UNIT IV INTRODUCTION OF AUGMENTED REALITY

System Structure of Augmented Reality-Key Technology in AR-- AR software development - AR software. Camera parameters and camera calibration. Marker-based augmented reality. Pattern recognition. AR Toolkit

Augmented Reality – Introduction

Augmented Reality (AR) is a general term for a collection of technologies used to blend computer generated information with the viewer's natural senses. A simple example of AR is using a spatial display (digital projector) to augment a real world object (a wall) for a presentation. Augmented reality technology was invented in 1968, with Ivan Sutherland's development of the first head-mounted display system. However, the term 'augmented reality' wasn't coined until 1990 by Boeing researcher Tim Caudell. The term “augmented reality,” as well as the first true device of this kind, was created back in 1990 by Boeing researcher Tom Caudell and his colleague David Mizell. Augmented reality was first achieved, to some extent, by a cinematographer called Morton Heilig in 1957. He invented the Sensorama which delivered visuals, sounds, vibration and smell to the viewer.

Just two years later, Louis Rosenberg created Virtual Fixtures, the first AR system that was used by the U.S. Air Force. The device made use of a heads-up display (HUD) connected to two physical robot arms that the user could move through an upper-body exoskeleton that acted as a controller. The user saw the computerized robot arms in his visor, together with other computer-generated virtual overlays that simulated objects, barriers or guides existing in the real world. Today, in less than 30 years, AR technology has made a huge leap forward both in terms of performance and usability as well — so much that these clunky early models look like hilarious sweded movie cardboard equivalents of the modern devices! Augmented reality (AR) is a technology that lets people superimpose digital content (images, sounds, text) over a real-world environment. AR got a lot of attention in 2016 when the game Pokémon Go made it possible to interact with Pokémon superimposed on the world via a smartphone screen.

Augmented reality has been a hot topic in software development circles for a number of years, but it's getting renewed focus and attention with the release of products like Google Glass - Wearers communicate with the Internet via natural language voice commands Augmented reality is a technology that works on computer vision based recognition algorithms to augment sound, video, graphics and other sensor based inputs on real world objects using the camera of your device. It is a good way to render real world information and present it in an interactive way so that virtual elements become part of the real world. Augmented reality displays superimpose information in your field of view and can take you into a new world where the real and virtual worlds are tightly coupled. It is not just limited to desktop or mobile devices. Google Glass, a wearable computer with optical head-mounted display, is a perfect example. A simple augmented reality use case is: a user captures the image of a real-world object, and the underlying platform detects a marker, which triggers it to add a virtual object on top of the real-world image and displays on your camera screen.

Real-World Examples

AR applications can become the backbone of the education industry. Apps are being developed which embed text, images, and videos, as well as real–world curriculums. Printing and advertising industries are developing apps to display digital content on top of real world

magazines. With help of AR, travelers can access real-time information of historical places just by pointing their camera viewfinder to subjects. AR is helpful in development of translation apps that can interpret text in other languages for user. Location based AR apps are major forms of AR apps. Users can access information about nearest places relative to current location. They can get information about places and choose based on user reviews. With the help of Unity 3d Engine, AR is being used to develop real-time 3D Games.

The Opportunity

It is estimated that 2.5 billion AR apps will be downloaded annually and will generate revenue of more than $1.5 billion by 2015. This is because AR apps will not be limited to conventional mobile apps. There will be new markets like Google Glass which will open more forms of development and use.

Development

To develop augmented reality apps … First - need to choose development tools. There are two major forms of augmented reality, marker- based AR and marker-less AR. A marker-based AR works on concept of target recognition. The target can be 3D object, text, image, QR Code or human-face called markers. After detection of the target by AR engine, you can embed the virtual object on it and display it on your camera screen. Qualcomm Vuforia SDK is our recommended framework to develop native apps. Marker-less AR, also known as location-based AR, uses GPS of mobile devices to record the device position and displays information relative to that location. Some of the examples of marker-less AR are apps like Layar and Wikitude that let you view information of nearby restaurants and other establishments.

Barriers - need to cross

Although going forward AR seems to have a huge potential market, there are some factors which could slow down mass adoption of augmented reality. Some of the factors are:

- Public Awareness and reach of Mobile AR
- Technological Limitations
- Addressing Privacy Issues
- Mobile Internet Connectivity in Emerging Markets

AR can be considered a technology between VR and telepresence. While in VR the environment is completely synthetic and in telepresence it is completely real, in AR the user sees the real world augmented with virtual objects. A telepresence robot is a remote-controlled, wheeled device that has wireless internet connectivity. Typically, the robot uses a tablet to provide video and audio capabilities. TelePresence robots are commonly used to stand in for tour guides, night watchmen, factory inspectors and healthcare consultants. Examples of telepresence include remote manipulation of probes in the deep sea, working with dangerous chemicals, controlling operations on a space probe, or even manipulating surgical instruments just a few feet away.

I. System Structure of Augmented Reality

Augmented reality is achieved through a variety of technological innovations; these can be implemented on their own or in conjunction with each other to create augmented reality. They include:

General hardware components are the processor, the display, the sensors and input devices. Typically a smartphone contains a processor, a display, accelerometers, GPS,

camera, microphone etc. and contains all the hardware required to be an AR device. Displays – while a monitor is perfectly capable of displaying AR data there are other systems such as optical projection systems, head-mounted displays, eyeglasses, contact lenses, the HUD (heads up display), virtual retinal displays, EyeTap (a device which changes the rays of light captured from the environment and substitutes them with computer generated ones),Spatial Augmented Reality (SAR – which uses ordinary projection techniques as a substitute for a display of any kind) and handheld displays.

Sensors and input devices include – GPS, gyroscopes, accelerometers, compasses, RFID, wireless sensors, touch recognition, speech recognition, eye tracking and peripherals. Software – the majority of development for AR will be in developing software to take advantage of the hardware capabilities. There is already an Augmented Reality Markup Language (ARML) which is being used to standardize XML grammar for virtual reality. There are several software development kits (SDK) which also offer simple environments for AR development. ARML - is a data standard to describe and interact with AR scenes. ARML consists of XML grammar and ECMAScript (European Computer Manufacturers Association). XML grammar is used to describe the location and appearance of virtual objects in the scene. ECMAScript binding is used to allow dynamic access to the properties of the virtual objects, as well as event handling. ARML focuses on visual augmented reality (i.e. the camera of an AR-capable device serves as the main output for augmented reality scenarios). ECMAScript is a general-purpose programming language, standardised by Ecma International according to the document ECMA-262. It is a JavaScript standard meant to ensure the interoperability of web pages across different web browsers. ECMAScript is commonly used for client-side scripting on the World Wide Web, and it is increasingly being used for writing server applications and services using Node.js. Ecma International is a nonprofit standards organization for information and communication systems.

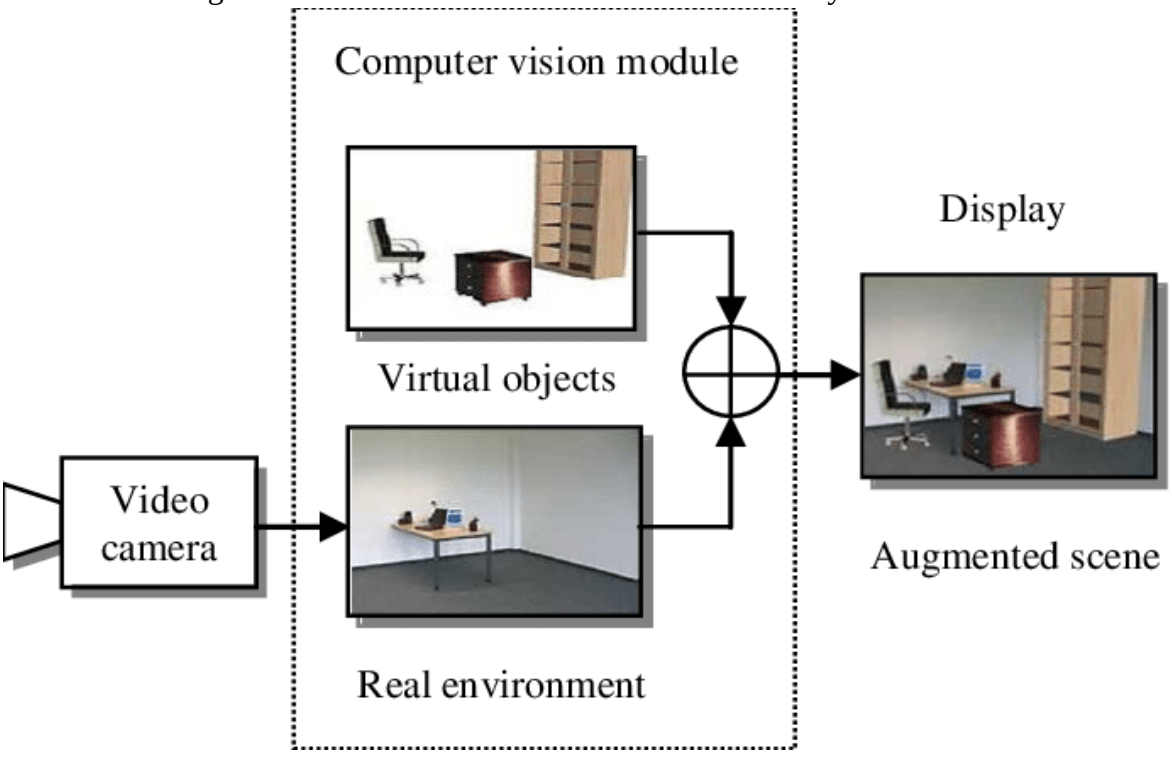


Fig.4.1 System Structure of AR

The blend of directed-perception (from physical world) and computer-mediated perception needs to be in the real-time - to provide a great AR experience, --- Sensors.

Sensors connect the physical world to the computer-mediated box. It can be a camera, accelerometer, GPS, compass or microphone. Sensors make the first building-block of AR architecture. Sensors can be classified into two categories: (1) The one measuring a physical property of the environment that is not applicable to a human sense. e.g. Geo-location. (2) The ones capturing a physical property, directly detectable by human sensing capabilities. e.g. Camera. Some data of context analyzer and even some data of sensors is transferred to the brain of the system, the MAR Execution Engine. The main job of the engine is to verify if the conditions established by the AR designer and expressed in the MAR scene are met. It is a relatively complex part of AR architecture and has an orchestration role receiving messages from sensors, end-users, services, and rendering to the end-user. The results produced by the MAR execution Engine could be presented to the end-user by using specific displays, such as screens, loud-speakers, vibration devices. The user can also interact with the system using UI components.

User

AR technology is to provide artificial stimuli to cause the users to believe that something is occurring in the virtual world. Take Tesco store finder created by Junaio AR browser as an example: the purpose of this application is to provide the users with the awareness of Tesco's location, opening time, website and further information. Users are playing an important role in what takes place in AR architecture. Normally, AR architecture could happen in the mind of one user. But sometimes it could be two or many. Construct 3D - a geometric construction AR system aimed to promote students spatial capability due to the fact that they could view geometric entities in different sides. Two or many users could use an electric pen to modify all geometric entities, find the geometric relationship and work on the construction together. The concept of collaborative augmented reality is based upon two or many AR users. AR tennis is another application engaged with two users: two players could sit across the table from one another and hold their phone to view a virtual court overlaying the real world background and play the tennis game. Many AR researches focus on the normal users without particular group of people. Some of research designed AR application for children with autism and cognitive disability. ‘user’ - refer to an individual who manipulates and controls, the immediate intended beneficiary of an AR system. For example, doctors could use augmented reality as a visualization aid and possibly collect 3-D dataset of a patient in real time during the surgery. AR system brings the benefits for both doctors and patient. However, the AR user should be the surgeon who watches and controls the AR system rather the patient.

