FOOD SHOPPING ASSISTANT WITH AUGMENTED REALITY

TABLE OF CONTENTS

1. INTRODUCTION

1.1. CONTEXT

By some estimates, the number of internet connected devices reached 8.7 billion in 2012. This total would include traditional computer devices, mobile devices, as well as the new industrial and consumer devices that we think of as things (e.g. printers). There will be about fifteen billion devices connected by 2015, and around forty billion devices by 2020.[1] [2]

In this rapidly expanding market "going mobile" is the future and an ever-growing array of devices allows users to access the internet from virtually anywhere in the world in various ways. Be it a smartphone, a tablet or a combination of these (the so called "phablet"), most people now have a powerful computing device that they carry around every day to stay "connected".

The increased hardware availability made way to software specifically developed for the new platforms, users being able to choose from thousands of applications for any domain/field of interest. Beside entertainment purposes (games, media), people can enhance the functionality of their devices by installing applications that perform certain tasks they need. Some applications focus on daily human activities and needs such as sharing information (pictures, ideas), remembering things, having a quality sleep, doing sport the right way, eating healthier or shopping.

1.2. PURPOSE

The proposed application intends to help its users do their food shopping efficiently and correctly; to offer access to its functionality in a very intuitive and easy manner it employs AR technology.

The main goal is to provide effective, live assistance to a person shopping for food. Products are automatically recognized and the user receives, in real time, information or tips regarding them. Based on a previously provided profile specifying eating habits and any medical condition that imposes restrictions, the application notifies about products that may not be recommended due to their nature and/or suggest similar products. The video seen by the person as he or she moves around the shop with the phone pointed towards the shelves is augmented with tooltips indicating sections of the shop where other products that may be of interest can be acquired.

The idea behind the proposed application emerged as increasingly powerful mobile devices got more and more into our daily lives not only for entertainment purposes, but also with the promise to ease out things we do not like to do and enhance those we enjoy. The application follows this very popular and practical trend of extending the capabilities of our gadgets and transforming them into valuable helpers that we carry around daily.

When choosing to develop a web application several aspects were taken into account. The idea that the next app OS is the Web Browser is very exciting and was the main reason

behind the decision. The most important benefit that immediately follows is that, in this way, the application will be available on any OS[3]. There is no need to rewrite the code several times or even worry about multiple OS, the app will simply run everywhere in a web browser.

In addition, these days a web-based application can have similar performances as a native, OS hosted one, due to new technologies implemented by manufacturers that are specifically tailored for low-powered devices.

1.3. BACKGROUND

The application combines concepts from two different emerging domains: *Augmented Reality* (AR) and *Recommender Systems*. Both sported a great increase in usage in the last decade and still have more to offer as technology improves.

1.3.1. AUGMENTED REALITY

Augmented Reality (AR), a part of "Mixed Reality", is a system that makes virtual and real objects appear together in real time. Such a system generates a composite view where images of virtual, computer-generated objects have been superimposed onto images of a real scene in such a way that the virtual objects appear to be part of the original scene to the human eye. This is a growing area in the field of virtual reality research; a prominent meeting place and display for AR technology is the annual international conference ISMAR[4].

AR is a growing area in the field of virtual reality research. The world environment provides a wealth of information that is difficult to duplicate in a computer. Fully virtual worlds are either very simplistic, such as the environments created for immersive entertainment and games, or the systems that can create a more realistic environment have a very high cost (the order of millions of dollars), such as flight simulators.

Coming as an alternative to these, AR has the goal to add information and meaning to a real object or place. Unlike virtual reality, augmented reality does not create a simulation of reality. Instead, it involves a scene from the real world used as a base to which contextual data is added to deepen a person's understanding of the subject. Some AR projects are researching and attempting to implement various ways to blur the line between the reality the user is experiencing and the virtual, computer generated content added on top of it.

Interaction categories

AR research and development introduced a number of new and exciting interaction techniques. Many of these techniques can be translated to mobile AR scenarios. To more easily classify methods, we categorize mobile AR apps based on the interaction the user needs to perform:

- *Embodied Interaction* – Interaction on the device itself (e.g., using the touchscreen)
- *Tangible Interaction* – Manipulation of AR scene using real-world objects
- *Interaction by Navigation* – Navigate the scene using body movement

Application categories with market traction

AR apps have been successfully used in the following application domains:

Education

- Museum/exhibit augmentation letting visitors 'see the unseen'
- Children's books come to life with a new way of story-telling
- Text books with scientific visualization in 3D

Instructional

- Interactive product manuals
- Step-by-step instructions to assemble or troubleshoot a product
- Guided AR tutorials, where the instructions are augmented on the actual product

Entertainment / Gaming

- Shooter
- Strategy
- Virtual pets
- Puzzles
- Action
- Sports simulation

Media & Advertising

- "Unlock" experiences to reveal hidden surprises
- Product catalogues with engaging or fun content
- Product enhancement/customization
- Extended engagement, like 'behind-the-scenes' information for readers in magazines
- Treasure hunts, e.g., over multiple pages in magazines
- Coupon redemption

Augmented reality may also be used in other contexts and scenarios:

- medicine – to help surgeons perform optimal interventions in operating rooms;
- aviation – to show pilots relevant data about the area they are flying in;
- training – to provide necessary data about objects to students or technicians during practical work;

Apart from these applications, the technology may be used in Engineering Design, Robotics / Telerobotics, Manufacturing, Maintenance and Repair, Consumer Design, Customer support / Sales.

Opportunities and challenges of mobile AR apps

Using augmented reality technology on mobile handsets presents a completely new set of interaction challenges and opportunities. There are new ways to view objects, to interact with augmented or real-world objects. Group interactions with other people when using AR is

an entirely new area for applications, where all users reach into a common scene in a tangible way or with helper objects.

AR allows new ways of holding the device, to create extensions, fixations, and holders. Tablets enable different AR use cases to interact comfortably directly on the screen with the augmented scene.

About Wikitude

Wikitude GmbH are the creators of the world's first mobile augmented reality platform and the company behind the internationally renowned Wikitude World Browser for iOS, Android, BlackBerry and Windows Phone devices. The AR platform has been voted "Best Augmented Reality Browser" three years in a row: 2009, 2010 and 2011. Wikitude provides industry leading computer vision technology and is leading the international AR technology standardization, chairing the working group in the Open Geospatial Consortium (OCG).

A new era is about to begin as Wikitude is enabling AR on the Web. The Wikitude Lab has created an AR solution called 'AR window for the Web', which enables any mobile web page to include augmented reality (patent pending).

Wikitude's AR window enables users to view a live video stream from the smartphone's camera on a mobile web page, with additional augmented content superimposed on it. This means that the immersive experience of connecting the real world with computer-generated content is brought to a whole new level.

The opportunity of AR on the web is enormous. While AR has been limited to app users only up until today, AR can now be integrated in millions of mobile web sites out there, allowing users to interact directly with the real world while browsing the web. Wikitude is trying to push the boundaries of what's technologically possible.[5] [6]

1.3.2. RECOMMENDER SYSTEMS

Recommender (or recommendation) systems are based on information filtering (IF) and attempt to present information items (movies, music, books, news, images, web pages, etc.) that are likely of interest to the user. The recommendations are generally calculated using the characteristics of an item or the user's social environment (*content-based* versus *collaborative filtering*). Looking at these approaches in practice, we can distinguish a more detailed classification of recommendations:

- Item-related – recommend things based on the item itself
- User-related
    - Personalized – recommend things based on the individual's past behaviour
    - Social – recommend things based on the past behaviour of similar users

These methods can also be used together as it is the case for most e-commerce Web sites, where such systems got very popular from the beginning due to the obvious business advantages they brought.

The difference between the methods previously mentioned can be illustrated with a few examples:

- *Pandora Radio* uses structural analysis of an item for its recommendations
- *Strands* focuses on social behaviour analysis around an item [7]
- *Aggregate Knowledge* does a structural analysis of an item, paired with a behavioural analysis around that item

An example of a recommender system using the *content-based* technique is *WhatShouldIReadNext.com*, a site where users can enter a title of a recent book they have read and enjoyed to see other books that they are likely to enjoy as well.

An example of a recommender system using *collaborative filtering* is Netflix, which suggests movies that a user might like to watch based on the user's profile. When building the profile of a user, his previous ratings and watching habits are compared to those of the other users.

Also, two of the most known internet companies combine the various approaches to recommendations:

- Amazon.com – probably the canonical example of recommendations technology on the Web – uses all three approaches (personalized, social and item). Amazon's system is very sophisticated, but at the heart all of its recommendations are based on individual behaviour, plus either the item itself or behaviour of other people on Amazon;
- Google – focuses on „personalized" recommendations: it customizes the user's search results "when possible" based on the user's location and/or recent search activity. When logged into his Google Account, the user "may see even more relevant, useful results" based on his web history. Beside these, the company's ranking algorithm PageRank is dependent on social recommendations (i.e. who links to a webpage) and its "Did you mean" feature can be counted as item recommendation. Furthermore, the recent launch of Google+ social network brought a new way for the search engine to learn user's preferences with the „+1" button that allows people to „Recommend on Search, Share on Google+";

1.4. PERSONAL CONTRIBUTION

The previous research projects that I participated into provided the necessary background knowledge for the technologies behind the application.

