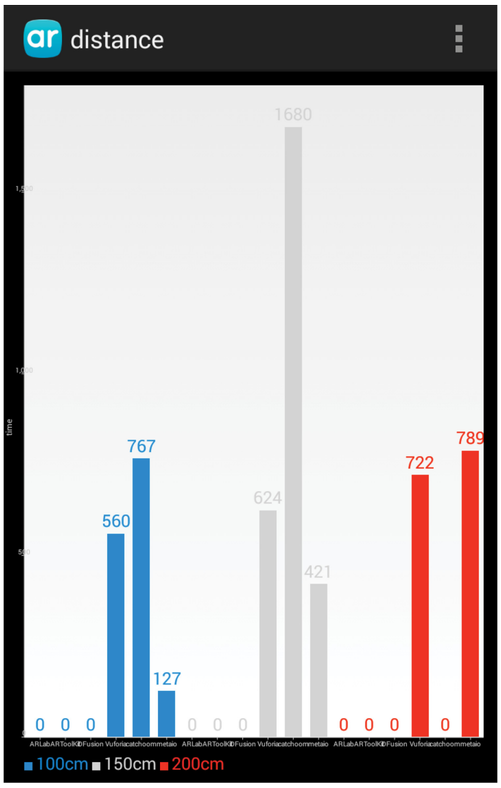


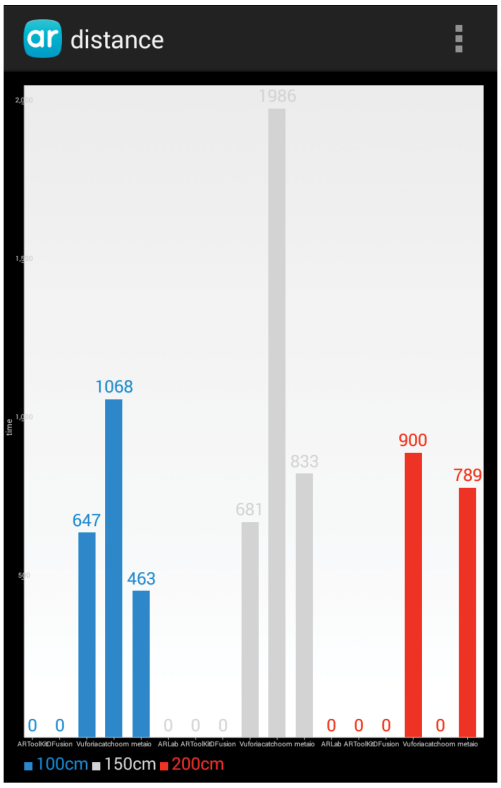


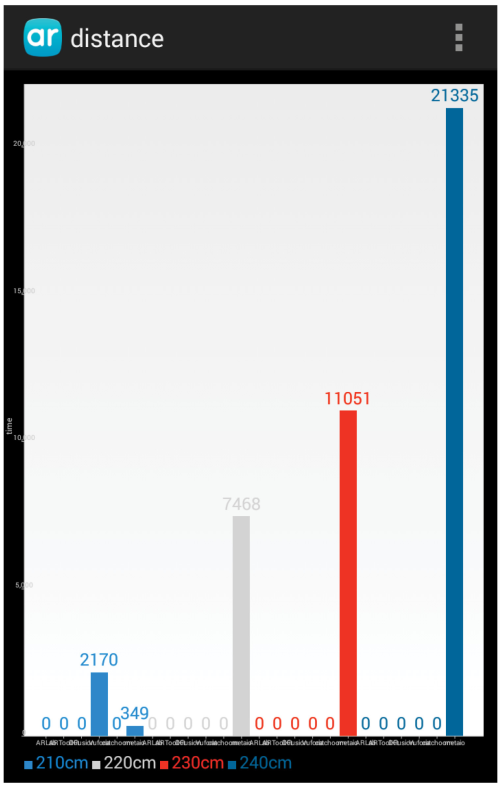


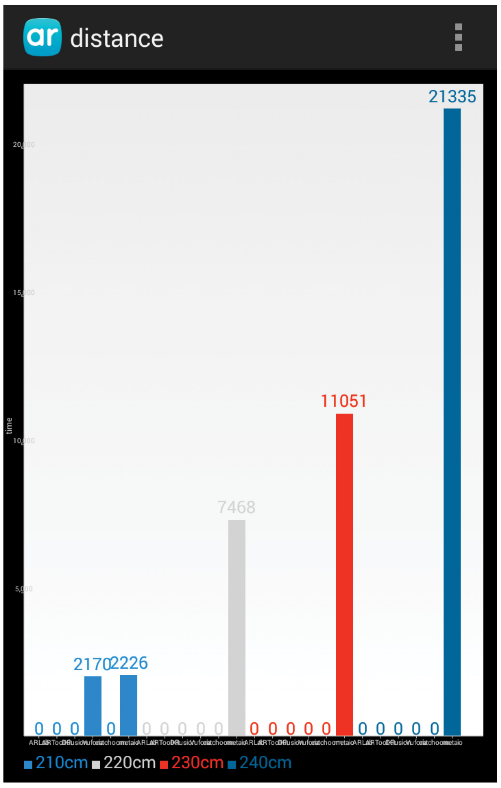

distance
17341
785
0
755
660
994
257
0
757
431
931
213
0
0
947
480
1170
187
time
70cm 80cm 90cm