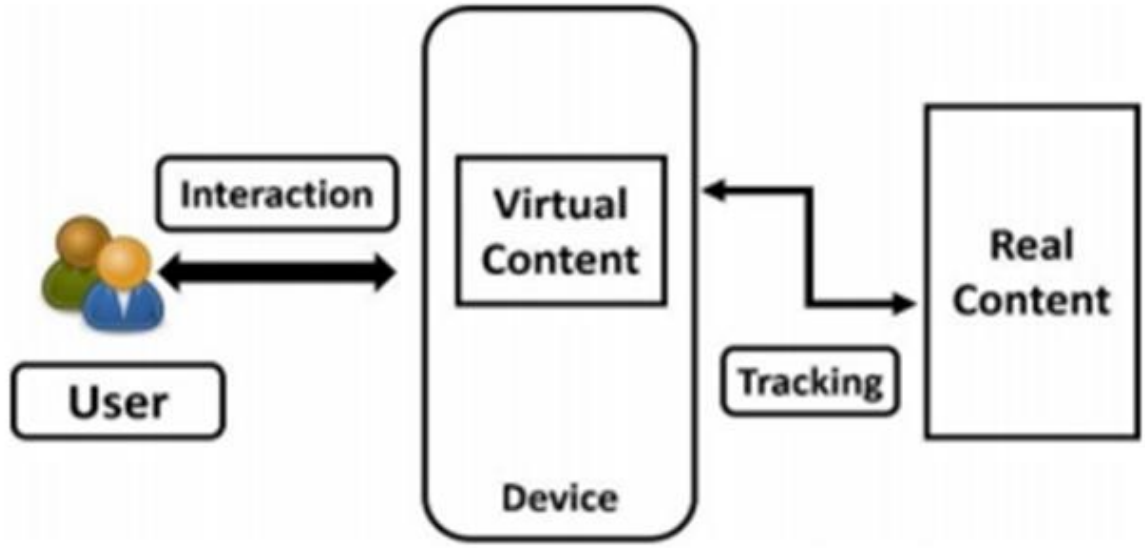


Fig. 4.2 Augmented Reality Architecture comprised by six elements

Interaction

Interaction is composed of two components including inter- and action. Inter means the state between or among things. Action means there is an influence and something that has been done. Therefore, interaction can be simply speaking that one entity does something and the other entity responds in some way. In the user-based augmented reality (AR) system, the process of interaction in this research mainly concentrates on a trigger caused by users and the response of AR system which can occur between users and AR device or users and virtual content. For example, if people try to use Tesco store finder by Junaio AR browser to find the location of the nearest Tesco, they have to launch a Tesco App or channel first. And then, people may adjust the position of the mobile device (e.g. Iphone or Ipad) to see the overlaid virtual bubble. The action of adjusting the AR device's physical position can be described as the interaction between users and AR device interaction. This action will end in a response of identifying the virtual target store on the device. Then, if people want to get more information about the particular Tesco store (e.g. opening time, distance or phone number), they need to click a virtual icon that visually indicates the store's information. After that, some more overlaid information will be presented in another pop-up slide. The action of clicking the virtual Tesco bubble means users and virtual content interaction. The response of new pop-up page implies the process of interaction has been done. However, in some of the particular AR scenario, the process of interaction between users and AR device or users and virtual content doesn't exist anymore. National geographic demonstrated a spectacular view of national geographic content on a large screen that visitors could see themselves along with the augmentations in their world. Users do not interact with the augmented reality device (the big screen) and the virtual national geographic content in this scenario.

Device

The term of device means the carrier or object could acquire physical world information and provide the compelling augmentation. It could be the mobile device, desktop computer, and big screen with projector or etc. three hardware functions in all AR device - sensors, processors and displays. Sensors recognize the state of the physical world which the AR system needs to be deployed. For example, a camera, one of the most popular AR sensors, could capture the physical world image and provide information to the AR users.

GPS or other compass system aims to identify the location and orientation of user. A processor processes the sensors' information and generates the signals required to drive the display. Very often, the AR system will rely upon not just the processor on the device, but the processor on the server as well. A display will show the coexistence that users could sense the combination of physical world and virtual world. Based on these requirements of functions, the smart mobile phone or tablet seems to be the appropriate AR device compromising by a camera to capture, processors to process and a screen to display. The mobile device held by one hand could run different applications, which is moveable, easy to use and accessible from anywhere and anyplace.

Virtual content

Virtual content means the digital information presented by AR device, which plays the most important role in the AR architecture. The modality of virtual content could be 3D animation, 2D image, text, website, audio information or even vibration. AR users will not concern too much on devices, but will be attracted by different virtual content. Participants often strongly express curiosity to what a digital device could provide but rarely if ever affection to the device itself. A key feature of virtual content is that the virtual information can be changed dynamically. Going back to the Tesco example, when users use the app and move around, the virtual content (Tesco bubble) will pop up automatically. An icon visually indicating a real-world store can be clicked and more Store information will be presented. This additional information like the opening time website or the video instruction replaces the previous virtual bubble. The content of virtual information has been changed easily and completely.

Examples

Scanning a QR code using phone's camera provides additional information (so, AR) on screen. Google Glass and other head-up displays (HUD) like Vuzix Waveguide Lens put Augmented Reality directly into the glasses. These glasses could be used as reminders for patients undergoing medication. Real time battlefield data could be available to soldiers wearing these. We are also familiar with the various filters on Snapchat and Instagram, an aspect of AR. In the Netherlands, an application called Layar is available for download, which uses the phone's camera and GPS capabilities to gather information about our surroundings. We could point at a building and enquire about its history, whether it's on sale and more. AR Defender 2 is a mobile game for iOS users that helps you attain amazing experience, by turning any real world area into a virtual battlefield. Niantic has already achieved a lot by developing Ingress and Pokémon Go. Apps like Augment, is helpful for designers that allows users to upload 3D models and visualize them in a physical space.

Real content

Real content is the real-world information directly presented by device without any rendering, which includes geographic location, physical objects and real-world environment. Taking Tesco app example again, AR devices not only display the virtual Tesco bubble (virtual content), but present the users' location information surrounding the real-time environment, which means real content. However, while users look through the AR device, the real content will be more or less hidden. For example, Word lens AR translator generates the virtually translated words, which replace the real-world words. Users have to remove the AR device if they need to see the original words.

They cannot see both the virtual and real content simultaneously. Obstruction of real content is the intrinsic risk of augmented reality.

Tracking

Tracking describes the way of generating virtual content based on the real content, comprising three different features: synchronicity, antecedent and partial one to one. Due to the changes of the real content, an AR virtual counterpart has to be updated synchronously. For example, Word lens is an AR translation application that scans foreign text and displays the test translated in real time. Once the user changes his or her point of view to another word, the displayed translation on the device rapidly changes in the same time. If the process of generating virtual information is delayed for a long time, viewers are unable to obtain the useful information. The feature of antecedent means the real content (physical text) exists or happens before the virtual content (the translated digital word). If the virtual content is created before the real-world content, the virtual element is meaningless because it has no real world interpretation. Partial one to one describes another tracking feature of augmented reality. There is one and only one real content to correspond with the virtual content. However, there might be one or more than one piece of virtual information to correspond with the real-world content. That means the real content can be superimposed to different modality of virtual content. AR users could be one or many who might have interacting with the device or virtual content by adjusting or clicking. Virtual content is the additionally computer-generated information displayed on the AR device via an array of tracking transformation based on real-world counterpart. The AR architecture could bring benefits to AR designers and provide a more explicit basis on which to articulate AR criteria, classification and function.

Why Augmented Reality is Important

Development of AR technology is set to revolutionize industries from retail to military to education to tourism and transform the way that is been interacted with the digital world every day. Augmented reality has many uses in different fields like in archaeology, architecture, visual arts, commerce, education, video games and military training etc. Some applications of AR are - AR is being used to aid research in archaeology. AR can be used to recreate different structures and overlay them on the real environment so that researchers can study it correctly.

Importance

AR applications in smartphones include Global Positioning System (GPS) to locate the person’s location and its phone’s inbuilt compass to find device orientation. Augmented reality can be used in the field of tourism to enrich visitor’s experience during visits like the Eiffel tower has an AR app that can show - throughout history when it was being built. and the list goes on. that’s why AR and VR companies have raised more than $3 billion in 2017 in funding, thus 2018 has been doubled the year when AR goes mainstream, $45 billion globally 2019. The augmented reality (AR) and virtual reality (VR) market is set to grow by USD 162.71 billion, progressing at a CAGR of almost 46% during 2021-2025. It is sure that in the coming years it will change the way that technology is being looked and improve the integration of technology in our daily lives.

II. Key Technology in Augmented Reality

The Augmented Reality Technology is an important branch of Virtual Reality Technology. On the basis of virtual reality technology, augmented reality technology uses computer graphics technology and visualization technology to superimpose virtual images generated by computer operations to real pictures. Various technologies are used in augmented reality rendering, including optical projection systems, monitors, handheld devices, and display systems, which are worn on the human body. A head-mounted display (HMD) is a display device worn on the forehead, such as a harness or helmet-mounted.

- Intelligent display technology,
- 3d registration technology and
- intelligent interaction technology

constitutes the core technology circle of AR and play an important role in the development of AR.

Intelligent display technology

According to relevant data, more than 65% of the information acquired by human beings comes from their own vision, which has become the most intuitive way for human beings to interact with the real environment. With the development of intelligent display technology, augmented reality becomes a possibility, which is pushed to a new height by the various kinds of display devices generated based on intelligent display technology. Specifically, there are three main categories of display devices that occupy an important position in the field of AR technology today. First, helmet display (HMD *head mounted display*) was born in 1968. The optical perspective helmet display developed by Professor Ivan Sutherland makes it possible to superimpose simple graphics constructed by computers on real scenes in real time. In the later development, optical perspective helmet-mounted display and video perspective helmet-mounted display constitute the backbone of helmet-mounted display. Second, handheld device display, relying on the augmented reality technology of handheld display, handheld device display is very light, small, especially the popularity of smart phones, through video perspective to the use of augmented reality technology to present. Third, other display devices, such as PC desktop displays, match the real-world scene information captured by the camera to a three-dimensional virtual model generated by the computer and are ultimately displayed by the desktop display.

3D registration technology

As one of the most critical technologies in the augmented reality system, 3d registration technology enables virtual images to be superimposed accurately in the real environment. The main flow of 3d registration technology has two steps. First, determine the relationship between the virtual image, the model and the direction and position information of the camera or display device. Second, the virtual rendered image and model are accurately projected into the real environment, so the virtual image and model can be merged with the real environment. There are various ways of 3d registration, such as

- the registration technology based on hardware tracker,
- the 3d registration technology based on computer vision,
- the 3d registration technology based on wireless network and
- the mixed registration technology, among which the former two are the most popular.

For the three-dimensional registration technology based on computer vision, it sets the reference point to realize the determination of the direction and position of the real scene by the camera or the display.

Intelligent interaction technology

Intelligent interactive technology is closely related to intelligent display technology, 3d registration technology, ergonomics, cognitive psychology and other disciplines. In AR systems, there are a variety of intelligent interactions, including hardware device interactions, location interactions, tag-based or other information-based interactions. With the development of intelligent interaction technology, augmented reality not only superimposes virtual information to real scenes, but also realizes the interaction between people and virtual objects in real scenes. This interaction is based on the fact that people give specific instructions to the virtual object in the scene, and the virtual object can make some feedback, thus enabling the audience of the augmented reality application to achieve a better experience.

III. AR software development and AR software

AR Software

AR software works in conjunction with devices such as tablets, phones, headsets, and more. These integrating devices contain sensors, digital projectors, and the appropriate software that enables these computer-generated objects to be projected into the real world. Once a model has been superimposed in the real world, users can interact with it and manipulate the model. These solutions have additional uses aside from placing a 3D model into the real world. AR is commonly used for entertainment purposes—specifically gaming. This software can also be used to display contextual information. Users can point the hardware's camera display at an object to display valuable data.

Why Use AR Software?

As AR is still a young technology, it provides certain advantages to businesses that other software cannot offer. The following are just a few of the benefits to use AR software in the business.

Product view – AR technology allows potential customers to view and interact with the product or service before purchasing. This can enable them to make better-informed decisions.

Enhance content – AR technology allows users to embed various types of data onto content. People can point their device at a real-life object to learn whatever kind of information is necessary, instead of needing to search for it elsewhere.

Training – AR solutions enable users to train employees more thoroughly than they can get through documentation and meetings. This software allows for trainees to learn job responsibilities by fully visualizing them, instead of just reading about job duties.

Productivity – This software enables users to improve workflow and processes at their business. This is particularly true for manufacturing-based organizations. Factory line workers can spot potential dangers quicker, along with accessing necessary resources.

Engage your audience – Consumers are inundated with print and television advertisements for various products and services, to the point where they don't pay much attention to them. Inserting augmented reality into advertisements will catch the eye of your target demographic.

Who Uses AR Software?

AR software can be utilized by users in a number of different fields, such as:

Retail – Users in the retail industry can leverage AR technology so consumers can virtually test out products before they make a purchase. For example, AR retail applications allow users to upload a photo of themselves and visualize what a particular piece of clothing would look like on their body. Shoppers could also use these kinds of applications to visualize what a piece of furniture would look like in their house.