As such, in the field of Computer Vision some experience was acquired by working under the guidance of professor, Dr. Sabin C. Buraga at the project called *"PETI: Patient Eye Tracking Interface"*, together with fellow colleagues Mihăila Alina-Elena, George Alexandru

Vlad and Iulian Vascu. The project uses HAAR classifiers to detect blobs (with descriptors for the eyeballs) and the implementation that we used as a base was coded with the popular and powerful OpenCV library. Falling under the ACM category "H.5.2 Information interfaces and presentation - User Interfaces – Input devices and strategies", the project was included at the *RoCHI 2011 conference (http://rochi2011.utcluj.ro)* and more information about it is available on the dedicated website at http://peti-hci.blogspot.com/

For the recommender system, it was very helpful to have been worked at the project called „*PERE: A Location-based personal Semantic Web Recommender",* under the guidance of professor, dr. Sabin C. Buraga, professor Lenuţa Alboaie and together with fellow colleagues Mihăila Alina-Elena, George Alexandru Vlad and Iulian Vascu. For this project we used Pearson's correlation on arrays built out of scores calculated from the user's Facebook profile likes. The project was included at the *AQTR conference: http://www.aqtr.ro/*[8].

Getting the application from concept to implementation involved making decisions about the tools to be used for development. When the project was started there were already enough powerful, solid libraries implementing functionality necessary for AR; creating yet another implementation from scratch was neither optimal nor required. Instead, a detailed review was made in order to find the most appropriate existing toolkit, considering the particular needs of the envisioned application. More than twenty candidates were analysed and tested and the list was reduced to a couple of alternatives:

Vendor/name	**SDK**	**AR Browser**
Vuforia (Qualcomm)	yes	no
IN2AR	yes	no
D'Fusion	yes	no
AR23D	yes	no
ARmedia	yes	no
ALVAR	yes	no
Wikitude	yes	*yes* (multiple OS)
Metaio	yes	*yes* (multiple OS)
Layar	yes	*yes* (multiple OS)
PointCloud	yes	*yes* (iOS only)

Building a web application that would be accessible from any OS ruled out some excellent SDK/libraries like Vuforia due to being platform specific; such SDKs have a separate distribution for each major OS (Android, iOS, Windows) and this implies coding a different native application with each version in order to cover multiple OS. This also ruled out Flash-based SDKs because support is not great among Android family of devices and inexistent on iOS.

All the highlighted alternatives include a custom AR application for one or more OS; these "AR browsers" act like a client for AR content built specifically for each of them, allowing users to browse and use AR "channels" or "experiences". Only Wikitude and Metaio support actual applications with complex scripting and Wikitude was chosen as it goes one step further, allowing development of actual web applications (without AREL). In fact, the Wikitude application can be used as a regular web browser that in the same time has the capability to run any ARchitect-enabled web application.

1.5. SIMILAR APPLICATIONS

The application can be roughly seen as an AR-enabled companion, similar in some ways with existing solutions deployed on the web and on mobile platforms.

On one hand, there are quite a few examples of AR technology being used in a manner close to what the current application intends to do. Two examples are:

- *TwittARound* – does something similar to AR "browsers" showing live tweets around the user's location in such a way that the user is able to see where a tweet comes from and how far away is that place.
- *Augmented ID* – allows its users to point their devices at someone's face and automatically discover information about that person (Facebook, Twitter, business card, a photograph, etc.) as long as he or she is an *Augmented ID* user too.

On the other hand, the idea of offering the user potentially useful suggestions during an activity in which he or she is engaged in is not new, as previously discussed. The way the current application uses the recommendation technology can be observed on various companies that have an online presence:

- when shopping for things on Amazon.com, eMag.ro or many other stores the potential buyer gets additional products brought to his attention based on his record of products previously bought or visited (35% of Amazon's sales are from recommendation);
- when listening to music on Pandora Radio the user receives recommendations that take into account the songs or artists shown interest for;

These features are employed on websites that can be accessed from any PC and on most mobile devices connected to the internet, but the interaction with them is not specifically tailored for a mobile experience, lacking such purpose. With the increasing ubiquity of internet-accessing smart phones, it is now possible to offer personalized, context-sensitive recommendations on these devices as well.

This provides an opportunity for the current application to fill in the "gap" by combining the two technologies and making them available outside of the computer. The application intends to be a useful basic recommender system, employing unconventional information display using AR, which runs on mobile platforms.

There is an increasing wave of newly developed applications targeting specifically mobile platforms. One of them that inspired the idea behind the current application is *The Eatery* created by *Massive Health*[9] for iPhone users. As its creators say, it focuses on keeping the eating „record" of a person while allowing the user to rate each meal – identified by a photo – and encouraging friends to jump in with their own ratings and comments regarding how good is that food. When the history of a user shows signs of trouble the application kicks in with suggestions that may be used to improve the eating habits up to a healthy point.

The current application does not have the same coaching and lifestyle-changing goals, thus it is not an alternative or a better version of the previously mentioned one. The collaborative „food journal" generally received good feedback from the community which indicates that food could be a good choice as subject for the application. However, this proposal is focusing on the process of buying the right products efficiently, not on what someone is actually eating.

Instead of having friends/others comment and rate the user's choices and letting him read each comment and find/decide, based on those and the average ratings, if his choices were right or not, the proposed application does the judging part automatically. No comments are required from users and even ratings are optional – the simple act of a user "buying" certain items increases the chances for those items to be recommended to other buyers that share a similar purchase history with that user. This basic collaborative filtering technique plays a smaller role in the system as the suggestions are mostly "item-based", restricted by the profile of the user.

The system is not meant to be an advisor of the healthiest possible products, but instead a tool to ease shopping and discover similar products that the user would probably like/enjoy while taking into account his medical condition.

2. APPLICATION DESCRIPTION/ USER INTERFACE

This chapter presents the user interface and the interaction with the application.

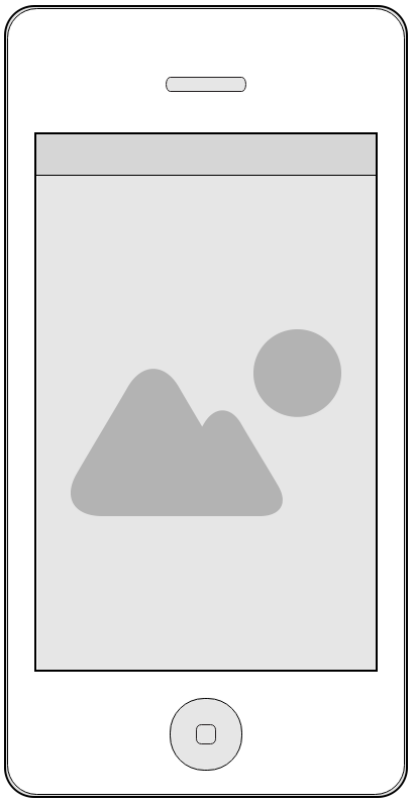
The interface is designed to allow a person to use the device as a Heads Up Display (HUD) while running the application. For this, the video coming from the camera is displayed on the whole screen (as background) and any UI items are displayed on top of it (in the foreground).

To provide a great experience the UI is as non-invasive as possible – only the necessary menu items and information are displayed over the video at any given moment of time. In contrast to this, the video stream is displayed at all times in the background during the usage of the application.

2.1. WELCOME SCREEN

2.1.1. PRIOR TO USER LOGIN

At start-up the user is presented with a pop-up asking for him to log in to the application; for convenience and a quick access two very popular social websites can be used to complete this step (Facebook and Twitter).



If the user is not logged in to any of these two, the authentication to the app is a 2 steps process: first the user logs into the service of his choice then he accepts the request of the app to access his personal data from that service. If the user is already logged in to either Facebook or Twitter on the device then accessing the application becomes a one-step action as he only has to grant the necessary permissions to the app.

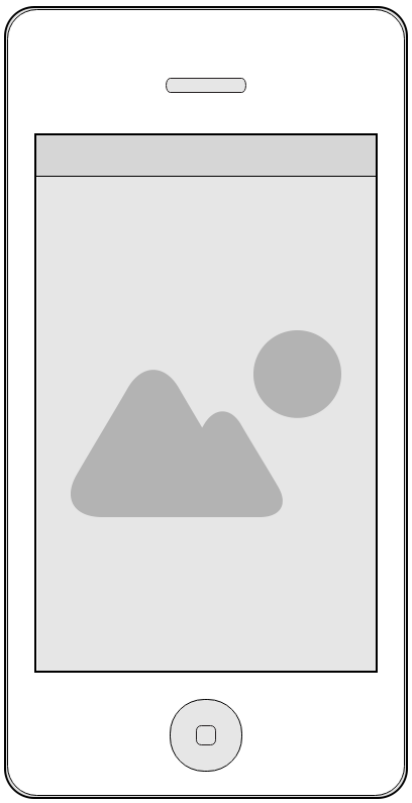
2.1.2. LOGGED IN USER

After the user logs in he is greeted with his name and the top bar of the application shows up, providing quick access to the options menu and the “Log out” function with 2 buttons placed in the left and right sides of the bar.