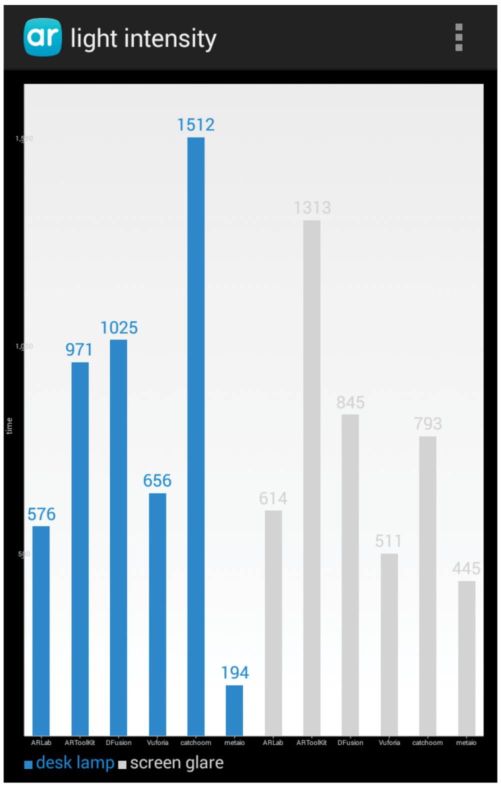

distance
17475
2301
0
876
704
1033
581
0
1537
674
1116
505
0
0
1545
695
1195
491
time
70cm 80cm 90cm

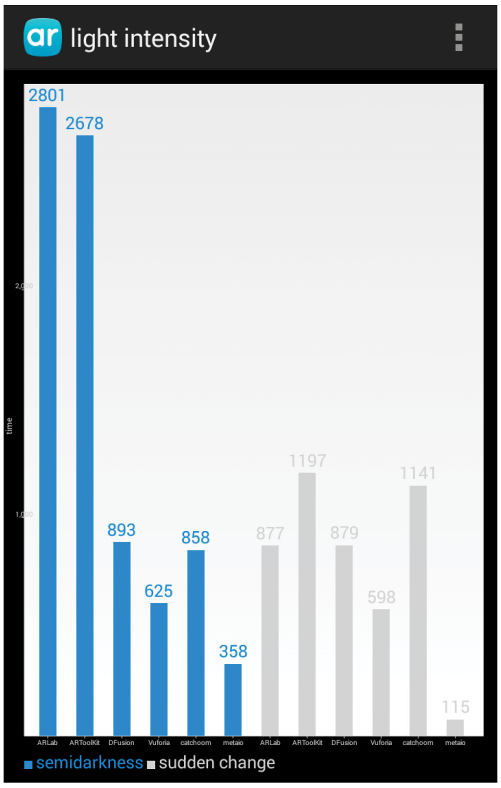

distance
1680
0
0
0
560
767
127
0
0
0
624
421
0
0
0
722
0
789
time
100cm 150cm 200cm

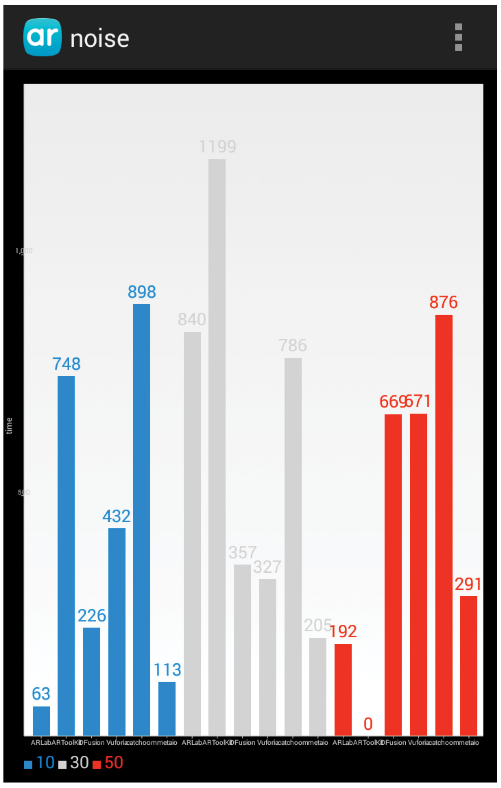

distance
1986
0
0
647
1068
463
0
0
0
681
833
0
0
0
900
0
789
time
100cm 150cm 200cm

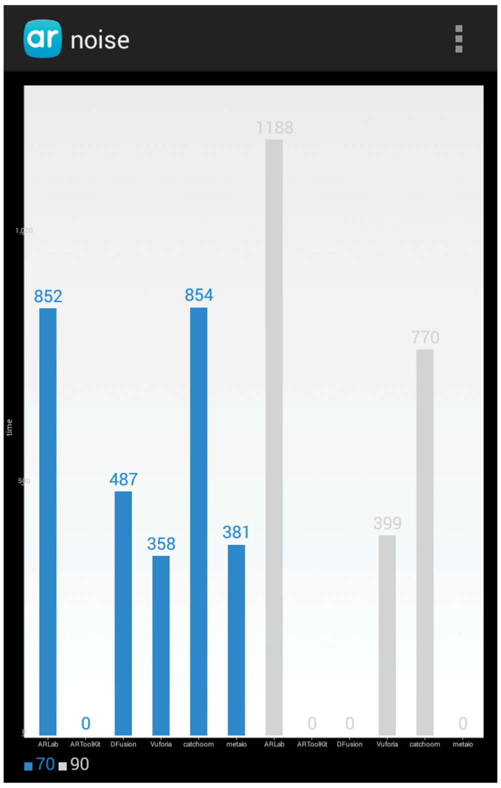

distance
time
21335
11051
7468
2170
349
0 0 0
0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
■ 210cm ■ 220cm ■ 230cm ■ 240cm