Education – AR technology is being increasingly used in the classroom to supplement lessons. For example, if a teacher was doing a lesson on astronomy, AR software could project a map of the solar system so students could visualize what they are learning about.

Repair and maintenance – Employees performing manual labor can wear AR glasses to help with repair and maintenance jobs. AR software can be used to project valuable data and inform the user where a certain part is supposed to go.

Medical – Doctors, particularly surgeons, can use AR technology for training purposes. All the documentation and videos out there are not realistic enough to prepare a surgeon for what surgery is really like. AR technology can help trainee surgeons visualize what the actual act of surgery would be like.

Kinds of AR Software

However, the following are some of the main types of AR software on the market now:

AR visualization software – This type of software enables organizations to create immersive experiences for consumers to interact with. AR visualization software users can upload 3D content and scale the image, adjust the color, and incorporate the additional details needed to give the best user experience possible.

AR content management system (CMS) – An AR CMS lets users bulk upload raw 3D content that will eventually become the basis for AR experiences. This content can be managed and edited within the platform.

AR SDK – These tools allow users to build digital objects that will blend into the real world that will eventually become fully fledged AR experiences.

AR WYSIWYG editor software – This software enables users with limited to no coding background to create customized AR experiences. These tools have drag-and-drop capabilities that let users upload 3D objects and drop them directly into previously designed scenes.

AR game engine software – These solutions give game developers the framework for creating AR video game experiences. Using AR game engine software, users can create and edit 3D characters that can interact with the real world.

AR training simulator software – AR training simulator software leverages AR technology to train employees for certain jobs.

Industrial AR platforms – These solutions are typically used by organizations in the industrial field. These tools include interactive AR content that improves these employees' productivity, effectiveness, and safety.

AR Software Features

Content management – Many AR solutions, regardless of the specific category they fall into, provide users with the ability to store and manage their content. This can range from raw 3D content that will serve as the basis of an AR experience to content that has already been designed.

Editing – AR solutions should allow for users to edit the 3D model they upload into the platform. Users can scale the image, adjust the color, and incorporate any additional details needed.
Hardware integration – In order to provide the intended AR experience for a consumer, the software must integrate with devices that support AR software. This includes glasses, Android and Apple mobile phones, and tablets.
Drag-and-drop – Some AR development solutions are designed to be user-friendly for those with little to no coding experience. Tools like this offer a WYSIWYG editor, which allows users to upload 3D objects and insert them into previously designed scenes so that they eventually become AR experiences.
Additional AR Features
Analytics – Some AR tools, such as products in the AR visualization software space, will provide analytics capabilities for users. This lets businesses see how consumers interact with the 3D object within AR mobile applications, which should be supported on both Apple and Android devices.
Upload content – AR software products allow businesses to upload 3D content necessary for their specific business purposes. This is particularly relevant for AR training simulators, as businesses need to ensure the software will support the content needed for trainees to learn the job at hand.

Trends Related to AR Software
AR advertising — Various brands are beginning to introduce augmented reality into their promotions. AR can enhance a user's experience with your brand. Entertainment companies will likely avail themselves of this technology, so they can bring various elements of a show or movie to life.
Health care — Not only can AR technology help to train surgeons, but it can assist them once they are already well-versed in their work. Some surgeons have already used AR while operating on human hearts, so they can visually see the clogged vessels they are working on. AR will likely continue to grow in the healthcare field, as it can help caregivers make the best-informed decisions in life-or-death situations.
Android and Apple mobile sales — Smart phones are among the devices that can support AR technology. As AR software becomes more and more common in the marketplace, mobile phone manufacturers will likely begin to compete to build the phone best equipped to support AR. Android and Apple phones will likely go head-to-head with each other.
Wearable AR — Developers have begun to set their sights on wearable AR technology, specifically glasses. And as this continues to grow, developers are working to make these glasses more ergonomic. This technology is anticipated to become smaller, more form-fitting, and better attuned to human senses.
Opening field of view — As AR glasses are on the rise, developers are also working to open up the field of view. Most glasses limit the field of view to about 45–50 degrees, compared to the human eye's 120 degree field of view. AR developers are working to close that gap.

Potential Issues with AR Software
Cost

One of the biggest factors that has hindered AR from becoming mainstream is the cost. It can be very expensive to purchase the hardware to support AR technology. Streaming the

content is also very costly. Content for these solutions needs to be streamed in a very high resolution and rendered at a high refresh rate. This content also requires a large bandwidth for streaming. All these factors add up, making consumers wary of adopting AR.

Accessibility and education

Due to the cost, AR technology is not too accessible to the masses. Since very few people are exposed to this software, it is hard for them to conceptualize the wide-ranging uses that AR can offer. Unless developers change the user experience and the messaging around this technology, it will be difficult to get past this hurdle. There are two broad classes of AR apps: - marker-based apps and location-based apps. Marker-based apps use predefined markers to trigger the display of AR overlays on top of the image. Location-based apps use GPS, accelerometer, or compass information to display AR objects on top of physical ones.

- To choose an AR SDK, the most important criteria to consider are:
    - cost,
    - supported platforms,
    - image recognition and tracking support,
    - Unity support,
    - OpenSceneGraph support,
    - GPS, etc.

Augmented reality (AR) has become the new trend in the digital world and can hardly meet a person who is not familiar with it after the boom that Pokemon Go brought into the lives of the average mobile user. Though many people consider AR to be only an entertainment technology, it's actually widely used in multiple industries like healthcare, e-commerce, architecture and many others. The potential of AR is seamless and brands are already utilizing this technology in their business to provide a brand new user experience. Companies implement AR to create product demos, interactive advertising and provide real-time information to customers. It was proved that when people touch or interact with a product, they are more likely to buy it due to the emotional bond established. According to a Statista forecast, the market of augmented and virtual reality is expected to reach the size of $215 billion in 2021 end. Being a rapidly growing market with huge potential, AR attracts both huge corporations like Google, Apple, Facebook, etc., as well as smaller businesses.

AR Software Development

There are many types of Augmented Reality applications exist. Before starting the development of a augmented reality app - choose between two broad categories: location apps and marker-based apps.

Marker-based applications

Marker-based apps are based on image recognition. They use black and white markers as triggers to display AR content. To see the augmented component, you have to point the camera on a marker's position anywhere around you. Once the device recognizes the marker, an app overlays the digital data on this marker and you can see the augmented object. When building a marker-based app, the images or their descriptors should be provided beforehand to simplify the process of searching them when the camera data is being analyzed. In other words, the objects are already hard-coded in your app, so they are easier to detect. It's no

wonder that the majority of AR apps are marker-based. They are especially popular in advertising.

Location-based applications

Location-based AR apps work without markers. They detect the user's position with the help of a GPS, an accelerometer, or a digital compass and overlay the augmented reality objects on top of real physical places. The most famous location-based app is surely Pokemon Go. These apps can send notifications to the user based on their location to provide them new AR content related to a given place. For example, an app could give recommendations about the best restaurants nearby, and show how to get there. As an additional example, an app could help you find your car inside a huge parking using GPS.Main Criteria to Choose an Augmented Reality SDKs. When it comes to choosing a development kit, it's easy to get frustrated by the number of tools available. In order to pick the SDK that best suits the project, make sure it supports all the features - app requires.

Main points to consider -

Cost

Pricing is the first distinguishing mark of an AR SDK. For those who want to try AR development for the first time, the best options are free open-source AR SDKs, which are open to contributions and can be extended with new features proposed by developers. Paid SDKs in most cases offer several pricing plans, depending on the user's needs. As it happens, free tiers have limited possibilities and are meant to be a "demo version" of the full product. Building a complex app with large, dynamic content will likely require a commercial license.

Platforms

If the plan is to develop app for iOS or Android, there won't be any problems when choosing an augmented reality toolkit, since nearly all of them support them. Meanwhile, the choice of tools that are compatible with Windows or macOS is rather small. Still, - can build your app for Windows computers or smart phones using augmented reality development kit, supporting the Universal Windows Platform (UWP).

Image recognition

This feature is a must-have for any AR app as it allows to identify objects, places and images. To this aim, smart phones and other devices use machine vision together with camera and artificial intelligence software to track images that can be later overlayed with animations, sound, HTML content etc.

3D recognition and tracking

3D image recognition and tracking is one of the most valuable features of any AR SDK. Due to the tracking, an app can "understand" and enhance the large spaces around the user inside of large buildings such as airports, bus stations, shopping malls, etc. Applications supporting it can recognize three-dimensional objects like boxes, cups, cylinders, toys etc. Currently, this technology is commonly used in mobile games and e-commerce.

Unity support

Unity is known to be the most popular and powerful game engine worldwide. Though it's usually used for developing computer games, it can also be utilized for making AR apps with powerful effects. Whether you are going to create a cutting-edge experience or extend a more traditional idea with new techniques, multipurpose tool like Unity allows you to implement both.

OpenSceneGraph support

OpenSceneGraph is an open source 3D graphic toolkit (application Programming interface). It's used by app developers in such domains as computer games, augmented and virtual reality, scientific visualization and modeling.

Cloud support vs local storage

When developing AR mobile applications, you have to decide whether user data will be stored locally or in the cloud. This decision is mostly driven by the number of markers you are going to create. If the plan is to add a large number of markers to the app, consider storing all this data in the cloud, otherwise the app will use much storage on the device. Furthermore, having an idea of the number of markers the app uses also matters because some augmented reality SDKs support a hundred markers while others support thousands. On the other hand, storing markers locally (i.e., on-device) enables users to run the augmented reality app offline, which could be convenient as Wi-Fi or mobile-data is not available.

GPS support (geolocation)

If the aim is to create a location-based AR application, geolocation is a fundamental feature that must be supported by the AR tool that is used. GPS can be used both in AR games like Pokemon Go as well as in apps made to overlay data on some nearby locations (for example to find the nearest restaurant).

SLAM support

SLAM means Simultaneous Localization and Mapping. It is an algorithm that maps the environment where the user is located and tracks all of their movements. AR apps containing this feature can remember the position of physical objects within some environment and position virtual objects accordingly to their position and users movements. SLAM has huge potential and can be used in many kinds of apps, not only AR apps. The main advantage of this technology is the ability to be used indoors while GPS is only available outdoors.

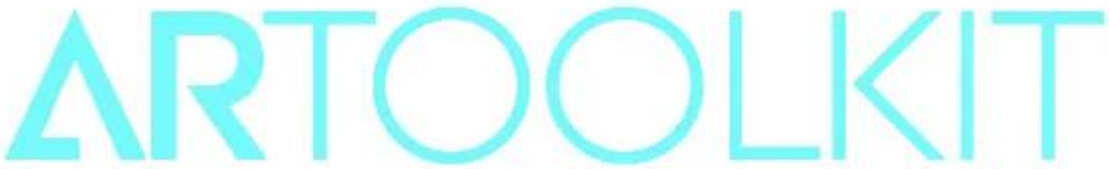
Augmented Reality SDK for Mobile Apps

- To create augmented reality app, list of popular tools are available on the market. These toolkits are considered to be the most relevant and appropriate based on the set of features they provide and their value for money. Some of them are free.
    - Vuforia – best for Marker-based apps
    - ARToolKit - best for Location-based apps
    - Google ARCore - best for Marker-based apps
    - Apple ARKit - best for Marker-based apps
    - Maxst - best for Marker-based apps
    - Wikitude - best for Marker-based apps

- **Vuforia** is a leading portal for augmented reality application development that has a broad set of features.

  Vuforia augmented reality SDK:

- Recognizes multiple objects including boxes, cylinders, and toys as well as mages.
- Supports text recognition including about 100,000 words or a custom vocabulary.
- Allows creating customized VuMarks, which look better than a typical QR-code.
- Allows creating a 3D geometric map of any environment using its Smart terrain feature
- Turns static images into full motion video that can be played directly on a target surface.
- Provides a Unity Plugin.
- Supports both Cloud and local storage.
- Supported platforms: iOS, Android, Universal Windows Platform, Unity. Pricing: free version, classic version - $499 one time, cloud - $99 per month and Pro version for commercial use.



- **ARToolkit** is an open-source tool to create augmented reality applications.