When the user taps on the “Log out” button a confirmation pop-up is displayed; if the user confirms the intention to leave the application then he is disconnected and the application returns to the 2.1.1 screen.

Tapping on the “Menu” button makes the options menu slide in from the left side but not completely as to fully cover the screen (allowing the user to peek at the video from the background). The menu offers the user access to the following functionalities:

- See a list of all the products identified since log in and either rate, mark as bought or disregard any of them (“Review products”)
- Review and update his personal details (“Personal information”)
- Review the total cost and the list of the products marked as bought (“Shopping basket”)
- Enable or disable certain functions of the application, like the informative balloons for identified products or the history (“Preferences”)



2.2. PRODUCT “RADAR” SCREEN

After the user logs in and whenever the side menu or the “Log out” pop-ups are closed the application returns to this default screen. The top bar is the only element constantly visible in this state.

2.2.1. STAND-BY/ SEARCH MODE

When the user doesn’t specifically perform an action (by opening a menu, for example) the application simply shows the video stream. The top bar’s title indicates to the user that the application is waiting for him to point the device around towards products.

2.2.2. PRODUCT RECOGNIZED

As soon as a product is identified the application augments the video with information about it. The informative bubble that shows up next to the product can be tapped by the user for more details; if no action is taken then such bubbles automatically disappear after 30 seconds.

2.3. PRODUCT DETAILS SCREEN

If the user taps on a product’s bubble any visible bubble is hidden and an overlay is shown that contains further details about the identified product. If available, similar products are also suggested at this point. On this screen the user also has the possibility to rate, mark as bought or disregard for future sessions the current product.

The application allows the user to alter its functioning. By choose to have particular products not recommended again the application will adapt its future suggestions. The user can also disregard particular products and these will not be recognized in the future.

3. TECHNICAL DETAILS

This chapter offers a detailed view over the two domains the app is taking advantage of, followed by the list of the tools and technologies used in the application; finally, the most important aspects of the implementation are laid out.

CHALLENGES

Both fields have their share of complexity which derives mainly from the common functional need of estimating "things", whether it is the similarity between two images, what is the positioning of an object in an image/frame or how related are two or more user profiles or users past actions.

AUGMENTED REALITY

Implementing a feasible AR system involves a mix of complex analysing and rendering algorithms and the appropriate hardware. The system must be able to execute the following operations:

1. *Detection*: real world points of interest/references are identified in the video stream
2. *Content generation*: simple overlays or even 3D objects are rendered on demand
3. *Merging*: items from step #2 are correctly positioned relative to the items from step#1 and are simultaneously displayed to the user

As for the hardware, a high-resolution video camera for input can enhance the functioning of the system. A sharper image may help reduce the inconsistencies that normally occur with poor lighting conditions or partially hidden objects, in which cases the detection is not precise.

The *merging* operation consists of placing the generated content "inside" the real world imagery, positioning the virtual content in the right coordinates with respect to the correct 3D perspective of the camera.

Both *detection* and *merging* are non-trivial and usually require intense computations that have to be optimized as much as possible in order to allow the system to react in real-time, any delays affecting the end-user experience.

Processing divided in two stages:

- unique feature detection
- verification/identification

The performance of an AR system can be rated by a couple of factors, such as detection rate, false positive rate, inter-marker confusion rate, minimal detection size and the immunity to lighting or occlusion.

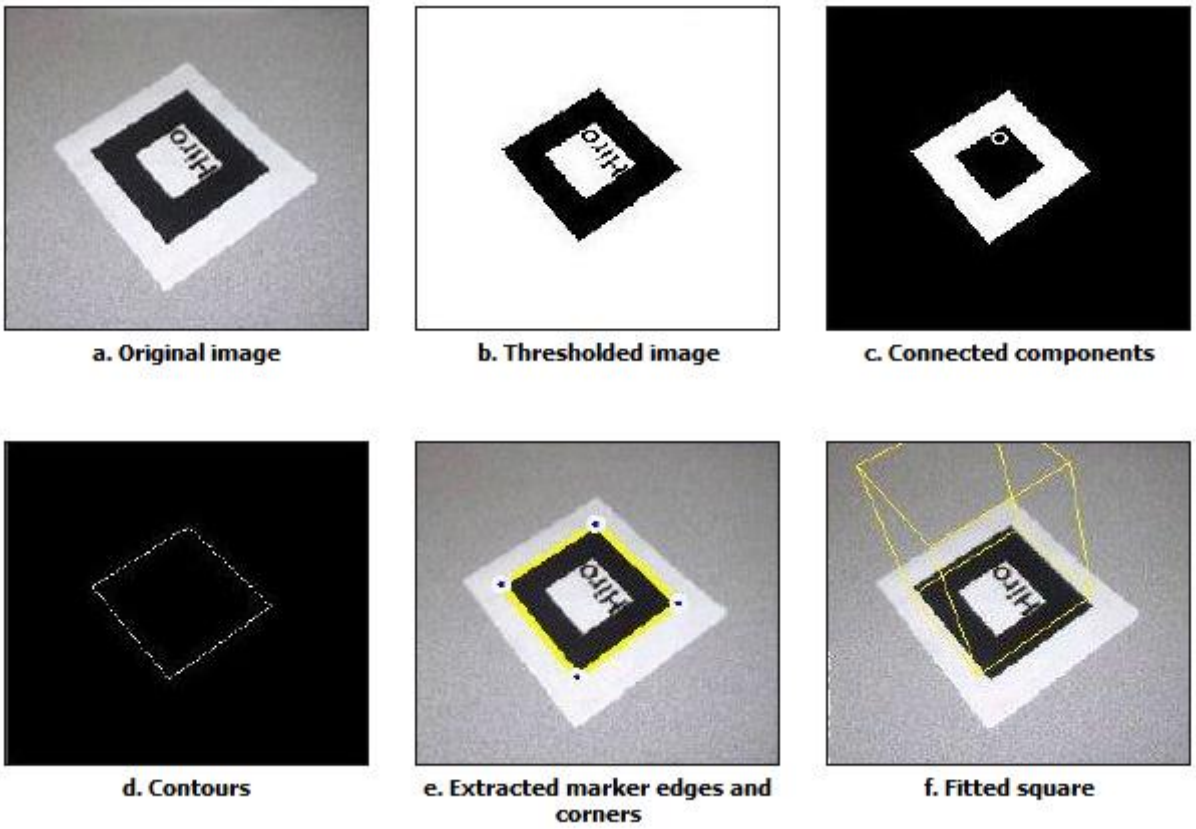
Detection and *merging* can be achieved in various ways depending on the requirements/context:

A. Marker-based

If possible, the system can take advantage of (fiducial) markers to detect objects. **Fiducial marker systems** consist of patterns that are mounted in the environment and automatically detected in frames (images) from a video or a camera stream. They help solve the correspondence problem, automatically finding features in different frames that belong to the same object point in the real world.

Marker-based systems are based on one of the following methods:

1. *A basic corner detection approach with a fast pose estimation algorithm* – as employed by ARToolKit library[10]. The following sequence illustrates the functioning of the system:



a. Original image | b. Thresholded image | c. Connected components

d. Contours | e. Extracted marker edges and corners | f. Fitted square

2. *An approach based on working with affine representations*[11] – as described by J.Vallino from Rochester Institute of Technology in his PhD thesis[12].

Not based on relating all reference frames to a common Euclidean 3D reference frame, it specifies the camera, real world and virtual objects in an affine representation (four non-coplanar points in the scene define this affine frame).

The technique utilizes the following property of affine point representations[13,14]: *Given a set of four or more non-coplanar 3D points represented in an affine reference frame, the projection of any point in the set can be computed as a linear combination of four points in the set*.

The affine representation allows the calculation of a point's projection without the requirement of having any information about the camera position or its calibration

parameters. However, it does represent only those properties of an object that are preserved under an affine transformation (lines, intersections of lines and planes and parallelism).

The problem of correctly merging a virtual image with an image of a real scene is reduced to:

1. tracking a set of points defining the affine basis that may be undergoing a rigid transformation;
2. computing the affine representation of any virtual objects;
3. calculating the projection of the virtual objects for a new scene view as linear combinations of the projections of the affine basis points in that new view;

3. *An edge-based detection approach* – this is a prototype proposed by *Corné den Otter* and *Reinder Nijhoff*[15] from Infi (a Dutch software company).