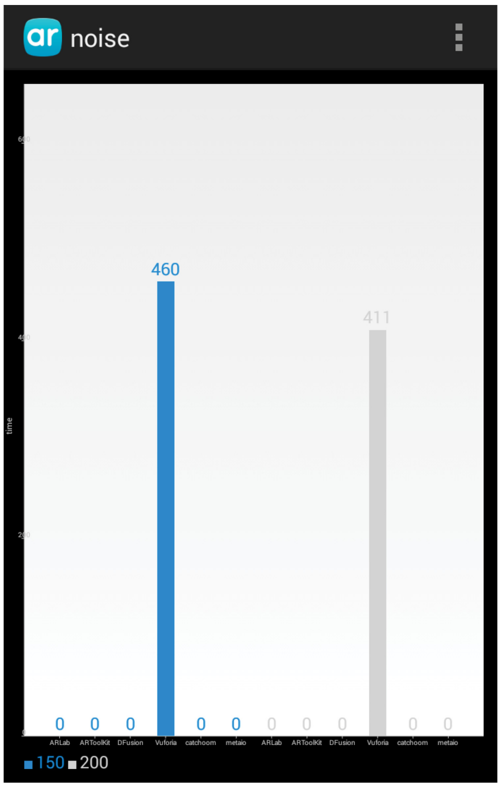

distance
time
21335
11051
7468
2170
2226
0 0 0
0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
■ 210cm ■ 220cm ■ 230cm ■ 240cm

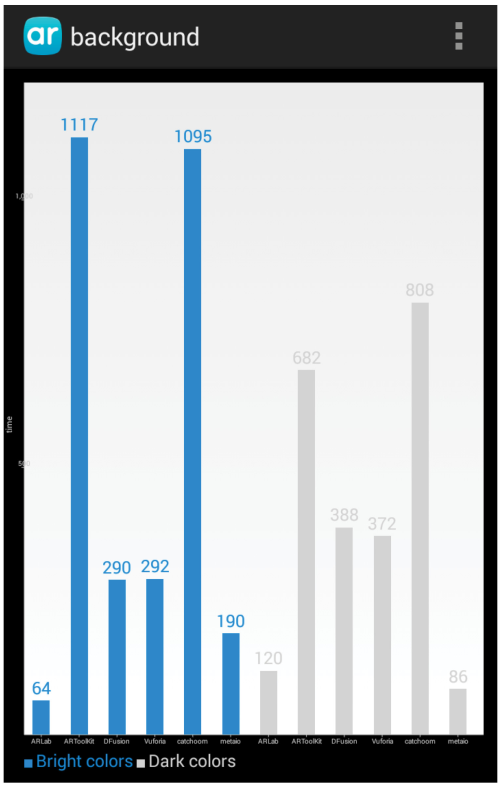

light intensity
time
576
971
1025
656
1512
194
614
1313
845
511
793
445
ARLab
ARToolKit
DFusion
Vuforia
catchoom
metaio
ARLab
ARToolKit
DFusion
Vuforia
catchoom
metaio
■ desk lamp ■ screen glare

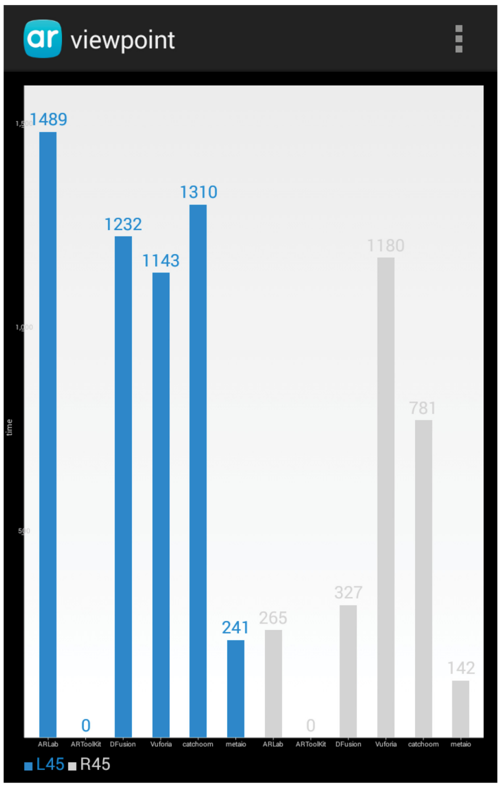

light intensity
time
2801
2678
893
625
858
358
877
1197
879
598
1141
115
ARLab
ARToolKit
DFusion
Vuforia
catchoom
metaio
ARLab
ARToolKit
DFusion
Vuforia
catchoom
metaio
■ semidarkness ■ sudden change

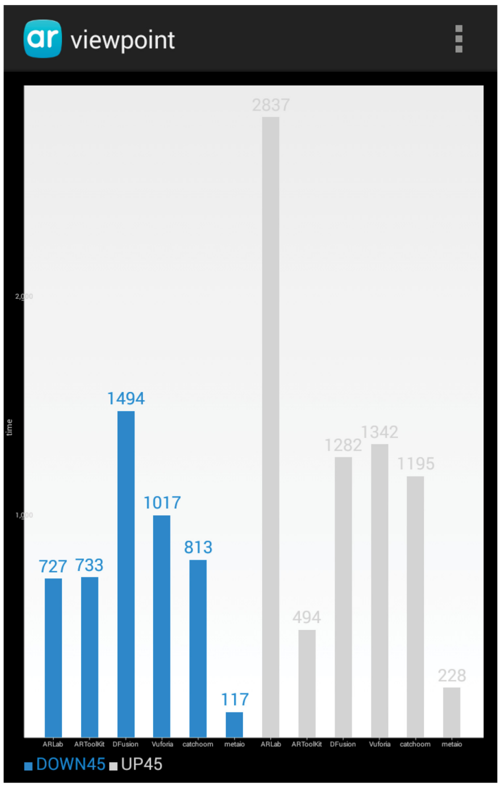

noise
time
63
748
226
432
898
113
840
1199
357
327
786
205
192
0
669
671
876
291
ARLab ARToolKit DFusion Vuforia catchoom metaio ARLab ARToolKit DFusion Vuforia catchoom metaio ARLab ARToolKit DFusion Vuforia catchoom metaio
10 30 50