  Even though it's a free library, it provides a rather rich set of features for tracking, including:

- Unity3D and OpenSceneGraph Support.
- Supports both single and dual camera.
- GPS and compasses support for creation of location-based AR apps.
- Possibility to create real-time AR applications.
- Integration with smart glasses.
- Multiple Languages Supported
- Automatic camera calibration.
- Supported platforms: Android, iOS, Linux, Windows, Mac OS and Smart Glasses.
- Pricing: free

Google ARCore

- With two millions Android active users, Google could not miss the chance to give developers an opportunity to create AR apps on this operating system. That's how **Google ARCore** appeared.
- This toolkit works with Java/OpenGL, Unity, and Unreal.
- It provides features such as:
- Motion tracking - ARCore can determine the position and orientation of the device using the camera and spot the feature points in the room. That helps to place virtual objects accurately.
- Environmental understanding - Due to the possibility of detecting horizontal surfaces, virtual objects can be placed on tables or on the floor. This feature can be also used for motion tracking.
- Light estimation - This technology allows the app to match the lighting of the environment and to light virtual objects so they look natural within the surrounding space. With the help of smart light tracking developers can now create very realistic objects.
- Supporting devices: Currently: Google Pixel, Pixel XL, Pixel 2, Pixel 2 XL, Samsung Galaxy S7-S8+, Samsung A5-A8, Samsung Note8, Asus Zenfone AR, Huawei P20, OnePlus 5 ARCore is designed to work on devices running Android 7.0 and higher.
- Pricing: free

- With iOS11, Apple introduced its own **ARKit**, announced during Apple's Worldwide Developers Conference in June 2017.
- Here are the features of Apple's augmented reality SDK for iOS:
- Visual Inertial Odometry (VIO) allowing to track environment accurately without any additional calibration.
- Robust face tracking to easily apply face effects or create facial expressions of 3D characters.
- Tracking the light level of environment to apply the correct amount of lighting to virtual objects.

- Detecting horizontal planes like tables and floors, vertical and irregularly shaped surfaces.
- Detecting 2D objects and allows developers to interact with them.
- Integration with third-party tools like Unity and Unreal Engine.
- Devices: iPhone 6s and 6s Plus, iPhone 7 and 7 Plus, iPhone SE, iPad Pro (9.7, 10.5 or 12.9) – both first-gen and 2nd-gen, iPad (2017),iPhone 8 and 8 Plus, iPhone X
- Pricing: free

- **MAXST** has two SDKs available: a 2D SDK for image tracking and a 3D SDK for environment recognition.
- Here is the list of features of the 3D SDK:
- MAXST Visual Simultaneous Localization and Mapping for tracking and mapping environments. When you track the surroundings , the map is automatically extended beyond the first view along with the move of the camera. Maps can be also saved for the later uses.
- Saving files created with Visual Simultaneous Localization and Mapping to render 3D objects wherever you like on it to create more immersive AR experiences.
- QR and barcode scanning.
- Extended image tracking and Multi-target tracking - can track the target as far as the camera can see it and can also track up to 3 images at the same time.
- Tracking and placing digital objects in relation to the plane.
- Unity plugin integration.
- Supported platforms: Android, iOS, Mac OS and Windows.
- Pricing: free version, Pro-One time fee - $499, Pro-Subscription - $599 per year, Enterprise version.

- **Wikitude** has recently introduced its SDK7, including support for simultaneous localization and Mapping.
- The tool provides currently the following features:
- 3D recognition and tracking.
- Image recognition and tracking.

- Cloud recognition (allows to work with thousands of target images hosted in the cloud).
- Location-based services.
- Smart glasses integration.
- Integration with external plugins, including Unity.

- Supported platforms: Android, iOS, Smart Glasses (currently Google Glass, The Epson Moverio BT-200, and the Vuzix M100).
- Pricing: Pro version - €2490 per year per app, Pro3D - €2990 per year per app, Cloud - €4490 per year per app, Enterprise version.
- Needless to say, augmented reality technology is trendy. Each new AR app launch causes waves of excitement.
- Therefore, savvy developers are trying to master this technology and launch their own AR apps.
- Now, developers have a wide choice of AR toolkits to create both marker-based and location-based apps.
- The first step to get started is picking up the augmented reality SDK most suited to comply with their requirements.
- Then compare features such as image and 3D recognition, storage possibilities, Unity and SLAM support, etc., for development teams to easily select the best toolkit for their future apps.

IV. Camera Parameters and Camera Calibration

Augmented Reality is used for a wide range of applications in computer vision such as computer-aided surgery, repair of complex machines, establishment modifications, interior or structural design. In Augmented Reality applications the user's view of the real world is enhanced by virtual information. This additional information is created by a computer which has a model of the real world and the model of some real world objects in which the user is located. These real objects are tracked, so the computer knows the location and rotation of them. The Augmented Reality system imagines a virtual camera in the virtual world which can see a range of virtual objects corresponding to real world objects. The additional virtual data is superimposed over these real objects. This visual enhancement can either have the form of labels, 3D rendered models, or even shading modifications. With the help of optical see through displays the user can see both the virtual computer-generated world on the screen and the real world behind it. In general these are displayed on an see through head mounted display (HMD) to get an Augmented Reality view.

These optical see through devices present a special challenge because the system has no access to the real world image data as at a video see through device. So the HMD represents a virtual camera for which several parameters must be accurately specified as well.

Necessary tasks needed for calibrating virtual cameras

- A virtual camera defines a projection of the virtual 3D world to the 2D image plane.
- As shown in figure 4.3 the user sees the computer generated 2D image appearing in his HMD about one meter in front of his face.
- The virtual world objects are registered in 3D.
- In order to see the right objects at the desired positions the virtual camera must provide the correct projection.
- Finding this projection is called camera calibration.

- Once the projection is found the viewing component of the Augmented Reality system uses it to represent the virtual world.

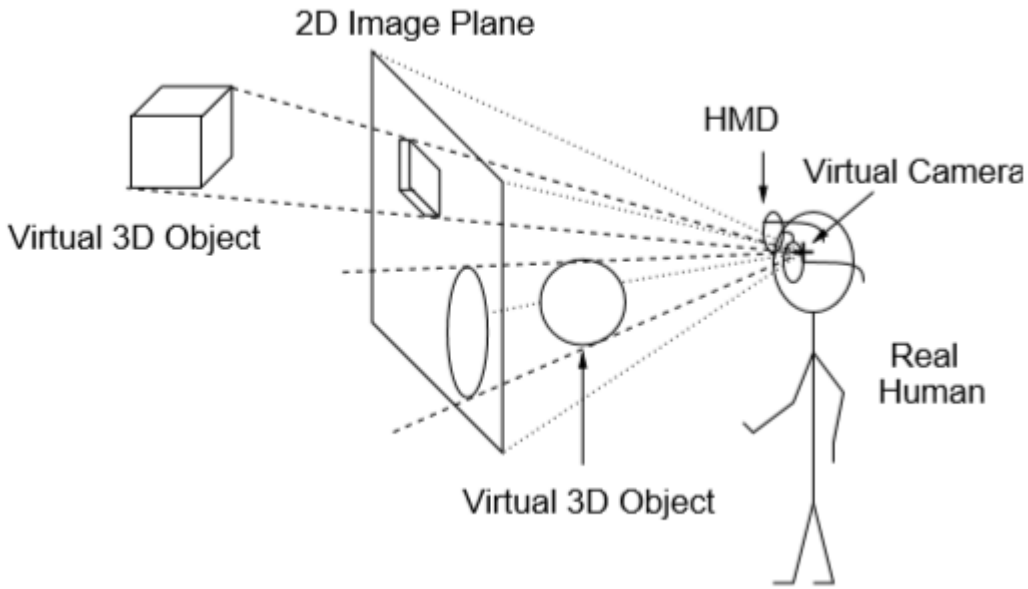


Fig. 4.3 The 3D virtual world is mapped to the 2D image plane

Virtual Camera / Camera Model

A camera maps the 3D virtual world to a 2D image. This mapping can be represented by a 3×4 projection matrix P. Each point gets mapped from the homogeneous coordinates of the 3D virtual world model to homogeneous coordinates of its image point on the image plane. In general this matrix has 11 degrees of freedom (3 degrees of freedom for the rotation, 3 more for the translation, and 5 from the calibration matrix K) and can be split up into two matrices P = KT. Whereas the matrix K holds the internal camera parameters, such as the focal length and aspect ratio. T is a simple transformations matrix which holds the external camera parameters that are the rotation and the translation. A finite camera is a camera whose center is not at infinity. Let e.g. the center be the origin of an Euclidean coordinate system, and the projection plane z = f. It is also called the image plane or focal plane. The distance f is called the focal length. The line from the optical camera center vertical to the image plane is called the principal axis or principal ray of the camera. The point where this line intersects the image plane is called the principal point or the image center. Furthermore the plane through the camera center parallel to the image plane is called the principal plane. Assume that the Augmented Reality system already knows the position Tmarker and the orientation Rmarker of the tracked HMD marker represented as the transformation F in figure 4.5.

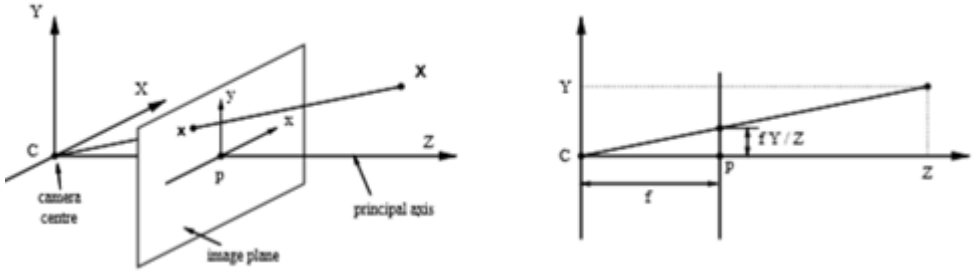
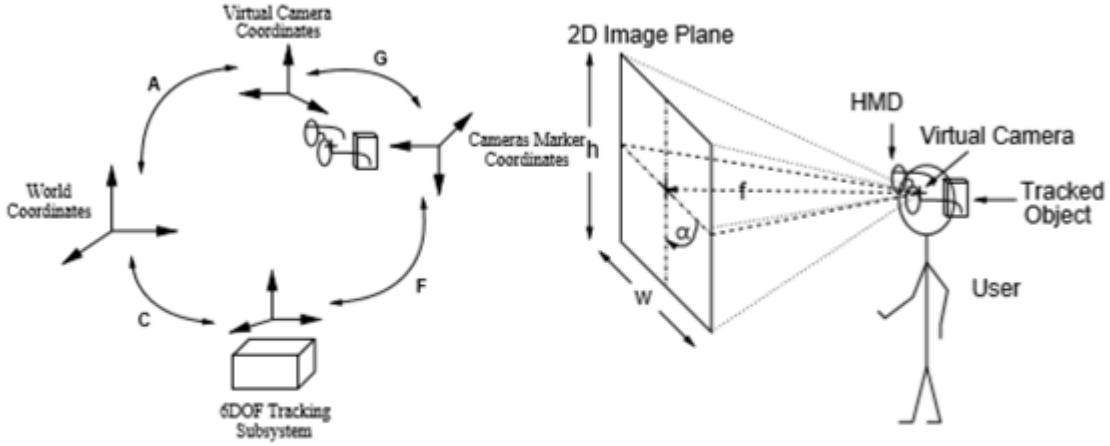


Fig. 4.4 Camera Geometry



(a) Extrinsic parameters and the relevant coordinate system transformations

(b) intrinsic parameters such as focal length f, aspect ratio w/h, and the skew represented as the angle α

Fig. 4.5 The Camera Calibration Parameters

In order to get the position and orientation of the virtual camera center the additional transformation G should be found. The position Tcamera2marker can be found by measuring with a ruler the distances in all three directions X,Y, and Z of the cameras bodies coordinate system to the virtual cameras center which should be approximately between the user's eyes. And the orientation Rcamera2marker can be found by measuring the angles χ, ρ, and σ between the X-, Y-, and Z-axis of the camera marker and the corresponding axis of the virtual camera coordinate systems. Constituting these angles, the desired transformation is

$$G = Rcamera2marker\ [I|-Tcamera2marker]$$

Now the viewing subsystem of the Augmented Reality system knows the pose of the virtual camera relative to the tracking subsystem's coordinate system. As the transformation C is also constant, because the tracking subsystem is rigidly fixed in the laboratory, it can be measured the same way as well. Thus the overall transformation matrix A that maps the virtual camera center to the world coordinate system is

$$A = GFC$$

where A is a 3 × 4 projection matrix that transforms world coordinates to camera coordinates. C is a 4×4 homogeneous transformation matrix that maps world to tracker coordinates. During the implementation phase it is assumed that the world and the tracker coordinate systems are equal, e.g. C = I. Furthermore F is also a 4 × 4 homogeneous transformation matrix that maps the coordinate system of the camera marker to the tracker coordinate system. At last G is the 3×4 projection matrix that defines the camera transformation relative to the coordinates of the camera marker. The matrix G is the desired projection matrix, as F is known to the Augmented Reality system by the tracking subsystem.