It is based on Martin Hirzer's article from 2008, 'Marker Detection for Augmented Reality Applications'[16], explaining an edge based marker-detection algorithm that is outlined below:

- Step 1: Divide image in regions
- Step 2: Detect edgels in regions
- Step 3: Find segments in regions[17] [18]
- Step 4: Merge segments to lines
- Step 5: Extend lines along edges
- Step 6: Keep lines with corners
- Step 7: Find markers

After finding all markers, for each marker the 4 intersections of the lines in the chain are considered to be the corners. Calculating the positions of corners by these line-intersections gives a robust algorithm. Even if only 3 lines are detected and/or a corner is occluded, the marker will be correctly detected most of the time.

Now we have the coordinates of the detected markers. The next step would be to identify markers and distinguish them from each other in order to use this algorithm in an augmented reality application.

B. Markerless

Markerless AR systems can be achieved based on:

- *Using a planar structure to enable "ad hoc" AR* – if the scenery contains a planar structure, a different method using homographies may be used to augment the scene without the need of any markers.

In his thesis[19], *Björn Liljequist* proposes an AR system that requires a planar structure to be present in the scene and takes advantage of this fact to augment the scene with computer-generated graphics without using any markers.

As the author shows in his paper, when the scene consists of a plane the mapping of points between images is described by **3x3** matrices $\mathbf{H}^{j}_{i}$ known as homographies. Furthermore the mapping of points on the scene plane to points in an image is also described by a homography $\mathbf{H}^{i}_{s}$. As indicated by the author, we can automatically estimate the homography $\mathbf{H}^{i}_{s}$ between any two images i and j based on corresponding corners in the two images. The focal length of the camera along with the scene to image homography $\mathbf{H}^{i}_{s}$ for the first image can be calculated through user input. Since $\mathbf{H}^{i}_{s} = \mathbf{H}^{i}_{1}$ $\mathbf{H}^{1}_{s}$ this means that we can estimate the scene to image homography $\mathbf{H}^{i}_{s}$ for each and every one of the images in the sequence. Furthermore the camera matrix Pi for any image i in the sequence can be reconstructed if we know $\mathbf{H}^{i}_{s}$. This lead the author to formulate the following basic algorithm for camera tracking:

1. Let the user specify an area in the first image indicating the plane region. The plane region is the part of the image containing the studied plane.
2. Determine focal length and the scene to image homography $\mathbf{H}^{i}_{s}$ for the first image through user input. This is done by having the user indicate four points in the first image corresponding to the corners of a rectangle on the scene plane.
3. Find corners in each image of the sequence.
4. For each image i > 1 in the sequence:

(a) Match corners between image i-1 and i. Consider corners within the plane region only.

(b) Estimate the homography $\mathbf{H}^{i}_{i-1}$ describing the mapping of points between adjacent images based on the corner matches.

(c) Calculate $\mathbf{H}^{i}_{s} = \mathbf{H}^{i}_{i-1}\ \mathbf{H}^{i-1}_{s}$.

(d) Reconstruct the camera matrix P^{i} based on the estimated homography $\mathbf{H}^{i}_{s}$.

(e) Update the plane region using the estimated homography $\mathbf{H}^{i}_{i-1}$.

This algorithm can be improved in several ways; for example, by estimating homographies between images that are not adjacent in the sequence. Also, it is possible to handle the case where a new plane becomes dominant in the scene. In addition, there are techniques for handling varying focal length and for improving the results after initial estimations of the camera matrices have been calculated for the entire image sequence.

- *Using feature extraction and tracking for one or more "target" images* – this is the approach used by the Vuforia SDK and implicitly by Wikitude's AR Browser.

In pattern recognition and in image processing, feature extraction is a special form of dimensionality reduction. Feature extraction involves simplifying the amount of

resources required to describe a large set of data accurately. It can be used in the area of image processing which involves using algorithms to detect and isolate various desired portions or shapes (features) of a digitized image or video stream.[20]

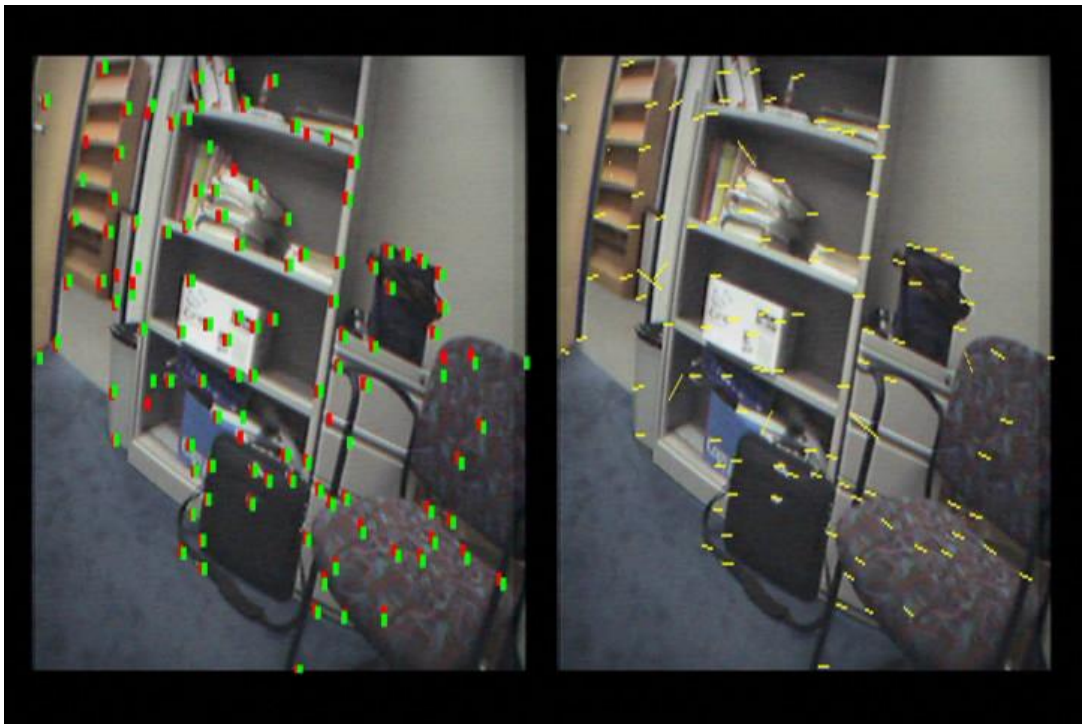
Types of feature extraction:

- **Low-level**
  - Edge detection
  - Corner detection
  - Blob detection
  - Ridge detection
- **Shape based**
  - Thresholding
  - Blob extraction
  - Template matching
  - Scale-invariant feature transform

In most feature tracking systems there are 2 stages: first, generating local features by extracting interest points (be it edges, corners, or other blobs) then tracking them in successive sequences. Admittedly, feature extraction and tracking have a close relationship, which means that the way the two are combined plays a key role in the system's functioning.

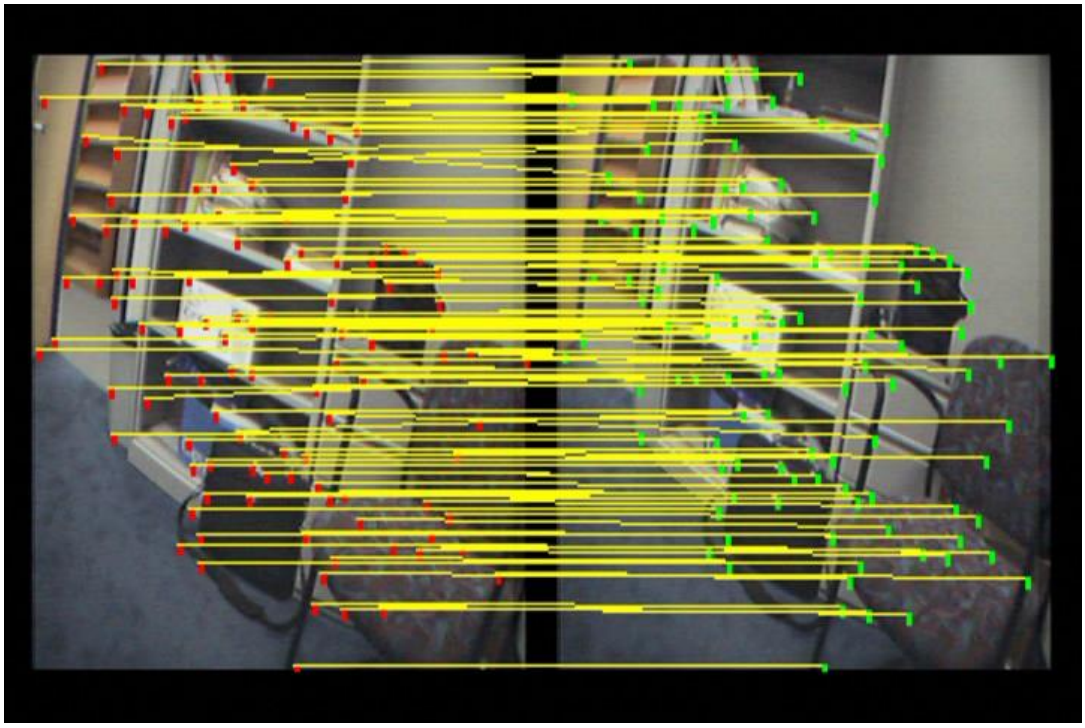
There are multiple methods to obtain features out of an image: histograms of gradients, wavelets, LESH, GLOH, ORB, SURF, SIFT, MSER, FERNS.