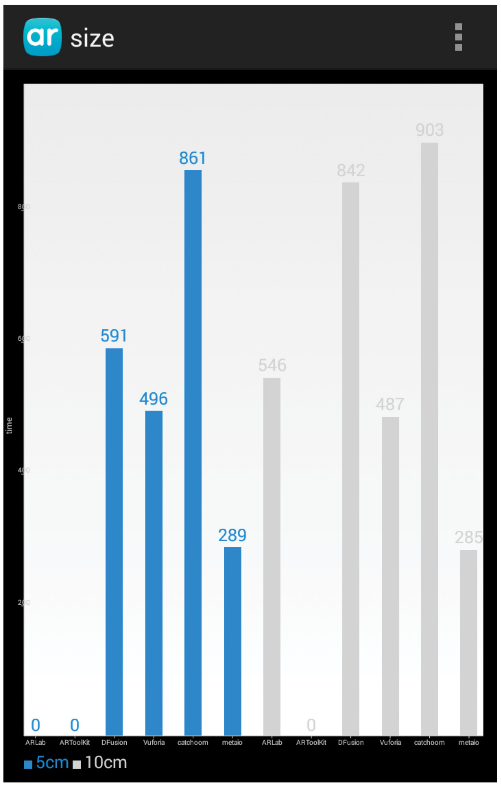

noise
time
852
0
487
358
854
381
1188
0
0
399
770
0
ARLab ARToolKit DFusion Vuforia catchoom metaio ARLab ARToolKit DFusion Vuforia catchoom metaio
70 90

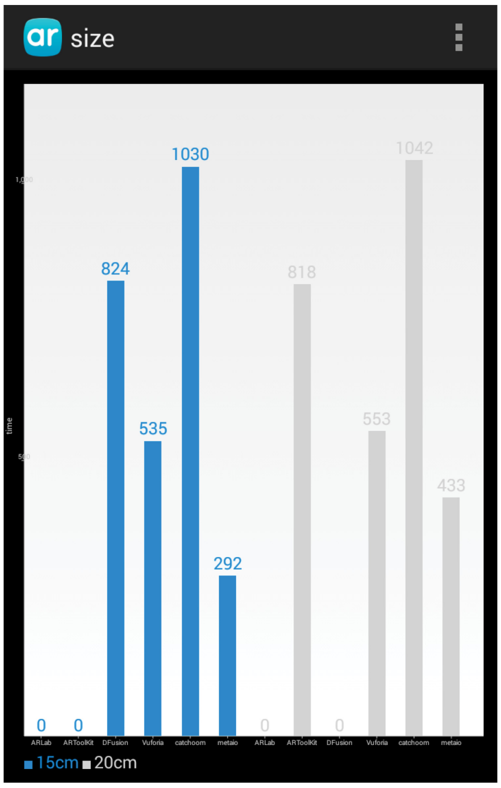

noise
time
0
0
0
460
0
0
0
0
0
411
0
0
ARLab ARToolKit DFusion Vuforia catchoom metaio ARLab ARToolKit DFusion Vuforia catchoom metaio
150 200

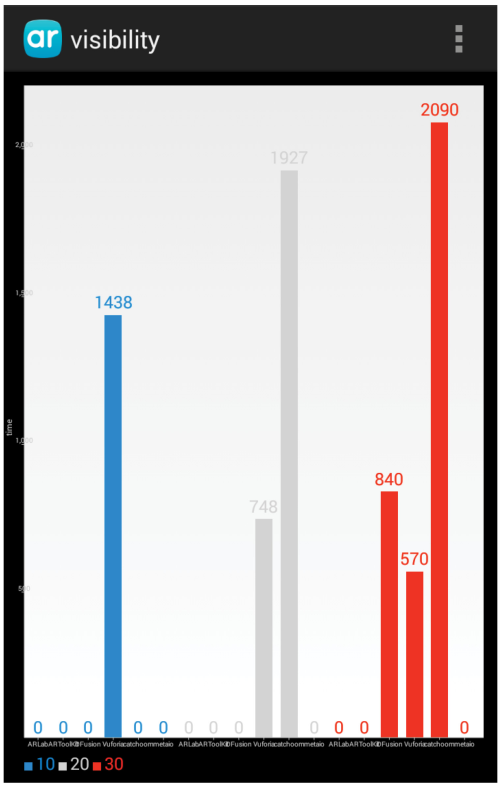

background
time
64
1117
290
292
1095
190
120
682
388
372
808
86
ARLab ARToolKit DFusion Vuforia catchoom metaio ARLab ARToolKit DFusion Vuforia catchoom metaio
Bright colors Dark colors