Calibration needed for -

In order to get an effective augmentation of the real world, the real and virtual objects must be accurately positioned relative to each other. The computer system contains in its virtual world a virtual camera which can see a range of several virtual objects. These are usually displayed on a head mounted display (HMD) to get an AR view. So the HMD

represents a virtual camera for which several parameters must be accurately specified as well. If these parameters do not fit properly, the virtual picture might have a different size than the real one or even be distorted. Once all the parameters are found and adjusted correctly, the user can use the AR system to augment the reality. But may be some other user wants to use the same AR system as well. And maybe he has another interocular distance or he wears the HMDs lightly different than the person who first adjusted all the parameters. Even if this person puts on the HMD for another session again, the primarily adjusted parameters will not fit as good any more. So the procedure of calibrating the virtual camera has to be kept simple, in order to make it possible for users who know nothing about the mathematical background of calibration to adjust the HMD anytime fast and precise. Another problem has always been the accurate adjustment of different displays because different algorithms are necessary.

Calibration is the process of instantiating parameter values for mathematical models which map the physical environment to internal representations, so that the computer's virtual world matches the real world. These parameters include information about optical characteristics and pose of the real world camera, as well as informations about the environment, such as the tracking systems origin and pose of tracked objects.

Functional Requirements of the calibration service- describe the interactions between the system and its environment.

1. Accurate Alignment

The main goal of a successful calibration is an Augmented Reality system in which all virtual objects are optimal adjusted. The HMD user should see the real objects and the corresponding superimposed virtual objects accurate aligned. So the distance, size, and form of them should fit to their real counterparts. Even when the user moves through the tracked space the visual enhancement shall keep correctly aligned.

2. Easy to Use

In order to get a practical solution for the calibration of see through devices, the parameters need to get estimated in a user-friendly procedure. Thus the user interaction where the calibration points are measured shall be intuitive and not impose a great burden on the user.

Nonfunctional Requirements – describe the user visible aspects of the system that are not directly related with the functional behavior of the system

1. Performance

The performance of the calibration procedure primarily depends on the performance of the middleware. As the calibration service depends on DWARF the desired components can be executed distributed on different computers. Thus the system latency (delay) should be kept at a minimum. It is vitally to receive the relevant measurement data in real time as the user is allowed to move. Thus the measured parameters change in real time, too. The actual calculation of the calibration parameters does not need to be in real time as it will be done just once. But the updating of the viewing component should be completed within a few seconds.

2. Accurate Tracking

For an accurate alignment the Augmented Reality system needs to know the exact pose of the virtual camera respectively the tracked 6DOF (freedom of movement of a rigid body in three-dimensional space) marker of the HMD in real time. The pose of other objects, such as the position of the 3DOF calibration points, must be known, too. As the real world location of these objects may change by moving these, the virtual objects need the same pose change. This is solved by the ART track1 tracking subsystem which updates the virtual model of the Augmented Reality system in real time.

3. Reliability

The system should guide the user in a way that it is guaranteed to obtain good results. Additionally it should provide hints on how good the accuracy is at the moment.

4. Quality of Service

The goal is to find the optimal solution where the measurement deviation is minimized. Furthermore error estimates should be provided to other DWARF components in order to be able to reduce error accumulation.

Pseudo Requirements

Pseudo requirements are imposed by the client that restricts the implementation of the system. The only pseudo requirement that occurred for the calibration method is that the prototypical implementation has to be done in context with ARCHIE. So the main focus has been a good aligned ARCHIE application rather than a perfect calibration method. Consequently the user interface controller of the calibration method needed to be written in Java depending on the object-oriented Petri net simulation framework called JFern .

NOTE: DWARF stands for Distributed Wearable Augmented Reality Framework. The name is an acronym representing the guidelines for the general system architecture. ARCHIE is the latest application. The acronym stands for Augmented Reality Collaborative Home Improvement Environment. The completion of the ARCHIE project provides new functionality to DWARF thereby making it more mature. Advanced Realtime Tracking (ART). For accurate position and orientation tracking, the infrared (IR)-optical Advanced Realtime Tracking subsystem ART track 1 is used with four cameras. Single Point Active Alignment Method (SPAAM) is a simple method to calibrate virtual cameras

V. Marker-based Augmented Reality

Marker-based augmented reality experiences require a static image also referred to as a trigger photo that a person can scan using their mobile device via an augmented reality app. The mobile scan will trigger the additional content (video, animation, 3D or other) prepared in advance to appear on top of the marker. Marker-based Augmented Reality uses a designated marker to activate the experience.

Popular markers include Augmented Reality QR codes, logos, or product packaging. The shapes or images must be distinctive and recognizable for the camera to properly identify it in various surroundings. There is another important factor of marker-based Augmented Reality. The marker-based AR experience is tied to the marker. This means that the placement of digital elements depends on the location of the marker. In most cases, the experience will display on top of the marker and move along with the marker as it is turned or rotated.

The key feature of Augment Reality in comparison to other image processing tools is that virtual objects are moved and rotated in 3D coordinates instead of 2D image coordinates. The main objectives of AR are analysis of changes in the captured camera frames and correct alignment of the virtual data into the camera scene based on the tracking results. In turn, a marker-based approach provides the accurate tracking using visual markers, for instance, binary markers (designed by ARUCO, METAIO, etc.) or with photo of real planar objects in camera scene.

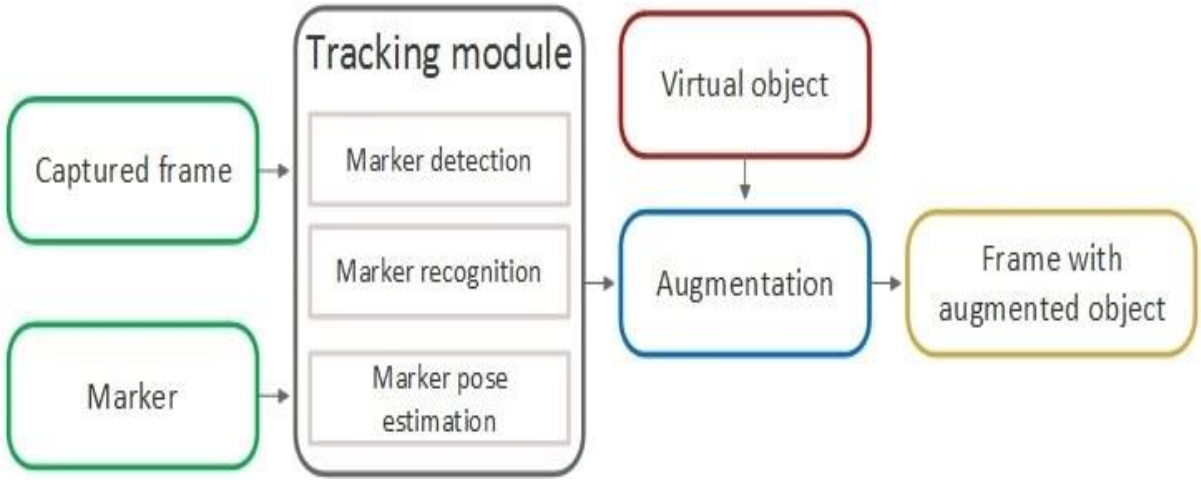


Fig. 4.6 AR System Flowchart

At first, the marker image should be there and then extract the consecutive camera frames. The tracking module in flowchart (Fig. in 4.6) is the core of the augmented reality system. It calculates the relative pose of the camera based on correctly detected and recognized marker in the scene. The term “pose” means the six degrees of freedom (DOF) position, i.e. the 3D location and 3D orientation of an object. The tracking module enables the system to add virtual components as a part of the real scene. And since the dealing is with camera frames in 2D coordinates system, it is necessary to use the projective geometry for virtual 3D object augmentation.

Detection and recognition

In the case of tracking by binary marker, the first necessary thing is to print the desired marker and place it in front of the camera. This requirement is an evident drawback of the tracking algorithm. The algorithm of detection is very simple and based on the marker nature:

- Application of adaptive thresholding to extract edges;
- Extraction of closed contours from binary image;
- Filtration of contours;
- Contours approximation and detection of quadrilateral shaped contours.

After above steps the marker candidates are stored for the further marker recognition. Each candidate is warped to the frontal view and divided on blocks. The task of recognition algorithm is to extract binary code from the marker candidate and compare it with the code of true marker. The most similar candidate is considered as a matched marker.

How to create marker-based augmented reality content?

Need - to create own augmented reality experience:

- a static trigger image
- digital content such as video or 3D object to feature on top of the chosen picture

- a software for combining the two pieces of content, (such as the Overly self-service AR creator)
- a mobile device with a compatible application to scan the marker and retrieve the AR content.

Getting the AR content straight

- what makes an excellent augmented reality marker?

An AR marker must be a photo specially created for our use. Make sure that the image is unique. Choose own design, a photo you have taken, or something from the business' library that no one else could easily access online. Using stock photos or Google images is not a good idea as someone else could be already using the same picture or will do so at a later stage and because of that it can show different content on the same marker. In short, there will not be ownership of that content. For the computer vision to detect the marker, use as many graphical elements and contrasts in the trigger photo as possible. The marker image is not a thing to be a minimalist about. The image recognition system thrives on different shapes, shadows, etc. Finally, fix the marker – where it is going to end up. Even if it looks great on the computer, ensure that it works when printed. Points to be noted: choose matte finish as glossy photos due to their reflection with light may be hard to read. AR markers are challenging to use for rounded objects such as cans or bottle labels, consider creating a bottle tag instead to ensure AR content looks good. Bottles can work, but usually it takes knowledge to create such things.

Environment considerations for marker-based augmented reality experiences

Even if the perfect marker is picked by design, there are still some other pointers that are key to the success of creating augmented experience. AR marker placement will be detrimental as the same trigger photo may not work at a bus stop as it does on a building facade or a small flayer. 50% rule for computer vision. Wherever the augmented reality marker is placed, consider that when people scan it with their mobile device, it must take up at least 50% of the camera screen — no less. So incase if a scene is intended to create in outdoor, consider how high up the banner is going to be. The higher it is, the bigger it has to be. If you're placing a marker on an A4 poster, consider how close people are going to be able to get to it, because for A4 one meter probably is optimal. If big banner ad is considered, ensure that the space for the AR content to be retrieved isn't limited. Consider that people need to be able to get the trigger photo within their camera screen. So if it is on a larger size, you should make sure that the environment surrounding it is vast enough for people to step back and scan the marker. If you place a large-scale ad on a busy junction, it may not be the best idea to add AR to it, because there may not be an appropriate space for people to stand to scan it or step back to get the image within their screen.

Outdoor AR markers are weather dependent

While an outdoor ad that has been lit up will work for 24/7, its AR functionality will not deliver the scans in all weather conditions. One of the challenges is night time, and if the AR trigger photo is consumed by night, you won't be able to see it nor scan it. Another point to consider when placing a marker outdoors is sunlight or shade. Both of these can affect if computer vision is capable of detecting your poster. Therefore, avoid placing AR markers on banners where there are drastic sunlight and shadow changes throughout the day.

Combining the AR content with a marker in 5 steps

- The platform - will most probably be web-based and require registration but once it is done, the steps should be similar for all. Upload the static design (trigger image) that

you want to bring to life. Then - asked to upload the content (based on the platform, the type of content can be uploaded). Choose the content size and its location in respect to the marker, add any CTAs if possible

- Preview the look on web and press publish
- In a few seconds, the system will be updated, and
- can take the mobile phone to test the marker and
- share it.

Marker recognition can be local or cloud-based, it means that marker databases can be stored on device and recognition also happens on device. The databases can also be stored on a cloud and recognition happens on a server, phone is only sending point clouds to server. Device-based recognition can happen immediately, but if cloud recognition is used, then it will take a while longer for the content to be downloaded from the server. Usually it takes a couple of seconds before the user can see any augmented reality experience.

Pros

- If the marker image is prepared correctly, marker-based AR content provides quality experiences and tracking is very stable, the AR content doesn't shake
- Easy to use, detailed instructions are not required for people who use it for the first time

Cons

- When the mobile camera is moved away from the marker, AR experience disappears and the trigger photo has to be scanned again. It is possible to use extended tracking, but in most cases, extended tracking makes things worse.
- Scanning will not work if markers reflect light in certain situations (can be challenging with large format banners in ever-changing weather conditions)
- Marker has to have strong borders/contrast between black and white colors to make tracking more stable.
- Smooth color transition will make recognition impossible.