Below is an example showing the output of feature detection (left) and tracking orientation (right) operations for 2 successive frames: [21]



This feature detector is using Harris Corner detection for extracting interest points and SIFT for generating feature descriptor.

The following sample shows the output of the feature matcher using distance between orientation distribution descriptor:



RECOMMENDER SYSTEMS

In order to create a system that offers relevant suggestions to its users, a couple of key aspects must be handled correctly. First, the context in which the system will function has to be analysed to decide what kinds of recommendations are needed. Obviously, the nature of the business plays a role in this, as not all scenarios allow just about any method discussed earlier to be deployed.

If opting for the *content-based* approach, then care must be taken when creating and enriching the ontologies that the algorithm relies on. Incorrect relationships or misplaced features can negatively affect the output of the system. While an automated process of acquiring features can be applied to texts or products with relative ease, this method is not suitable for other frequent types of items like music or movies. For these it comes down to relying on tags, which can contain noise that the system should be able to remove or at least reduce.

The *content-based* technique generally involves the use of ontologies to deepen the "understanding" of the automated tool about how each item is related to others. Such ontology is composed of a basic ontology containing the items themselves and domain specific ontologies holding data about the nature of those items. The specific and dynamic features that can be extracted by analysing this ontology are the key to the correct selection of items that can be suggested for a particular item.

In the case of *collaborative filtering*, a recommender system uses details of the registered user's profile together with opinions and habits of the whole community of users and compares the information to reference characteristics. This recommendation method assumes that it is likely for users with similar past activity to have common interests too.

Based on the behaviour (the ratings offered to items) and profile details of each user and taking into account the activity (rating/browsing) patterns of all users the system builds the „user cluster". The cluster is created in a „bottom-up" manner and is an advanced and effective way of grouping users with „similar" tastes together. This classification is the most important part of such a system and is constantly updated after each new action of a user.

Doing so makes the system adaptable in time and assures that users are always placed in the best-suited cluster partition. By analysing the clusters, the system can decide at any moment who are the most representative users that should be taken into account when offering suggestions to one particular user.

Choosing to use the *collaborative filtering* shifts the development efforts towards the clustering algorithm and the subsequent classifier. This approach is generally more complex because of the way the system works:

1. *Input*: collect data on the user's preferences
2. *Clustering*: use the data to assign the user to appropriate user clusters
3. *Output*: use the clusters to generate recommendations for the user

A key difficulty with implementing robust collaborative filtering technology in real time has been scalability: accurately finding the most similar users to a given user as the user base grows has been viewed by some as computationally intractable. This problem can be avoided by moving the most computationally intense calculations offline, using a batch process to divide the user base up into groups, or clusters, of users of similar tastes, and to compute a range of statistics associated with these user clusters. To generate recommendations for a new user, the system must simply search through the clusters (rather than through the users) to find ones that reflect the user's tastes, and then recommend based on the statistics associated with such clusters. Other aspects that could result in system malfunctioning and should be handled are sparse data, cold-start and popularity bias.

It can be noted that no matter what method is used to build a recommender system, the "hidden" goal is to create and maintain some sort of knowledge base to use later on. What is actually using that gathered data is another important component of any recommender system: the classifier. Its role varies from measuring similarities between items, profiles or past behaviour to finding the appropriate cluster(s) for a specific user. These tasks are solved using methods like cosine or Pearson's correlation, matrix factorization, tf*idf weight, KL-divergence, and K-means.

Fig.1 shows the structure of a hybrid system that uses the *content-based* and *collaborative filtering* techniques together.

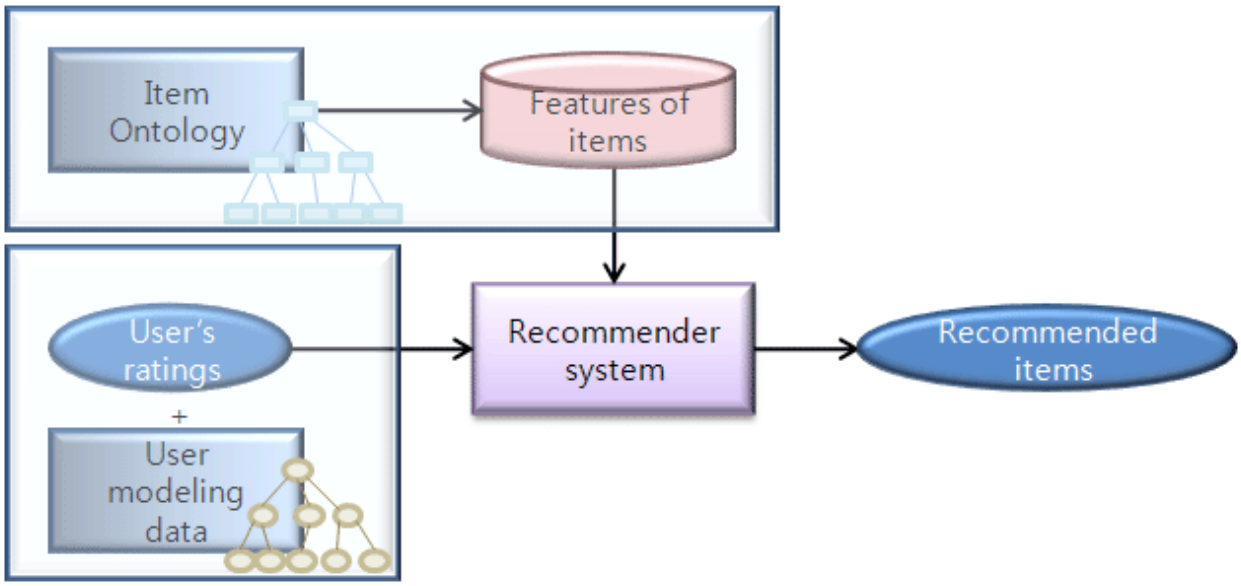


Figure 1 – Recommender system

TECHNOLOGIES

BACKGROUND

FastCV

FastCV is the first mobile-optimized computer vision (CV) library for ARM-based devices[22] and it is designed for developers interested in creating sophisticated CV apps, as well as CV middleware developers looking to build the frameworks necessary for everyday developers to include computer vision functionality in their apps.

FastCV is the framework at the heart of the Vuforia vision-based Augmented Reality SDK, because AR is much more precise and useful when it is based on camera input than on location-based estimates. The authors of FastCV anticipate it will be used by middleware developers to build additional frameworks that will allow developers of computer vision apps to build compelling experiences around CV that take advantage of:

- gesture recognition
- face detection, tracking and recognition
- text recognition and tracking
- depth of field calculations

The FastCV tools easily tackles some of the harder aspects of computer vision technology, allowing developers to focus on building compelling marketing platforms.

Vuforia

The Vuforia SDK won the Best Augmented Reality SDK Auggie Award at the Augmented World Expo 2013 in Santa Clara, California. Augmented World Expo is the world's largest event in the augmented reality (AR) industry and the 2013 event attracted 1,100+ AR professionals with 40 hours of sessions from industry leaders.

Contenders included AR SDK kits for any platform and the decision criteria were:

- Quality of the SDK: Includes breadth and depth of features, specifically applied to augmented reality SDKs
- User Experience: Includes design, ease of use, visual appeal that enable users to develop an innovative AR user experience
- Impact: Includes the potential to influence how people use augmented reality to advance their lives and businesses

In addition, Vuforia-based Wikitude AR browser won the title for “Best Augmented Realty Browser” from *Readers Choice Awards 2012* **of Augmented Planet**[23] and **Wikitude Architect** was nominated as the “Best Developer Tool or Platform” in the same community-driven voting.

Features

- Using both cameras
  Vuforia SDK 2.0 lets apps use the front camera on mobile devices, in addition to the rear camera. By supporting all cameras on the mobile device, developers have more flexibility to experiment with new use cases for AR apps.

- Cloud Recognition[24]
  Today's Vuforia apps work by recognizing an image from a database of about 100 targets on the device. This works really well for games and small branded campaigns. However, retailers or publishers of magazines, catalogues, newspapers or other print media have thousands or tens of thousands images. Just too many images are changing too often to have everything on the device. Developers can build AR apps that work against databases in the cloud with over a million images each. The Vuforia Web Services (VWS) provide RESTful APIs that allow applications to efficiently manage large volume databases. With these APIs, app developers can seamlessly integrate VWS into their own content management systems.