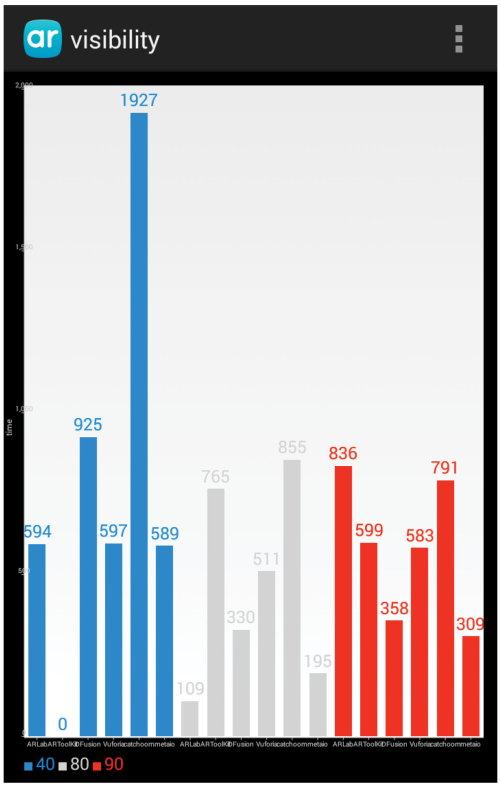

ar viewpoint
time
1489
0
1232
1143
1310
241
265
0
327
1180
781
142
ARLab
ARToolKit
DFusion
Vuforia
catchoom
metaio
ARLab
ARToolKit
DFusion
Vuforia
catchoom
metaio
■ L45 ■ R45

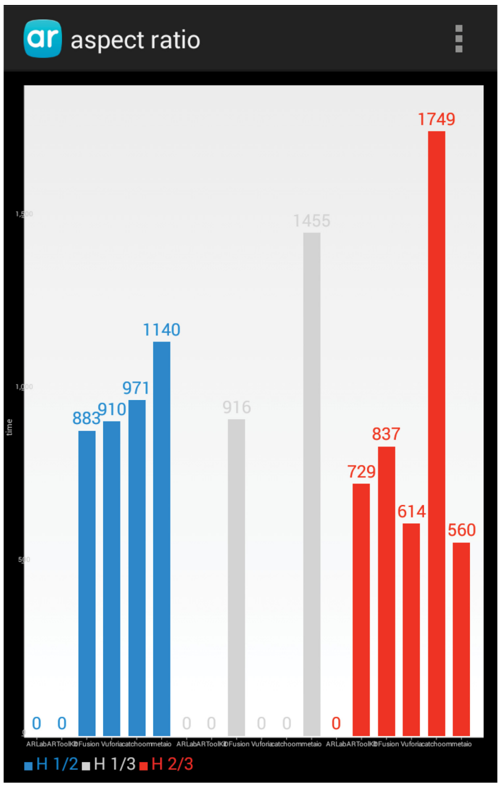

ar viewpoint
time
727
733
1494
1017
813
117
2837
494
1282
1342
1195
228
ARLab
ARToolKit
DFusion
Vuforia
catchoom
metaio
ARLab
ARToolKit
DFusion
Vuforia
catchoom
metaio
■ DOWN45 ■ UP45

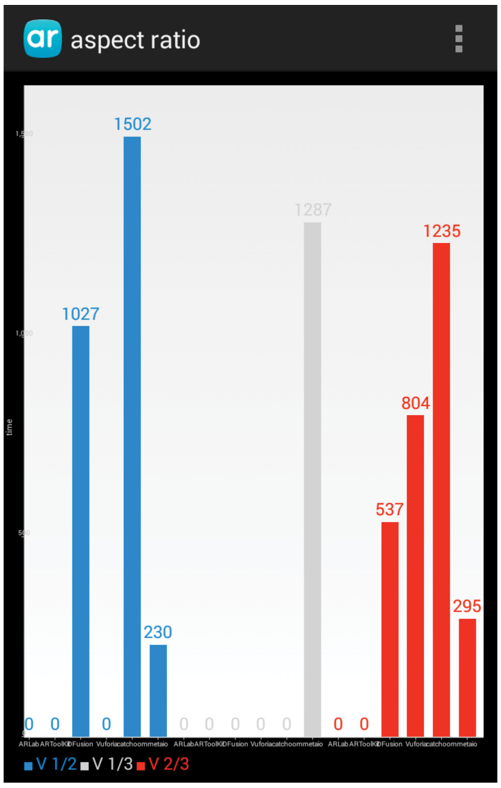

ar size
time
0
0
591
496
861
289
546
0
842
487
903
285
ARLab
ARToolKit
DFusion
Vuforia
catchoom
metaio
ARLab
ARToolKit
DFusion
Vuforia
catchoom
metaio
■ 5cm ■ 10cm

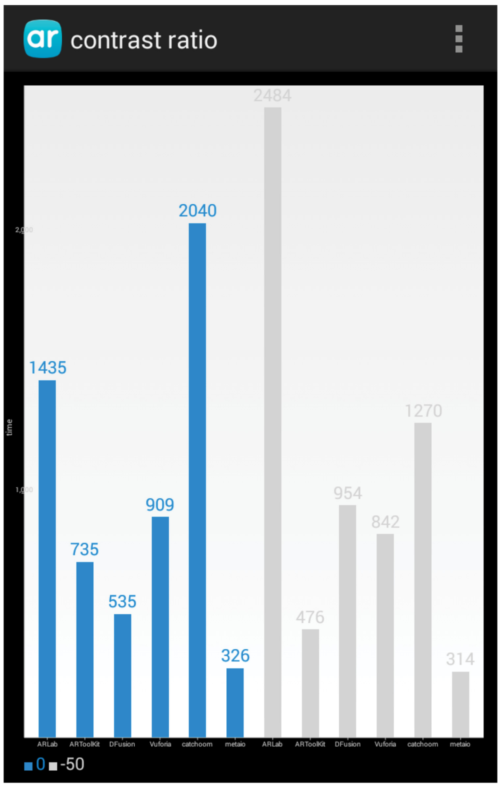

ar size
time
0
0
824
535
1030
292
0
818
0
553
1042
433
ARLab
ARToolKit
DFusion
Vuforia
catchoom
metaio
ARLab
ARToolKit
DFusion
Vuforia
catchoom
metaio
■ 15cm ■ 20cm