VI. Pattern Recognition

At the age of 5, most children can recognize digits and letters – small characters, large characters, handwritten, machine printed, or rotated – all easily recognized by the young. In most instances, the best pattern recognizers are humans, yet unaware to understand how humans recognize patterns. Pattern recognition is the automated recognition of patterns and regularities in data. Techniques for finding patterns in data have undergone substantial development over the past decades. Pattern recognition analyzes incoming data and tries to identify patterns. While explorative pattern recognition aims to identify data patterns in general, descriptive pattern recognition starts categorizing the detected patterns. Hence, pattern recognition deals with both of these scenarios, and different pattern recognition methods are applied depending on the use case and form of data. Consequently, pattern recognition is not *one* technique but rather a broad collection of often loosely related knowledge and techniques. Pattern recognition capability is often a prerequisite for intelligent systems. The data inputs for pattern recognition can be words or texts, images, or audio files. Hence, pattern recognition is broader compared to computer vision that focuses on image recognition. Automatic and machine-based recognition, description, classification, and grouping of patterns are important problems in a variety of engineering and scientific

disciplines, including biology, psychology, medicine, marketing, computer vision, and artificial intelligence.

Pattern?

In 1985, Satoshi Watanabe defined a pattern “as the opposite of a chaos; it is an entity, vaguely defined, that could be given a name”. In other words, a pattern can be any entity of interest that one needs to recognize and identify: It is important enough that one would like to know its name (its identity). Therefore, patterns include repeated trends in various forms of data. For example, a pattern could be a fingerprint image, a handwritten cursive word, a human face, or a speech signal. A pattern can either be observed physically, for example, in images and videos, or it can be observed mathematically by applying statistical algorithms.

Fig. 4.7 Examples of patterns: Sound wave, tree species, fingerprint, face, barcode, QR-code, handwriting, or character image

Recognizing a pattern?

Given a pattern, its recognition and classification can consist of one of the following two tasks: Supervised classification identifies the input pattern as a member of a predefined class (Descriptive). Unsupervised classification assigns the input pattern to a undefined class (Explorative). The recognition problem is usually posed as either classification or categorization task. The classes are either defined by the system designed (supervised classification) or are learned based on the similarity of patterns (in unsupervised

classification). Pattern recognition is constantly evolving, driven by emerging applications that are not only challenging but also more computationally intensive.

Goal of pattern recognition

The goal of pattern recognition is based on the idea that the decision-making process of a human being is somewhat related to the recognition of patterns. For example, the next move in a chess game is based on the board's current pattern and buying or selling stocks is decided by a complex pattern of financial information. Therefore, the goal of pattern recognition is to clarify these complicated mechanisms of decision-making processes and to automate these functions using computers.

Definition of pattern recognition

Pattern recognition is defined as the study of how machines can observe the environment, learn to distinguish various patterns of interest from their background, and make logical decisions about the categories of the patterns. During recognition, the given objects are assigned to a specific category. Pattern Recognition, as it is a constantly evolving and broad field. An early definition of pattern recognition defines it as "a classification of input data via extraction of important features from a lot of noisy data" (1978, Thomas Gonzalez). In general, pattern recognition can be described as an information reduction, information mapping, or information labeling process. In computer science, pattern recognition refers to the process of matching information already stored in a database with incoming data based on their attributes.

Pattern Recognition and Artificial Intelligence (AI)

Artificial Intelligence (AI) refers to the simulation of human intelligence, where machines are programmed to think like humans and mimic their actions. Most prominently, fields of artificial intelligence aim to enable machines to solve complex human recognition tasks, such as recognizing faces or objects. Accordingly, pattern recognition is a branch of Artificial Intelligence.

Pattern Recognition and Machine Learning

Today, in the era of Artificial Intelligence, pattern recognition and machine learning are commonly used to create ML models that can quickly and accurately recognize and find unique patterns in data. Pattern recognition is useful for a multitude of applications, specifically in statistical data analysis and image analysis. Most modern use cases of pattern recognition are based on artificial intelligence technology. Popular applications include speech recognition, text pattern recognition, facial recognition, movement recognition, recognition for video deep learning analysis, and medical image recognition in healthcare.

How does Pattern Recognition Work?

Historically, the two major approaches to pattern recognition are Statistical Pattern Recognition (or decision-theoretic) and Syntactic Pattern Recognition (or structural). The third major approach is based on the technology of artificial neural networks (ANN), named Neural Pattern Recognition. No single technology is always the optimal solution for a given pattern recognition problem. All three or hybrid methods are often considered to solve a given pattern recognition problem.

Statistical Pattern Recognition

Statistical Pattern Recognition is also referred to as StatPR. In statistical pattern recognition, the pattern is grouped according to its features, and the number of features

determines how the pattern is viewed as a point in a d-dimensional space. These features are chosen in a way that different patterns take space without overlapping. The method works so that the chosen attributes help the creation of clusters. The machine learns and adapts as expected, then uses the patterns for further processing and training. The goal of StatPR is to choose the features that allow pattern vectors to belong to different categories in a d-dimensional feature space.

Syntactic Pattern Recognition

Syntactic Pattern Recognition, also known as SyntPR, is used for recognition problems involving complex patterns that can be addressed by adopting a hierarchical perspective. Accordingly, the syntactic pattern approach relies on primitive subpatterns (such as letters of the alphabet). The pattern is described depending on the way the primitives interact with each other. An example of this interaction is how they are assembled in words and sentences. The given training samples develop how grammatical rules are developed and how the sentences will later be "read". In addition to classification, structural pattern recognition also provides a description of how the given pattern is constructed from the primitive subpatterns. Hence, the approach has been used in examples where the patterns have a distinct structure that can be captured in terms of a rule-set, such as EKG waveforms or textured images. The syntactic approach may lead to a combinatorial explosion of probabilities to be examined, requiring large training sets and very large computational efforts.

Template-matching

Template matching is one of the simplest and earliest approaches to pattern recognition. Matching is a generic operation that is used to determine the similarity between two entities of the same type. Therefore, template matching models try to discover similarities in a sample based on a reference template. Hence, the template matching technique is commonly used in digital image processing for detecting small sections of an image that match a template image. Typical real-world examples are medical image processing, quality control in manufacturing, robot navigation, or face recognition.

Neural network pattern recognition

AI pattern recognition using neural networks is currently the most popular method for pattern detection. Neural networks are based on parallel subunits referred to as neurons that simulate human decision-making. It can be viewed as massively parallel computing systems consisting of a huge number of simple processors with many interconnections (Neurons). The most popular and successful form of machine learning using neural networks is deep learning, which applies deep convolutional neural networks (CNN) to solve classification tasks. Today, neural network pattern recognition has the edge over other methods because it can change the weights repeatedly on iteration patterns. In recent years, deep learning has proven to be the most successful method to solve recognition tasks.

Hybrid pattern detection

After going through all the pattern recognition techniques, it is evident that no algorithm is always the most efficient for any use case. Therefore, combinations of various machine learning and pattern recognition algorithms lead to the best results or enable the implementation of efficient and optimized pattern detectors. Consequently, many pattern recognition projects are based on hybrid models to enhance the performance of the pattern recognizer for the specific use cases, depending on the type and availability of data. For example, deep learning methods achieve outstanding results but are computationally intensive, while "lighter" mathematical methods usually are more efficient. Also, it is

common to apply methods for data pre-processing before applying AI pattern recognition models. Using the hybrid model will enhance the performance of the entire application or detection system.

Process of finding patterns in data

The design of pattern recognition systems essentially involves

- data acquisition and preprocessing,
- data representation, and
- decision making.

The pattern recognition process itself can be structured as follows:

- Collection of digital data
- Cleaning the data from noise
- Examining information for important features or familiar elements
- Grouping of the elements into segments
- Analysis of data sets for insights
- Implementation of the extracted insights

Pattern Recognition examples

Stock market prediction

- Using pattern recognition for stock market prediction applications is a classical yet challenging task with the purpose of estimating the future value of a company stock or other traded assets. Both linear and machine learning methods have been studied for decades. Only lately, deep learning models have been introduced and are rapidly gaining in popularity.

Optical character recognition

- Optical character recognition (OCR) is the process of classification of optical patterns contained in a digital image. The character recognition is achieved through image segmentation, feature extraction, and classification.

Text pattern recognition

- Machine learning based pattern recognition is used to generate, analyze, and translate text. Hence, patterns are used to understand human language and generate text messages. Accordingly, text recognition on words is used to classify documents and detect sensitive text passages automatically. Therefore, text pattern recognition is used in the Finance and Insurance industries for fraud detection.

Handwriting recognition

- Handwriting recognition is used to compare patterns across handwritten text or signatures to identify patterns. Various applications are involved in the computer recognition of pen-input handwritten words. However, handwritten word recognition and spotting is a challenging field because handwritten text involves irregular and complex shapes.

Face recognition and visual search

- Image recognition algorithms aim to detect patterns in visual imagery to recognize specific objects (Object Detection). A typical image recognition task is image classification, which uses neural networks to label an image or image segment based on what is depicted. This is the basis of visual search, where users can easily search and compare labeled images.

Voice or speaker recognition

- Voice recognition systems enable machines to receive and interpret dictation or are able to carry out spoken commands and interact accordingly. Speech recognition is based on machine learning for pattern recognition that enables recognition and translation of spoken language.

Emotion recognition systems

- Machine learning in pattern recognition is applied to images or video footage to analyze and detect the human emotions of an audience. The goal is to indicate the mood, opinion, and intent of an audience or customers. Hence, deep learning is applied to detect specific patterns of facial expressions and movements of people. Those insights are used to improve marketing campaigns and customer experience.

Benefits of Pattern Recognition

- Pattern recognition methods provide various benefits, depending on the application. In general, finding patterns in data helps to analyze and predict future trends or develop early warning systems based on specific pattern indicators. Further advantages include:
- Identification: Detected patterns help to identify objects at different angles and distances (for example, in video-based deep learning) or identify hazardous events. Pattern recognition is used to identify people with video deep learning, using face detection or movement analysis. Recently, new AI systems can identify people from their walk by measuring their gait or walking pattern.
- Discovery: Pattern recognition algorithms allow "thinking out of the box" and detecting instances that humans would not see or notice. Algorithm patterns can detect very fine movements in data or correlations between factors across a huge amount of data. This is very important for medical use cases; for example, deep learning models are used to diagnose brain tumors by taking images of magnetic resonance imaging.
- In information security and IT, a popular pattern recognition example is the use of pattern matching with an intrusion detection system (IDS) to monitor computer networks or systems for malicious activity or policy violations.
- Prediction: Forecasting data and making predictions about future developments play an important role in many pattern recognition projects, for example, in trading markets to predict stock prices and other investment opportunities or to detect trends for marketing purposes.
- Decision-making: Modern machine learning methods provide high-quality information based on patterns detected in near real-time. This enables decision-making processes based on reliable, data-based insights. A critical factor is the speed of modern, AI pattern recognition systems that outperform conventional methods and enable new applications. For example, medical pattern recognition, to detect risk parameters in data, providing doctors critical information rapidly.
- Big-Data analytics: With neural networks, it became possible to detect patterns in immense amounts of data. This enabled use cases that would not have been possible with traditional statistical methods. Pattern recognition is vital in the medical field, especially for forensic analysis and DNA sequencing. For example, it has been used to develop vaccines to battle the COVID-19 Coronavirus.

Pattern recognition algorithms can be applied to different types of digital data, including images, texts, or videos. Finding patterns enables the classification of results to

enable informed decision-making. Pattern recognition can be used to fully automate and solve complicated analytical problems.

Interactive E-Learning System Using Pattern Recognition and Augmented Reality

The goal - is to provide students with realistic audio-visual contents when they are leaning. The e-learning system consists of image recognition, color and polka-dot pattern recognition, and augmented reality engine with audio-visual contents. When the web camera on a PC captures the current page of textbook, the e-learning system first identifies the images on the page, and augments some audio-visual contents on the monitor. For interactive learning, the e-learning system exploits the color-band or polka-dot markers which are stuck to the end of a finger. The color-band and polka-dot marker act like the mouse cursor to indicate the position in the textbook image. Appropriate interactive audio-visual contents are augmented as the marker is located on the predefined image objects in the textbook. This was applied to the educational courses in the school and obtained satisfactory results for real applications.