- User-defined targets
  Vuforia SDK 2.0 allows end users to play their AR games "anytime anywhere" by eliminating the reliance on pre-determined targets that a developer has to pick, instead enabling end users to select their own image targets from everyday objects like books, magazines and photos. It is as easy as taking a picture and creating a target, thus removing the need for an end user to carry around the target required for the AR game.

- Support for Play Mode for Unity
  Vuforia also supports Play Mode for Unity. The webcam APIs were updated to activate the PC's camera or USB webcam from inside the Unity Editor and let it see the real-world image you are building on. As a developer builds an app in Unity, he can move back and forth between the real and virtual worlds and debug directly in the Unity Editor.

The Vuforia ecosystem has 60,000+ developers in 130 countries with 4,300+ commercial apps in distribution[25].

WIKITUDE

AR Browser

This application, looking just like a regular web browser, handles all the AR-related tasks and runs the ARchitect-enabled web application inside it. It is a native application running on the device thus it has full access to all the computing power and the needed hardware of the device. By processing the video stream captured from the device's camera it performs the following operations on the frames:

1. feature detection;
2. feature matching between the output of step 1 and the custom dataset provided by the web application (that defines one or more image targets);
3. feature tracking and content rendering at the appropriate coordinates;

Similar applications:

- *SREngine* – does scene recognition (architecture, streets, posters, rooms, etc.);
- *Junaio* and *Layar* acts too like a "browser" of the real world for augmented content; the striking difference between these and Wikitude is how custom content is added and displayed.

ARchitect

The ARchitect library is the link between the native Browser application and the web application, providing full control over the AR engine through a JavaScript API[26]. This API is loaded inside the web application through a script:

```
<script src="architect://architect.js"></script>
```

For debugging purposes the ARchitect Desktop Engine library can be loaded as well:

```
<script type="text/javascript" src="ade.js" ></script>
```

Tracking initialization:

```
var tracker = new AR.Tracker("http://myserver.com/patternset1.zip", {
  onDisabled: function () {
    //tracker has been disabled by the system
  }
});
```

Generate a graphic with a size relative to both the target size and the distance from camera to the target:

```
// a circle used for representation
var circle = new AR.Circle(5);
```

Start to track a specific target and overlay the generated graphic:

```
// a Trackable2DObject using the "car" target in the tracker, using the circle as the digital representation.
var trackable2DObject = new AR.Trackable2DObject(tracker, "car", {
  drawables : { cam : circle }
});
```

LUNGO.JS FRAMEWORK

Lungo.js is a framework for developers who want to design, build and share cross device applications.

Features:

- *Optimized Apps*: Lungo is based on open web standards, such as HTML5, CSS3 and JavaScript.
- *Powerful JavaScript API*: A well-documented, robust JavaScript API offers complete control of the app; Lungo.js is modular and completely customizable
- *Multi-Device full support*: creating apps for each platform is difficult and the arrival of new devices does not make it easier. Lungo will suit all of them creating a unique and amazing UX. The applications created with it work in all of the popular platforms (iOS / Android / Blackberry / FirefoxOS). Lungo provides a consistent browser environment across mobile, TV's and desktop devices.
- It has support for numerous touch events like tap, double-tap or swipe and does not use images (the icons are obtained with a special font).

COUCHDB[27]

CouchDB is a database that "completely embraces the web". It stores data with JSON documents and allows access to them by using any web browser, via HTTP. By using JavaScript alone developers can query, combine, and transform documents. CouchDB works well with modern web and mobile apps. You can even serve web apps directly out of CouchDB. Developers can distribute their data or their apps efficiently using CouchDB's incremental replication. CouchDB supports master-master setups with automatic conflict detection.

CouchDB comes with a suite of features, such as on-the-fly document transformation and real-time change notifications, that makes web app development a breeze. It even comes with an easy to use web administration console that is served up directly out of CouchDB. The system was designed with distributed scaling in mind, as is highly available and partition tolerant, but also eventually consistent. CouchDB has a fault-tolerant storage engine that puts the safety of the data first.

CouchDB is often categorized as a "NoSQL" database, a term that became increasingly popular in late 2009, and early 2010. While this term is a rather generic characterization of a database, or data store, it does clearly define a break from traditional SQL-based databases. A CouchDB database lacks a schema, or rigid pre-defined data structures such as tables. Data stored in CouchDB is a JSON document(s). The structure of the data, or document(s), can change dynamically to accommodate evolving needs.

Features:

- **Views**
- **Schema-Free**
- **Distributed**:
    - Master → Slave replication
    - Master ↔ Master replication
    - Filtered Replication
    - Incremental and bi-directional replication
    - Conflict management

HTML5[28] [29] [30]

HTML5 is a mark-up language for structuring and presenting content for the World Wide Web and a core technology of the Internet. It is the fifth revision of the HTML standard (created in 1990 and standardized as HTML 4 as of 1997) and, as of December 2012, is a W3C Candidate Recommendation. Its core aims have been to improve the language with support for the latest multimedia while keeping it easily readable by humans and consistently understood by computers and devices (web browsers, parsers, etc.). HTML5 is intended to subsume not only HTML 4, but also XHTML 1 and DOM Level 2 HTML.

It is also an attempt to define a single mark-up language that can be written in either HTML or XHTML syntax. It includes detailed processing models to encourage implementations that are more interoperable; it extends, improves and rationalizes the mark-up available for documents, and introduces mark-up and application programming interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a potential candidate for cross-platform mobile applications. Many features of HTML5 have been built with the consideration of being able to run on low-powered devices such as smartphones and tablets. Research firm Strategy Analytics anticipated that sales of HTML5 compatible phones would top 1 billion in 2013.

The new mark-up language was developed based on pre-set standards:[31]

- New features should be based on HTML, CSS, DOM, and JavaScript.
- The need for external plugins (like Flash) needs to be reduced.
- Error handling should be easier than in previous versions.
- Scripting has to be replaced by more mark-up.
- HTML5 should be device-independent.
- The development process should be visible to the public.

CSS3[32] [33] [34]

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation semantics (the look and formatting) of a document written in a mark-up language. Its most common application is to style web pages written in HTML and XHTML, but the language can also be applied to any kind of XML document, including plain XML, SVG and XUL.

CSS is designed primarily to enable the separation of document content (written in HTML or a similar mark-up language) from document presentation, including elements such as the layout, colours, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content (such as by allowing for tableless web design).

CSS can also allow the same mark-up page to be presented in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader) and on Braille-based, tactile devices. It can also be used to allow the web page to display differently depending on the screen size or device on which it is being viewed. While the author of a document typically links that document to a CSS file, readers can use a different style sheet, perhaps one on their own computer, to override the one the author has specified.

CSS specifies a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called cascade, priorities or weights are calculated and assigned to rules, so that the results are predictable.

The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998), and they also operate a free CSS validation service.

Unlike CSS 2, which is a large single specification defining various features, CSS 3 is divided into several separate documents called "modules". Each module adds new capabilities or extends features defined in CSS 2, over preserving backward compatibility. Work on CSS level 3 started around the time of publication of the original CSS 2 recommendation. The earliest CSS 3 drafts were published in June 1999. Due to the modularization, different modules have different stability and statuses. As of June 2012, there are over fifty CSS modules published from the CSS Working Group., and four of these have been published as formal recommendations:

- 2012-06-19 : Media Queries
- 2011-09-29 : Namespaces
- 2011-09-29 : Selectors Level 3
- 2011-06-07 : Colour

Some modules (including Backgrounds and Borders and Multi-column Layout among others) have Candidate Recommendation (CR) status and are considered moderately stable. At CR stage, implementations are advised to drop vendor prefixes.

The development of CSS3 is going to be split up into 'modules'. The old specification was simply too large and complex to be updated as one, so it has been broken down into smaller pieces – with new ones also added. Some of these modules include:

- The Box Model
- Lists Module
- Hyperlink Presentation
- Speech Module
- Backgrounds and Borders
- Text Effects
- Multi-Column Layout

JAVASCRIPT[35]

JavaScript (JS) is an interpreted computer programming language. It was originally implemented as part of web browsers so that client-side scripts could interact with the user, control the browser, communicate asynchronously, and alter the document content that was displayed. More recently, however, it has become common in both game development and the creation of desktop applications.

JavaScript is a prototype-based scripting language that is dynamic, is weakly typed, and has first-class functions. Its syntax was influenced by the language C. JavaScript copies many names and naming conventions from Java, but the two languages are otherwise unrelated and have very different semantics. The key design principles within JavaScript are taken from the Self and Scheme programming languages. It is a multi-paradigm language, supporting object-oriented, imperative, and functional programming styles.

JavaScript's use in applications outside of web pages—for example, in PDF documents, site-specific browsers, and desktop widgets—is also significant. Newer and faster JavaScript VMs and frameworks built upon them (notably Node.js) have also increased the popularity of JavaScript for server-side web applications.