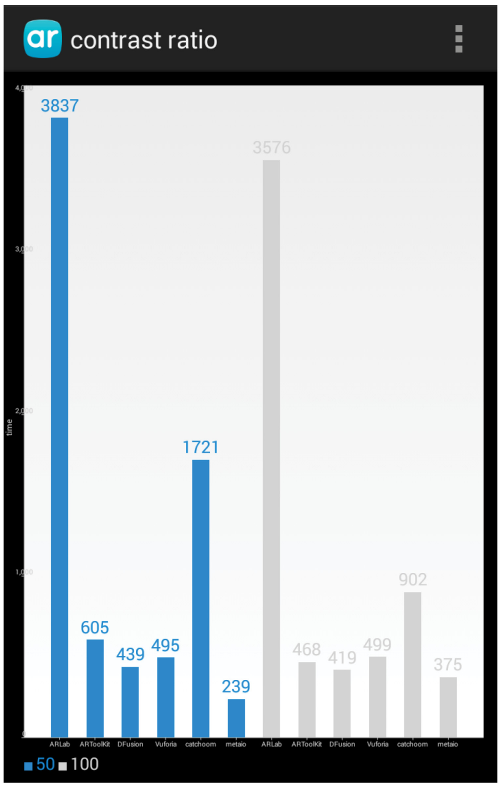

visibility
time
2,000
1,500
1,000
500
0 0 0 1438 0 0 0 0 0 748 1927 0 0 0 840 570 2090 0
ARLab ARToolKit D'Fusion Vuforia catchoom metaio ARLab ARToolKit D'Fusion Vuforia catchoom metaio ARLab ARToolKit D'Fusion Vuforia catchoom metaio
🟦 10 ⬜ 20 🟥 30

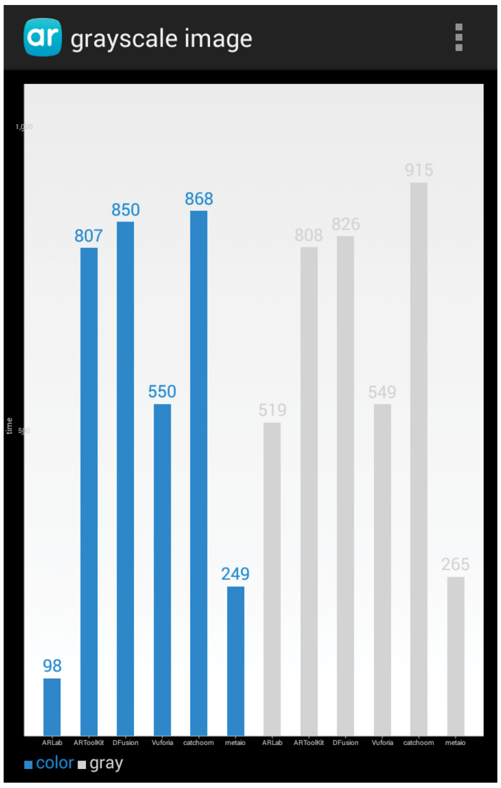

visibility
time
2,000
1,500
1,000
500
594 0 925 597 1927 589 109 765 330 511 855 195 836 599 358 583 791 309
ARLab ARToolKit D'Fusion Vuforia catchoom metaio ARLab ARToolKit D'Fusion Vuforia catchoom metaio ARLab ARToolKit D'Fusion Vuforia catchoom metaio
🟦 40 ⬜ 80 🟥 90

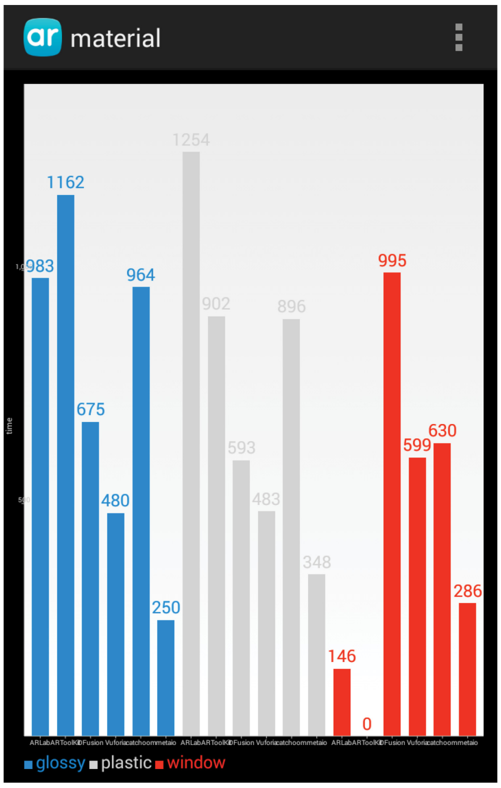

aspect ratio
time
1,500
1,000
500
0 0 883 910 971 1140 0 0 916 0 0 1455 0 729 837 614 1749 560
ARLab ARToolKit D'Fusion Vuforia catchoom metaio ARLab ARToolKit D'Fusion Vuforia catchoom metaio ARLab ARToolKit D'Fusion Vuforia catchoom metaio
🟦 H 1/2 ⬜ H 1/3 🟥 H 2/3


aspect ratio
time
1,500
1,000
500
0 0 1027 0 1502 230 0 0 0 0 0 1287 0 0 537 804 1235 295
ARLab ARToolKit D'Fusion Vuforia catchoom metaio ARLab ARToolKit D'Fusion Vuforia catchoom metaio ARLab ARToolKit D'Fusion Vuforia catchoom metaio
🟦 V 1/2 ⬜ V 1/3 🟥 V 2/3