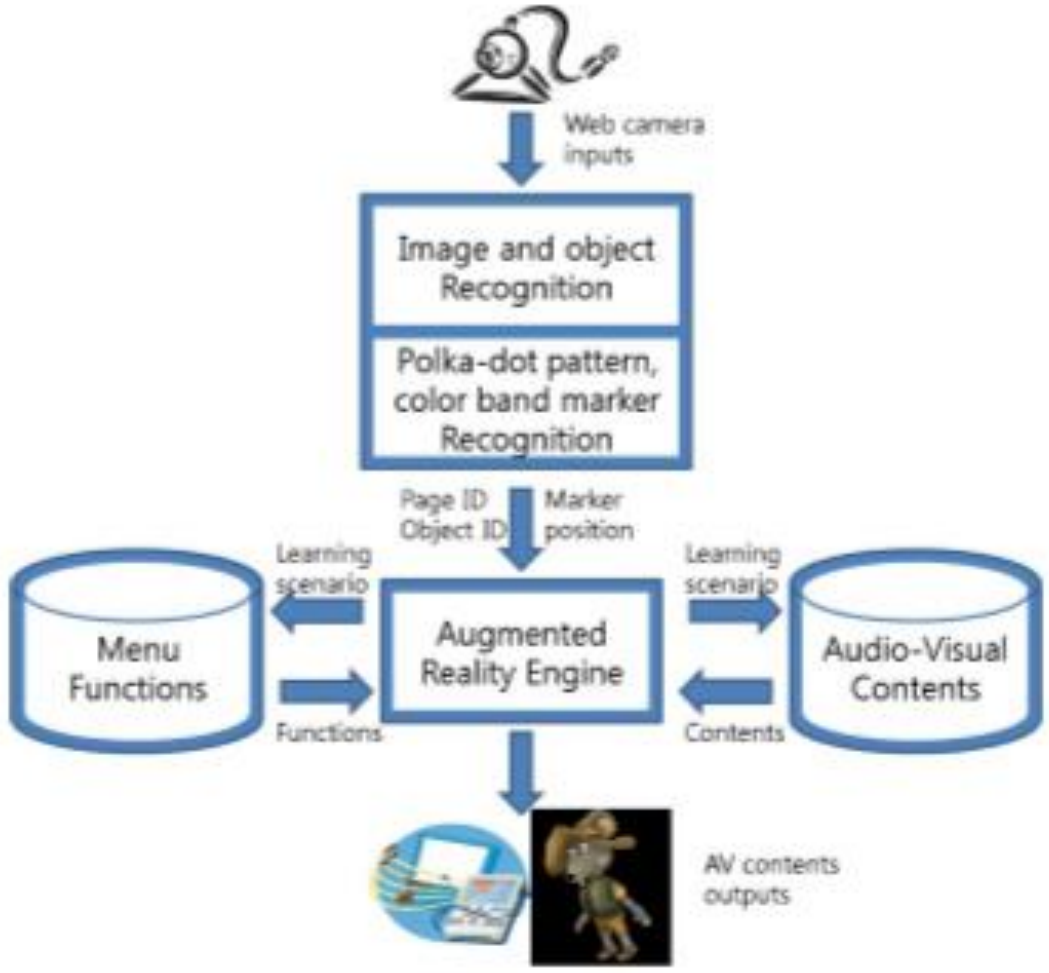


Fig. 4.8 Structure of e-learning system

The image and marker recognition enables students to learn interactively according to the predefined learning scenarios and audio-visual contents.

The e-learning system consists of image/object recognition, polka-dot pattern recognition, color-band marker recognition, augmented reality engine, audio-visual contents, and some learning scenarios of textbooks. The learning scenarios are the predefined processes when or

where to augment the contents. The scenarios combine the educational contents with information technologies to maximize the learning efficiency. And the augmented reality engine realizes the scenarios. Fig. 4.8 shows the structure of e-learning system. A web camera connected to the computer focuses on the textbook. The students study watching the textbook and the captured video frame where some audiovisual contents are augmented. When video frames from web camera are given, the recognition modules identify the image and objects on the textbook, and polka-dot or color-band marker. Database of images and objects in the textbook in advance - is available. The image/object recognition module identifies the current text page and objects that the student is studying. Using the identified pages and objects from recognition module, the system knows where the objects are located in the video frame. Then, some audio-visual contents are augmented on the computer monitor according to the predefined educational scenarios. The augmented reality engine matches the scenarios to information from the recognition modules, and plays the audio-visual contents automatically. Some interactive learning actions are possible by the polkadot or color-band marker. The marker is a kind of computer mouse, and indicates the location in the video frame. If the marker is located on the specific objects or menu bars, the object-based interactions are performed based on the educational scenarios and contents. The related visual contents are displayed on the marker even though the marker is moving. Some interactive actions, such as dragging the virtual object, scrubbing-based reaction, and menu selection, are also defined in the e-learning system. For the usefulness of e-learning system, many educational contents and scenarios are produced for the real school courses. In addition, the authoring tool is developed to produce the educational scenarios and interactions easily, since the system is designed for general purposes. Thus, any contents providers and educational organizations can exploit the e-learning system for their interactive learning courses.

Two markers are designed using polka-dot pattern or color-band. The markers are put on the fingers as bands, and act like the computer mouse. The markers indicate their locations in the video frame, which enables the students to interact according to the objects in the textbook. When the marker is located at a specific object or menu in the textbook, the corresponding audio-visual contents are augmented on the computer, or the predefined menu function is performed. And some interactive functions such as dragging and scrubbing object are defined to support various learning actions.

Polka-dot Pattern Recognition

The polka-dot patterns are rare in the usual textbook, and well recognized both in the grayscale and color images. The polka-dot band for a finger is used as a computer mouse. The polka-dot marker is exploited for interactive augmentation of contents and menu selection. To detect polka-dot pattern exactly in real-time, fast filters of integer operations, hierarchical searching, and edge information are used. Fig. 4.9 shows two array patterns of polka-dot markers. The array patterns are empirically selected by the polka-dot recognition algorithm. Since the marker on a finger is subject to be rotated and slanted at the camera viewpoint, the array pattern of dots should be invariant to the perspective variations. According to the recognition algorithm, the optimal array pattern was selected. The hexagonal array is the best pattern that is invariant to the perspective distortions of camera viewpoints.

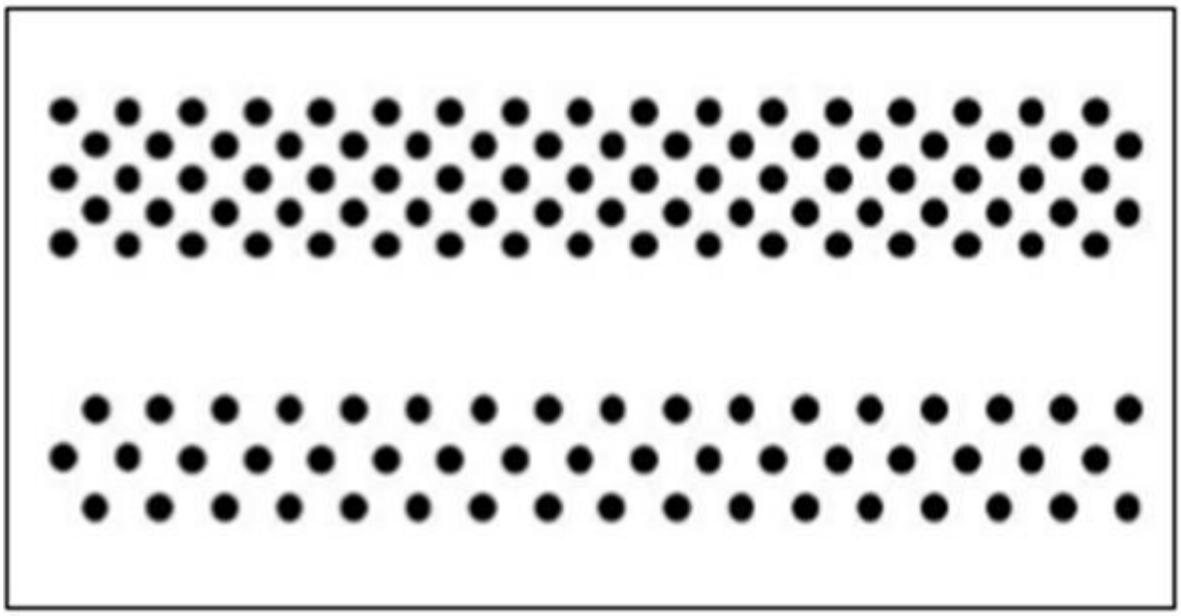


Fig. 4.9 Polka-dot patterns for interactive learning

The basic algorithm of polka-dot pattern recognition is the high pass filter in the horizontal and vertical directions. The high pass filter first finds the area where the grayscale pixel values are regularly varied with black and white pattern. For fast operation in marker detection, the search range is restricted based on the motion vector of previously detected polka-dot marker. The motion vector of polka-dot marker enables us to predict the next location. The next position of marker can be predicted; first detect the marker in the restricted search range. If the polka-dot marker is not detected in the restricted search range, the search range is expanded and the marker is found again. Finally, the detected marker is examined by edge information. Since the high pass filters can detect the complex textures or characters in the textbook as polka-dot markers, edge information is exploited to reduce the false positive errors. Since the characters or other complex textures usually have some line-edge properties unlike the polka-dot patterns, the false positive errors are decreased by the edge information. Fig. 4.10 shows some results of polka-dot pattern recognition. It is shown that one or two independent polka-dot markers are detected in the video frame. In the usual personal computer environment, the recognition is performed at higher than 25 frames per second for 640x480 resolution.

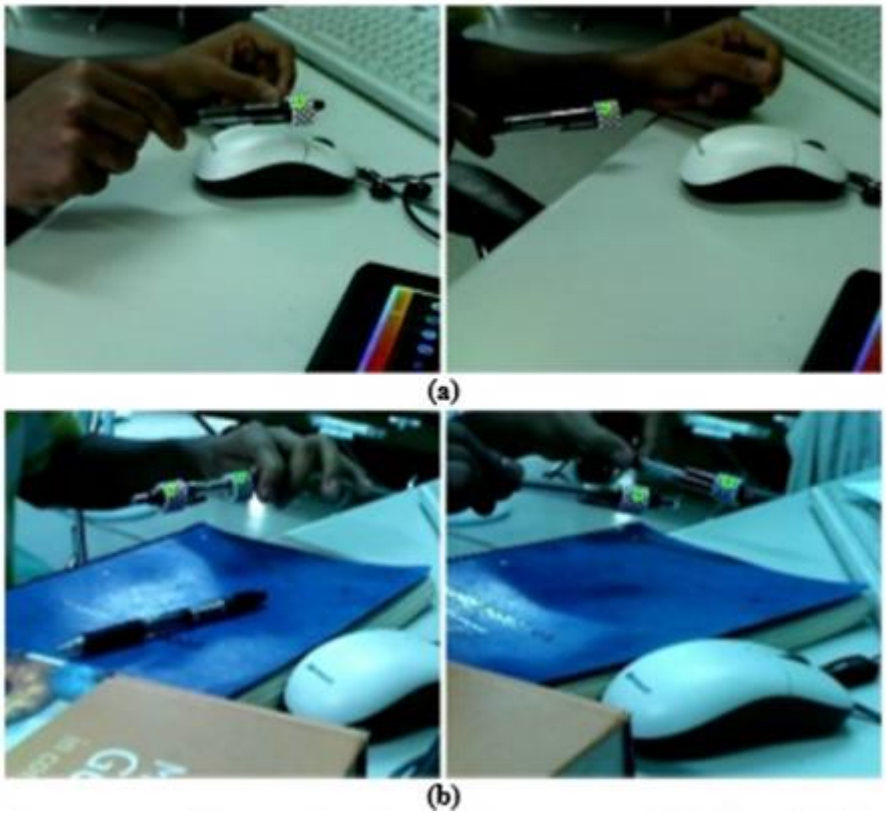


Fig. 4.10 Recognition Results of Polka-dot patterns. Multiple polka-dot markers are independently detected in the video frame. The light green squares mean the central position of polka-dot markers. (a) Single marker detection, (b) two markers detection

Color-band Recognition

Some interactions in the educational scenarios require two or more markers simultaneously to manipulate multiple objects. Since the polka-dot patterns have little distinct difference between them, it is difficult to operate the multiple markers independently. New multiple markers are needed to discriminate individually. Two color-band markers are designed which consist of three colors as shown in Fig. 4.11. The color-band markers are discriminated with each other and the polka-dot marker, thus, three markers are used simultaneously according to the educational scenarios and interaction.