JavaScript was formalized in the ECMAScript language standard and is primarily used as part of a web browser (client-side JavaScript). This enables programmatic access to computational objects within a host environment.

[36]Contrary to popular misconception, JavaScript is not "Interpretive Java". In a nutshell, JavaScript is a dynamic scripting language supporting prototype based object construction. The basic syntax is intentionally similar to both Java and C++ to reduce the number of new concepts required to learn the language. Language constructs, such as if statements, for and while loops, and switch and try ... catch blocks function the same as in these languages (or nearly so.)

JavaScript can function as both a procedural and an object oriented language. Objects are created programmatically in JavaScript, by attaching methods and properties to otherwise empty objects at run time, as opposed to the syntactic class definitions common in compiled languages like C++ and Java. Once an object has been constructed it can be used as a blueprint (or prototype) for creating similar objects.

JavaScript's dynamic capabilities include runtime object construction, variable parameter lists, function variables, dynamic script creation (via eval), object introspection (via for ... in), and source code recovery (JavaScript programs can decompile function bodies back into their source text)

Intrinsic objects are Number, String, Boolean, Date, RegExp, and Math.

JSON[37]

JavaScript Object Notation (JSON) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

JSON is built on two structures:

- A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
- An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

These are universal data structures. Virtually all modern programming languages support them in one form or another. It makes sense that a data format that is interchangeable with programming languages also be based on these structures.

In JSON, they take on these forms:

- An *object* is an unordered set of name/value pairs. An object begins with { (left brace) and ends with } (right brace). Each name is followed by : (colon) and the name/value pairs are separated by , (comma).
- An *array* is an ordered collection of values. An array begins with [(left bracket) and ends with] (right bracket). Values are separated by , (comma).
- A *value* can be a *string* in double quotes, or a *number*, or true or false or null, or an *object* or an *array*. These structures can be nested.
- A *string* is a sequence of zero or more Unicode characters, wrapped in double quotes, using backslash escapes. A character is represented as a single character string. A string is very much like a C or Java string.

- A *number* is very much like a C or Java number, except that the octal and hexadecimal formats are not used.
- Whitespace can be inserted between any pair of tokens.

Except for a few encoding details, this completely describes the language.

APIs

FACEBOOK API

Facebook Login

This Facebook feature provides a secure and easy way for people to log in to an application or website. Facebook Login makes it easy to connect with people using an app or a website. Developers can use the JavaScript or mobile SDKs to speed up the registration process and build a functional system in minutes.

The Facebook SDK for JavaScript provides a simple path to integrating Facebook login with websites and mobile web apps. Different platforms and different types of apps have different ways of completing the login process.

TWITTER API

Sign in with Twitter

This functionality offered by Twitter enables the placement of a button on a site or application which allows Twitter users to enjoy the benefits of a registered user account in as little as one click. Works on websites, iOS, mobile and desktop applications as well.

Features

- **Ease of use** - A new visitor on a site only has to click two buttons in order to sign in for the first time.
- **Twitter integration** - The Sign in with Twitter flow can grant authorization to use Twitter APIs on users behalf.
- **OAuth based** - A wealth of client libraries and example code are compatible with Sign in with Twitter's API.

Available for

- **Browsers:** Sign in with Twitter can be integrated on any website.
- **Mobile devices:** Any web-connected mobile device can take advantage of Sign in with Twitter.

ARCHITECTURE

The application stack is described by the following diagram:

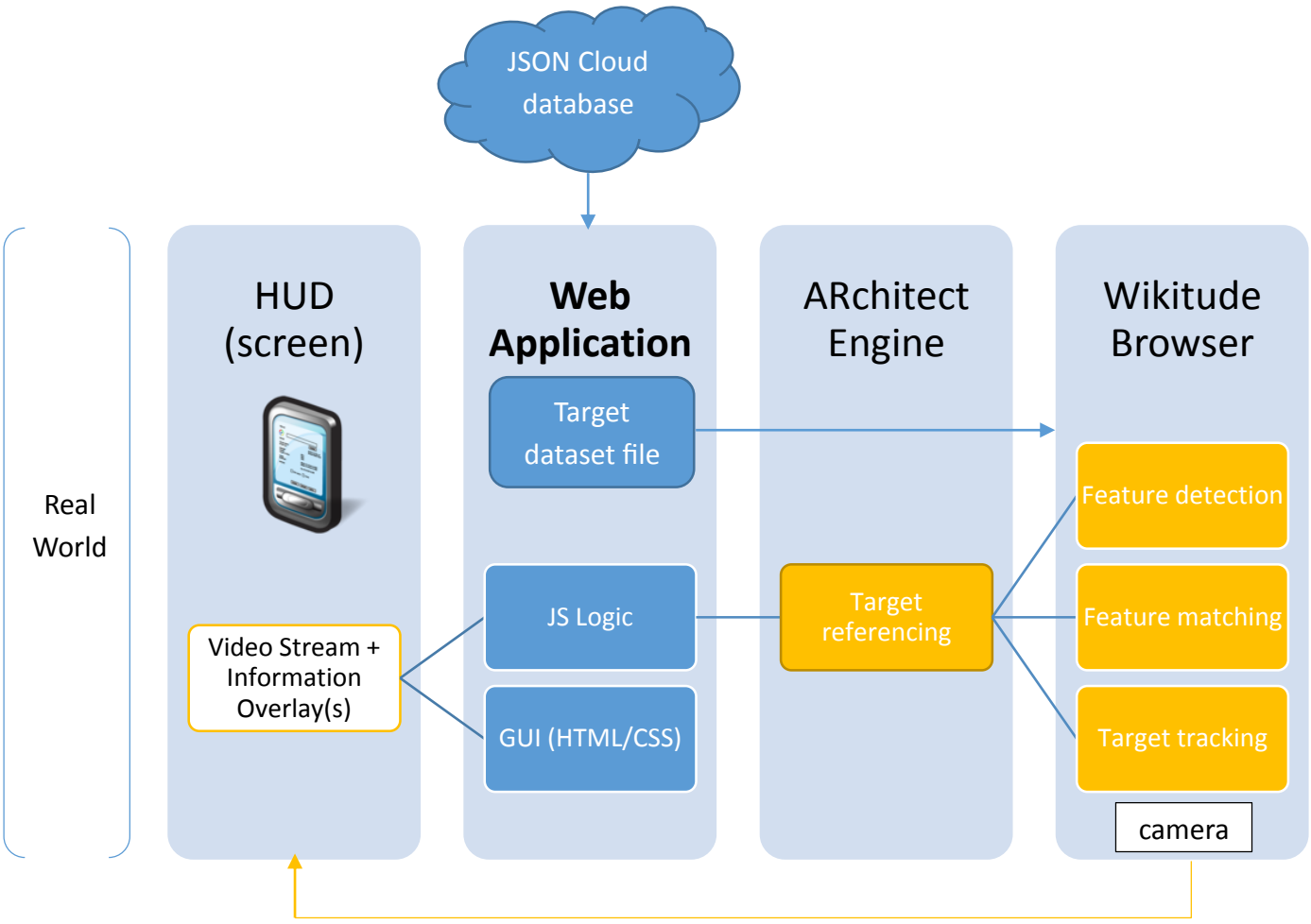


Figure 2 – System architecture

As visually indicated above, the application runs inside the Wikitude Browser and connects to it through the ARchitect library. The blue boxes identify the components of the application, while the yellow ones are parts of the Wikitude platform. Low-level AR related functionality is handled by Wikitude and exposed using ARchitect.

As it can be seen in Fig.2, the application logic is clearly separated from the user interface. The framework used as a base allows to write meaningful, semantic HTML code that benefits from HTML5's new features (like data attributes and fresh tags).

The structure of the application is the following:

- **actions**: contains the JS logic
- **app**: contains the HTML mark-up

    - **asides**: holds the sliding menus
    - **sections**: holds the sections and the pop-ups

- **components**: contains 3rd party libraries
    - **fancybox**: holds a plugin for displaying pop-ups
    - **lungo**: holds the framework files
    - **quojs**: holds the base library used by the framework
- **images**: contains images used in the UI
- **media**: contains images of products
- **stylesheets**: contains custom CSS that change the framework's default UI styling
- **index.html**: the entry point of the application

The application logic is divided into 3 modules that are implemented as separate classes and are located in the **/actions** directory:

- ARHandler
- RecommendationGenerator
- AppEngine

ARHandler

This module is responsible with obtaining "references" of the objects from the real world that are in the user's view. The application "learns" about an object automatically; while the user moves around the shelves with products, the image detection engine runs continuously in background scanning the frames captured by the camera. These are compared against the products images described in the provided dataset. When one or more matches are found (with a certain likelihood), they are traced to the products they belong.

When queried, the output of this module consists of one or more products represented by IDs that are communicated to the **AppEngine** module.