contrast ratio
time
1435
735
535
909
2040
326
2484
476
954
842
1270
314
ARLab ARToolKit DFusion Vuforia catchoom metaio ARLab ARToolKit DFusion Vuforia catchoom metaio
■ 0 ■ -50


contrast ratio
time
3837
605
439
495
1721
239
3576
468
419
499
902
375
ARLab ARToolKit DFusion Vuforia catchoom metaio ARLab ARToolKit DFusion Vuforia catchoom metaio
■ 50 ■ 100


grayscale image
time
98
807
850
550
868
249
519
808
826
549
915
265
ARLab ARToolKit DFusion Vuforia catchoom metaio ARLab ARToolKit DFusion Vuforia catchoom metaio
■ color ■ gray


material
time
983
1162
675
480
964
250
1254
902
593
483
896
348
146
0
995
599
630
286
ARLab ARToolKit DFusion Vuforia catchoom metaio ARLab ARToolKit DFusion Vuforia catchoom metaio ARLab ARToolKit DFusion Vuforia catchoom metaio
■ glossy ■ plastic ■ window

B. List of Abbreviations

2D	2 Dimensional
3D	3 Dimensional
ADT	Android Development Tools
API	Application Programming Interface
app	Mobile application
CAD	Computer Aided Design
CPU	Central Processing Unit
CRUD	Create Rename Update Delete
CSS	Cascade Style Sheet
CVS	Continuous Visual Search
DOF	Degree of Freedom
EABI	Embedded-Application Binary Interface
fps	frames per second
GPL	General Public License
GPS	Global Positioning System
GPU	Graphics Processor Unit
HMD	Head Mounted Display
HTML	HyperText Markup Language
HUD	Heads Up Display
NDK	Native Development Kit
IP	Intellectual Property
PC	Personal Computer
PDA	Personal Digital Assistant
RFID	Radio Frequency Identification
SD	Secure Digital

SDK	Software Development Kit
SLAM	Simultaneous Localization and Tracking
SQL	Structured Query Language

Bibliography

[ARL14a] ARLab. Ar browser. 2014. http://arlab.com/arbroswer. 23

[ARL14b] ARLab. Arlab. company. 2014. http://arlab.com/company. 22

[ARL14c] ARLab. Image matching. 2014. http://arlab.com/imagematching. 24

[ARL14d] ARLab. Image tracking sdk. 2014. http://arlab.com/imagetracking. 23

[ARP13a] ARPA. Arpa glass sdk. 2013. http://arpa-solutions.net/en/ARPA_GLASS. 25

[ARP13b] ARPA. Arpa industry. 2013. http://arpa-solutions.net/en/ARPA_Industry. 25

[ARP13c] ARPA. Arpa plugin 2.0 for unity. 2013. http://arpa-solutions.net/en/ARPA_Plugin_Unity. 25

[ARP13d] ARPA. Arpa sdk 2.2. 2013. http://arpa-solutions.net/en/ARPA_SDK. 27

[ARP13e] ARPA. Why arpa solutions? 2013. http://arpa-solutions.net/en/Company. 25

[ART13a] ARToolworks. Artoolkit for android. 2013. http://www.artoolworks.com/products/mobile/artoolkit-for-android/. 30

[ART13b] ARToolworks. Artoolkit for android examples. 2013. http://www.artoolworks.com/support/library/ARToolKit_for_Android_examples. 30

[ART13c] ARToolworks. Desktop. 2013. http://www.artoolworks.com/products/desktop/. 29

[ART13d] ARToolworks. Web. 2013. http://www.artoolworks.com/products/web/. 29

[ART14] ARToolworks. Welcome to artoolworks. 2014. http://www.artoolworks.com/. 28

[Azu97a] Ronald T. Azuma. A survey of augmented reality. Technical report, Hughes Research Laboratories, 8 1997. http://www.cs.unc.edu/ azuma/ARpresence.pdf. 4

[Azu97b] Ronald T. Azuma. A survey of augmented reality. Technical report, Hughes Research Laboratories, 3011 Malibu Canyon Road, MS RL96, 1997. 48

[Bon14] Kevin Bonsor. How augmented reality works. March 2014. http://computer.howstuffworks.com/augmented-reality.htm. 4

[Cat14a] Catchoom. Catchar mobile sdk. 2014. http://catchoom.com/product/mobile-sdk/. 33

[Cat14b] Catchoom. Cloud image recognition. 2014. http://catchoom.com/product/cloud-image-recognition-api/. 31

[Cat14c] Catchoom. Craftar: The ultimate ar toolbox. 2014. http://catchoom.com/product/craftar-augmented-reality-image-recognition/#. 31

[Com13] Mind Commerce. Augmented reality in gaming and entertainment. pdf, March 2013. A market research report. 9

[Dav12] Nils Davis. Augmented reality sdk comparison. September 2012. http://socialcompare.com/en/comparison/augmented-reality-sdks. 19

[Dol12] Jose Dolz. Markerless augmented reality. May 2012. http://www.arlab.com/blog/markerless-augmented-reality/. 15

[Dom10] Tobias Domhan. Augmented reality: Wie superman auf dem handy fliegt. 2010. http://studium.dhbw-stuttgart.de/informatik/projekte/augmented-reality.html. 21

[Imm12] Total Immersion. D'fusion mobile user guide. pdf, July 2012. D'Fusion documentation. 130

[JY95] Alex Waibel Jie Yang. Tracking human faces in real-time. Technical report, Carnegie Mellon University, Pittsburg Pennsylvania 15213, 1995. 60

[KR12] Greg Kipper and Joseph Rampolla. *Augmented reality: an emerging technologies guide to AR*. Syngress, Waltham, MA, 2012. 6

[Lan13] Ben Lang. An introduction to positional tracking and degrees of freedom (dof). 2013. http://www.roadtovr.com/introduction-positional-tracking-degrees-freedom-dof/. 45

[Lim13] Infiniti Research Limited. Global augmented reality market 2014-2018. pdf, November 2013. A market research report. 1

[MAL11] Lawrance J. Rosenblum Mark A. Livingston. Military applications of augmented reality. pdf, 2011. http://www.nrl.navy.mil/itd/imda/sites/www.nrl.navy.mil.itd.imda/files/pdfs/2011-_Springer_MilitaryAR.pdf. 8

[Mar13a] MarketsandMarkets. Virtual reality & augmented reality market - by technology (mobile & spatial ar, semi & fully immersive vr), by sensors & components (accelerometer, data glove, hmd, ics), by applications (medical, military, gaming), by geography (2013-2018). pdf, July 2013. A market research report. 1, 8

[Mar13b] MarketsandMarkets. Virtual reality & augmented reality market worth 1.06 billion by 2018. July 2013. http://www.marketsandmarkets.com/PressReleases/ar-market.asp. 1

[MC] Mike Bailey Matthew Clothier. Overcoming augmented reality tracking difficulties in changing lighting conditions. pdf. 7

[ME13] Martin Schön Martin Ebner, Walther Nagler, editor. *"Architecture Students Hate Twitter and Love Dropbox" or Does the Field of Study Correlates With Web 2.0 Behavior?* Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications, 2013. 10

[Meg13] Florencia Megawati. Paper-based catalog integrated with augmented reality. Master's thesis, Binus University, Jakarta, 2013. 17

[Met12] Metaio. The metaio sdk. 2012. http://www.metaio.com/products/sdk/features/. 39

[Met14a] Metaio. 2014: Ar will be on every smartphone. 2014. http://www.metaio.com/company/. 37

[Met14b] Metaio. The arengine. 2014. http://www.metaio.com/products/arengine/overview/. 37

[Met14c] Metaio. Junaio. 2014. http://www.metaio.com/junaio/. 37

[Met14d] Metaio. Metaio. 2014. http://www.metaio.com. 37

[Met14e] Metaio. Metaio cloud. 2014. http://www.metaio.com/cloud/. 37

[Met14f] Metaio. The metaio creator. 2014. http://www.metaio.com/products/creator/. 37

[Met14g] Metaio. Metaio cvs. 2014. http://www.metaio.com/visual-search/. 37

[Met14h] Metaio. Metaio engineer. 2014. http://www.metaio.com/engineer/. 37

[nie13] nielsen. Ring the bells: more smartphones in students hands ahead of back-to-school season, October 2013. http://www.nielsen.com/us/en/newswire/2013/ring-the-bells-more-smartphones-in-students-hands-ahead-of-back.html. 10

[Pap14] Helen Papagiannis. Role of augmented reality in medical industry. January 2014. http://thenextweb.com/dd/2014/01/01/medical-augmented-reality-will-seamlessly-save-life/#!zshvK. 11

[PMK94] Akira Utsumi Paul Milgram, Harou Takemura and Fumio Kishino. Augmented reality: A class of displays on the reality-virtuality continuum. In *Telemanipulator and Telepresence Technologies*, volume 2351, pages 282–292. SPIE, 1994. http://wiki.commres.org/pds/Project_7eNrf2010/_5.pdf. 4

[Pro06] Proceedings of the IEEE and ACM International Symposium on Mixed and Augmented Reality. *Enhanced Visual Realism by Incorporating Camera Image Effects*, 2006. 48

[Qua14] Qualcomm. Qualcomm vuforia. 2014. http://www.qualcomm.com/solutions/augmented-reality. 40

[Res13] Juniper Research. Mobile augmented reality:smartphones, tablets and smart glasses 2013-2018. pdf, November 2013. A market research report. 1

[Roc12] Dominik Rockenschaub. Entwicklung und anwendung einer systematischen vorgehensweise zur analyse marker basierter augmented reality frameworks für mobile endgeräte. Master's thesis, Johannes Kepler Universität Linz, Linz, Austria, 2012. 43

[Sil12] Sanni Siltanen. *Theory and applications of marker-based augmented reality.* Julkaisija - Utgivare, 2012. 16

[Sol14] ARPA Solutions. User guide android sdk. 2014. http://www.arpa-solutions.net/Developer_Guide/Android/Index.html. 73

[Tot13] TotalImmersion. D'for adobe flash. 2013. http://www.t-immersion.com/product/dfusion-adobe-flash. 33

[Tot14a] TotalImmersion. D'fusion mobile. 2014. http://www.t-immersion.com/products/dfusion-suite/dfusion-mobile#. 36

[Tot14b] TotalImmersion. D'fusion studio suite. 2014. http://www.t-immersion.com/products/dfusion-suite. 33

[Tot14c] TotalImmersion. Our story. 2014. http://www.t-immersion.com/about-us/company-fact-sheet. 33

[Vuf13a] Vuforia. The best experience on mobile. 2013. https://www.vuforia.com/why. 40

[Vuf13b] Vuforia. Developing with vuforia. 2013. https://developer.vuforia.com/resources/dev-guide/getting-started. 41

[Vuf13c] Vuforia. Resources. 2013. https://developer.vuforia.com/resources/sample-apps. 40

[Wig13] Michael Wiggins. Press release: Mobile augmented reality users to approach 200 million globally by 2018, finds juniper research. November 2013. http://www.juniperresearch.com/viewpressrelease.php?pr=410. 1, 2

[Wik14] Wikipedia. Augmented reality. March 2014. http://en.wikipedia.org/wiki/Augmented_reality. 4