This module also has to convert all AR related commands received from the **AppEngine** module into *ARchitect API calls* that actually trigger the output of generated content in the camera view. The pop-ups with details are shown by the AR rendering engine within the real scenery.

RecommendationGenerator

Having received the products IDs, this module queries the cloud database for information. For each product the module receives data previously collected by the system (like rating and purchase history). An algorithm that outputs other products that may be of interest for the current user analyses:

- the previous actions of users (who bought what)
- the interests of users that bought the same product as the one being queried

To solve the cold-start issue where there is no or little data available from other users the module runs a different algorithm that returns products similar by content (ingredients).

Just as for the **ARHandler** module, the output of this module consists of one or more products represented by IDs that are communicated to the **AppEngine** module.

AppEngine

This module is the one controlling the flow of the application. It uses the output of the **ARHandler** in 2 ways:

1. as input for the **RecommendationGenerator** together with the user ID in order to obtain similar or potentially interesting products
2. to query the cloud database for product details

This module collects pieces of information about the identified and recommended products that will be presented to the user via the HUD. The module establishes what is augmented in the camera view and passes the necessary commands to the **ARHandler** module.

At the same time, it takes care of generating the product details screen based on both the output from **RecommendationGenerator** module and the database. For each product, basic details are available (like name, producer, origin, ingredients).

4. CONCLUSIONS

CASE STUDY

A person goes to shop for food in a hypermarket. While looking around not being able to decide what to pick, the uncertain buyer grabs his mobile device and starts up the application. As he cannot decide whether to buy or not certain products, the buyer moves around with his device oriented towards the products he sees and thinks of.

As soon as the application recognizes one of them, the buyer is able to see useful information displayed near that product. Beside the price, he may learn how many others have bought that product the same day.

If available, the application also shows the buyer a few other products that may be good alternatives. Without having to look them over around the shop, the buyer finds out their price and other meaningful details that he may use to make a decision. If not happy with a certain suggestion he can mark it accordingly and that particular product will not be shown again to him in a similar circumstance.

When the user decides to buy a product, he ticks the "I bought this" option that he has on the screen for each of the discovered products since the application was started (the list can be cleared at any time). He is also asked to rate that particular product. This way he gives back to the community by letting others know, indirectly, how popular some products are. His actions will influence the suggestions other will see when looking for products that he too bought or that are similar to those.

OTHER USES OF AR

Below are a couple of other scenarios where this technology could be successfully utilized in conjunction with social networks and the internet. All cases imply the use of a mobile device (tablet, smartphone, etc.) equipped with a camera and taking advantage of GPS data:

- Imagine a retailer who wants to bring the best of the online shopping experiences to brick and mortar stores. Every product package or image in the store, when recognized, can bring associated content from the retailer's website and render it directly on top of the product or image. This provides a customer more choice in the store by allowing him to visualize product information like videos, slideshows, sizes, colours, reviews – things he can find easily online, but in a much more fun and engaging way. The retailer can potentially drive a commercial transaction from within the app itself with the convenience of having the product delivered home. The American Apparel AR app demonstrated at Uplinq® 2012 is a good example of this new kind of shopping experience. Apps tied to print publications – magazine and catalogues – can also benefit from the cloud services as users could interact with every

page in the publication. These are just a couple of examples of what one could do with cloud recognition.

- You take a photo of one of your friends in the park. His face is either automatically recognized by software like Google Picasa [38] or you tag it yourself. Using this information and a connection to the Internet the AR software adds potentially useful information to the picture, like the friend's name or information aggregated from social networks, like Twitter or Facebook. This way, above him, in the picture, his Facebook status or latest Tweet can be displayed (e.g. „That's Mark right here listening to Rihanna").

- You take a photo and want to share it online. Allowing your device to use the embedded GPS enables the AR software to get your current location. By querying a web service that provides coordinates mapping, relevant details about the place can be automatically added to your picture, either as an overlay or as tags. This way your friends can find out the location (place, street, town; e.g. „Times Square, NY") or learn about interesting touristic sights that may be around the area. If by mapping your location the software cannot decide where to „place" you, it can offer you the option to pick the correct address manually.

- In conjunction with existing web services, software doing object detection and/or OCR can be employed in order to provide information about the things you capture in a picture or that you film. Any bit of information extracted from the imagery provided by your device (text, shapes) can be used to query the web services for more details. For example, pointing your device to a street could allow the software to detect popular buildings (images of buildings to be used for comparison could be grabbed from websites like Wikipedia based on GPS location) and overlay details about them. Moreover, names of stores, institutions or even labels from commercials being displayed outside (on walls, panels or poles) could be recognized as text that, searched online, could return useful information, like the schedule or the role in the case of an institution or what you could buy in the case of a store.

- You take a fieldtrip and want to share your experience as you explore the nature. You may film; take photos or record audio during the trip. As soon as you do, the media you captured is tagged with your current location and is uploaded to your online sharing account. Querying web services providing touristic information, the media is augmented with historical, geological or geographical details. This way, instead of a watching a plain photo or video, your friends can also learn something about the things you just experienced. As your device uploads content online your friends can be instantly notified via popular social networks (by using your Facebook's status or posting on their walls or by tweeting from your account), so they can be a part of your experience at the same time with you.
- Entertainment – all kinds of multimedia content can be delivered into the user's world, providing an exciting way of experiencing things like games, movie trailers and so on.

- Customer support – postal or shipping companies can provide an easy way for their potential clients to learn how much they have to pay to send a specific item/package and the right size of envelope/package needed. The client will simply point his object to the camera to allow the AR application to “scan” it, analyse its shape and estimate dimensions or volume, and then offer the appropriate information overlaid to the object.

5. REFERENCES

[1] http://www.forbes.com/sites/quora/2013/01/07/how-many-things-are-currently-connected-to-the-internet-of-things-iot
[2] http://imsresearch.com/press-release/Internet_Connected_Devices_Approaching_10_Billion_to_exceed_28_Billion_by_2020
[3] https://developer.qualcomm.com/blog/next-app-os-web-browser
[4] http://ismar.vgtc.org
[5] http://www.sensorsmag.com/electronics-computers/news/wikitude-takes-augmented-reality-web-9924
[6] http://thenextweb.com/insider/2012/05/16/wikitude-takes-its-augmented-reality-beyond-apps-and-direct-to-the-mobile-web/
[7] http://www.strands.com
[8] Buraga, C.S., Alboaie, L., Mihăilă, A. E., Vlad, G. A., Vascu, I., Chetreanu, G. 2012. PERE: A Location-based personal Semantic Web Recommender.
[9] http://blog.massivehealth.com
[10] http://www.hitl.washington.edu/artoolkit/documentation
[11] http://en.wikipedia.org/wiki/Affine_transformation
[12] http://www.se.rit.edu/~jrv/research/ar/thesis.html
[13] Koenderink, J. J. and A. J. van Doorn (1991). "Affine Structure from Motion." Journal of the Optical Society of America A **8** (2): 377-385
[14] Ullman, S. and R. Basri (1991). "Recognition by Linear Combinations of Models." IEEE Transactions on Pattern Analysis and Machine Intelligence **13** (10): 992-1006
[15] http://www.infi.nl/blog/view/id/56/Marker_Detection_for_Augmented_Reality_Applications
[16] Martin Hirzer, Marker Detection for Augmented Reality Applications, 2008, http://studierstube.icg.tu-graz.ac.at/thesis/marker_detection.pdf
[17] J.C. Clarke, S. Carlsson, and A. Zisserman. Detecting and tracking linear features efficiently, 1996.
[18] http://en.wikipedia.org/wiki/RANSAC
[19] http://www.40noll.com/ar/abstract.asp
[20] http://en.wikipedia.org/wiki/Feature_extraction
[21] http://www.consortium.ri.cmu.edu/projFeatFramework.php
[22] https://developer.qualcomm.com/blog/introducing-fastcv-computer-vision-technology-tuned-mobile
[23] http://www.augmentedplanet.com/2012/10/readers-choice-awards-2012-results/
[24] https://developer.qualcomm.com/blog/vuforia-sdk-20-augmented-reality-power-cloud
[25] https://developer.qualcomm.com/blog/there-are-now-over-3000-apps-using-vuforia
[26] http://www.wikitude.com/external/doc/alr/index.html
[27] http://couchdb.apache.org
[28] http://en.wikipedia.org/wiki/HTML5tep3
[29] http://www.techradar.com/news/internet/web/html5-what-is-it-1047393
[30] http://www.technewsdaily.com/16388-what-is-html5.html
[31] http://www.1stwebdesigner.com/design/html5-introduction
[32] http://www.w3.org/Style/CSS/current-work
[33] http://en.wikipedia.org/wiki/CSS3#CSS_3
[34] http://webdesign.about.com/od/css3/a/aa061206.htm
[35] http://en.wikipedia.org/wiki/JavaScript
[36] https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript
[37] http://www.json.org/