Fig. 4.11 Color-band markers using three colors

The colors of the markers are selected from various experiments. The blue color is usually best recognized and most stable in the lighting variation. The blue band is located

at the center of color-band marker, and is searched first. The other colors have been chosen since they are well discriminated with each other and the blue color. Two color-band markers are designed with different combinations of colors as shown in Fig. 4.11. The color-band markers are detected by finding blue color first. The hue components in HSV (Hue Saturation Value) color space are used for robust detection in various lighting conditions. When blue color pixels are detected, the shape and area of blue region are examined - whether the blue region satisfies the condition of marker or not. Then, the other colors (Green and Red, or Yellow and Purple) are searched around the blue region. The color range and the area of the color region is considered to confirm the color-band pattern. The order of colors and ratios of color areas are compared with the predefined criterions. Fig. 4.12 shows that two color-band markers are independently detected in a video frame. The color ranges of color-band markers are optimized according to the lighting environment. Note that the color ranges should be changed with respect to the lighting conditions. Thus, method should be devised to adjust the color ranges of markers automatically when the e-learning system is setup.

Fig. 4.12 Recognition results of color-band markers. Two markers are consistently detected when they are moving

RECOGNITION OF IMAGE AND OBJECT

Image recognition is designed for identification of current text page or objects. When the text page or objects are identified, the related audio-visual contents are automatically played on the PC. Since the pose information of objects is obtained in the captured image, the visual 3-D contents are augmented according to the poses of objects. Augmented reality (AR) toolkits have used geometric markers to be recognized in the images. The AR markers consist of black/white geometric shapes in the square. The AR markers are well recognized in the various image distortions, and they have been popular for interactivity of virtual systems. However, since the AR markers are directly printed on the textbook pages, they do not look good for text design. Here, the goal is to replace the AR geometric markers with image objects and to design a natural interface using the image objects.

Feature Extraction

Since the images are subject to be rotated, distorted by perspective viewpoints, and changed by scales, robust features invariant to the image variations are extracted. Scale-

invariant features and some feature extraction algorithms are developed for image and objects recognition. The robust features called speeded up robust feature (SURF) can be exploited, which shows good recognition results and fast operation compared with SIFT. Since the e-learning system is also applied to the mobile devices like PDA (personal digital assistant) or mobile phone, SURF algorithm can be implemented with integer programming and optimized lookup tables. The first step of feature extraction is to detect the distinct points which are also invariant to image variations. The second step of feature extraction is to find a dominant orientation around the feature point. The orientation information normalizes the rotated images and objects. Thus, the images or objects are recognized in spite of rotational distortions. The last step of feature extraction is to describe the feature points as a vector structure. This descriptor recognizes the feature points. The square region around a feature point is selected for the descriptor. Note that the square is rotated by the dominant orientation before finding the descriptor. The size of square is related to the scale parameter. The square region is divided into 16 subregions, and 25 pixels (5x5) are sampled in each subregion

Feature Matching

The corresponding features are searched by the vector distance between descriptors. When features are extracted for image and object recognition, all pairs of features are examined by the vector distances. Then, the nearest (f1) and second nearest (f2) features are selected; and the nearest feature is matched with the features f. The only features that are on the same geometric relation are matched

Image and Object Recognition

With all pairs of matched features, the images or objects are recognized. The simplest method is to count the number of matched features. Without loss of generality, the image pairs that have the largest number of matched features are the same. There are some matching errors; homography is used to reduce matching errors. Since the homography reflects the geometric relations of features, it removes such mismatched features that satisfy the matching criterion without geometric correlation. Fig. 4.13 shows two example of image recognition. For real situations, the images or objects are occluded partially by the hand. As it is seen in Fig. 4.13, the images are well recognized under various image distortions, such as perspective distortion, luminance difference, scale difference, and occlusion. In Fig., the left images are the database images, and the right images are captured ones by the web camera. The images are well identified regardless of AR markers.

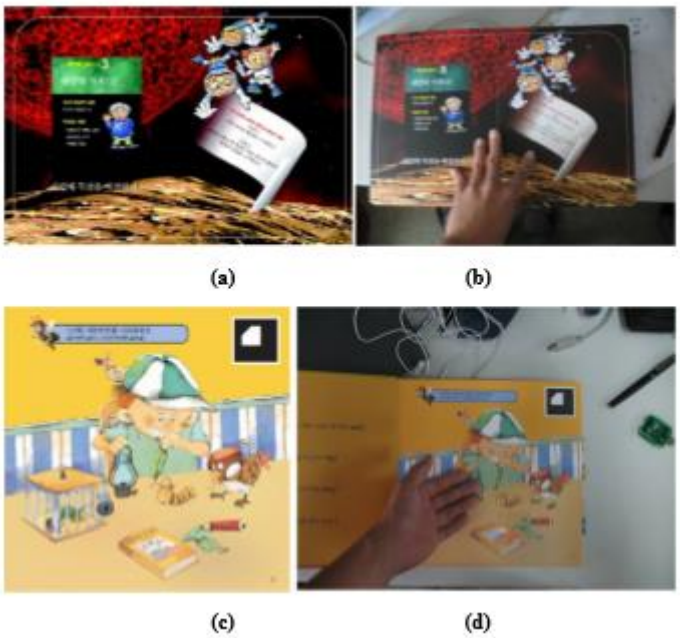


Fig. 4.13 Image recognition result (a) and (c) Original images recognized in the database regardless of AR markers, (b) and (d) Captured images by the webcam

Fig. 4.14 first shows that a moving graphic is augmented on the color-band marker. The left image (a) is the captured image by the web camera, and the right image (b) shows the augmented reality with graphic contents. The page ID is recognized by the image and objects in the text page. Then, the related audio-visual contents are augmented as the scenarios and student's interaction. The graphics are displayed above the marker so that the interactive augment reality is naturally performed. The augmented graphic objects also move as the marker moves.

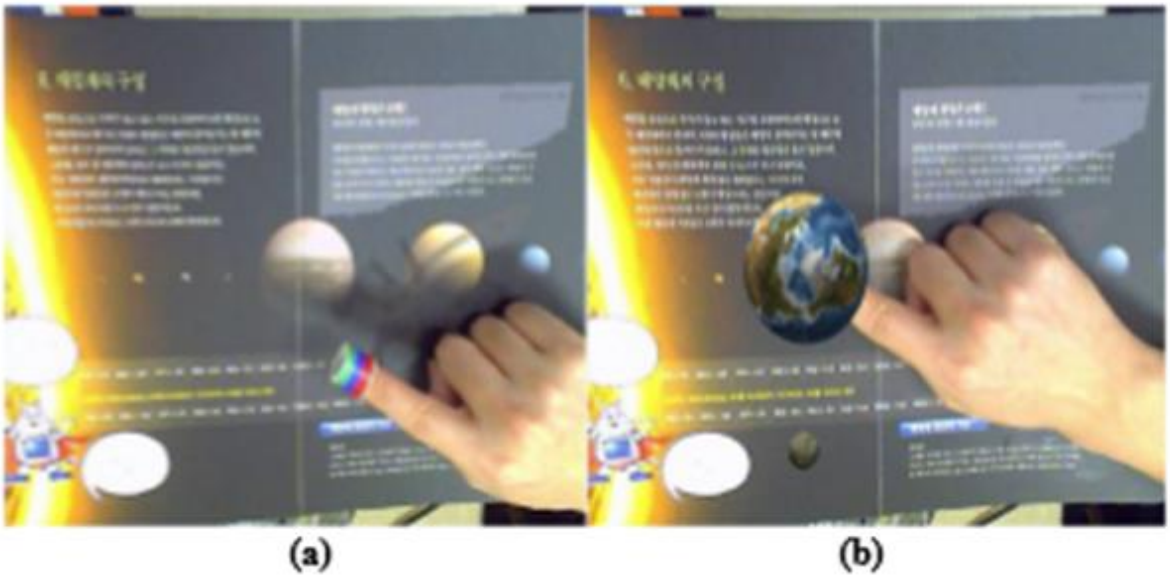


Fig. 4.14 Example of Augmented Reality using marker (a) a video frame captured by the web camera, (b) A visual content is augmented on the marker. The visual content is augmented on the marker, thus the marker is not seen on the monitor

Fig. 4.15 shows the commercial system and an exemplary image of interactive augmented reality. The interactive e-learning system using augmented reality was applied to the public

elementary school in the courses of English and Science. This interactive augment reality made the students have more interest in learning. Therefore, the e-leaning system not only provides with audio-visual contents, but also improves the learning efficiency and concentration of students. It is expected that the e-learning system is very useful in the various educational courses.

(a) (b)

Fig. 4.15 Example of interactive augmented reality using image and marker recognition

VII. AR Toolkit

ARToolKit is an open-source computer tracking library. It is used for creation of strong augmented reality applications that overlay virtual imagery on the real world. Currently, it is maintained as an open-source project hosted on GitHub. ARToolKit is a very widely used AR tracking library with over 160,000 downloads on its last public release in 2004. In order to create strong augmented reality, it uses video tracking capabilities that calculate the real camera position and orientation relative to square physical markers or natural feature markers in real time. Once the real camera position is known a virtual camera can be positioned at the same point and 3D computer graphics models drawn exactly overlaid on the real marker. So ARToolKit solves two of the key problems in Augmented Reality; viewpoint tracking and virtual object interaction.

ARToolKit is a C and C++ language software library that lets programmers easily develop Augmented Reality applications. Augmented Reality (AR) is the overlay of virtual computer graphics images on the real world, and has many potential applications in industrial and academic research. One of the most difficult parts of developing an Augmented Reality application is precisely calculating the user's viewpoint in real time so that the virtual images are exactly aligned with real world objects. ARToolKit uses computer vision techniques to calculate the real camera position and orientation relative to marked cards, allowing the programmer to overlay virtual objects onto these cards. The fast, precise tracking provided by ARToolKit should enable the rapid development of many new and interesting AR applications.

Features

- A multiplatform library (Windows, Linux, Mac OS X, SGI)
- A multi platform video library with:
    - multiple input sources (USB, Firewire, capture card) supported
    - multiple format (RGB/YUV420P, YUV) supported
    - multiple camera tracking supported
    - GUI initalizing interface
- A fast and cheap 6D marker tracking (real-time planar detection)
- An extensible markers patterns approach (number of markers)
- An easy calibration routine
- A simple graphic library (based on GLUT)
- A fast rendering based on OpenGL
- A 3D VRML support
- A simple and modular API (in C)
- Other language supported (JAVA, Matlab)
- A complete set of samples and utilities
- A good solution for tangible interaction metaphor
- OpenSource with GPL(General Public License) license for non-commercial usageTo develop an application - the source code for an existing example program: simpleLite.

The source code for this program is found inside ARToolKit installation in the directory examples/simpleLite/. The file is simpleLite.c. This program simply consists of a main routine and several graphics drawing routines. The functions which correspond to the six application steps described are shown in Table 1. The functions corresponding to steps 2 through 5 are called within the Idle() function.

Table 1: Function calls and code that corresponds to the ARToolKit applications steps.

ARToolKit Step	Functions
1. Initialize the video grabbing from the camera and load the marker(s)	setupCamera and setupMarker
2. Grab a video input frame	arVideoGetImage (called in mainLoop)
3. Detect the markers	arDetectMarker (called in mainLoop)
4. Calculate camera transformation	arGetTransMat (called in mainLoop)
5. Draw the virtual objects	Display
6. Close the application down	Quit

The most important functions in the program related to AR are main, setupCamera, setupMarker, mainLoop, Display, and cleanup. The GLUT library is used to handle the interaction with the operating system. GLUT, the OpenGL utility toolkit, is used to do things like open a window, and handle keypresses. However, GLUT is not required, and can be replaced with any library you like, e.g. MFC on Windows, Cocoa on Mac OS X, or QT (cross platform). The basics of a GLUT-based OpenGL application should be known before studying the code of simpleLite.c. OpenGL is a software interface to graphics hardware. This interface consists of about 150 distinct commands that you use to specify the objects and operations needed to produce interactive three-dimensional applications.

main

The main routine of simpleLite performs a number of setup tasks for the application. The first piece of AR specific code is near the top of main, where some variables that will be used to set up the application are declared:

```
char *cparam_name = "Data/camera_para.dat";
char *vconf = "";
char *patt_name = "Data/patt.hiro";
```

In this block - define the pathname of the camera parameter file the application will use, the video capture library configuration string, and the name of the marker pattern file the application will load and try to recognise. Next, first AR-specific function call:

```
// Hardware setup. //
if (!setupCamera(cparam_name, vconf, gARTThreshhold, &gARTCparam, &gARHandle, &gAR3DHandle))
{
fprintf(stderr, "main(): Unable to set up AR camera.\n");
exit(-1);
}
```

setupCamera loads a file containing calibration parameters for a camera, opens a connection to the camera, sets some defaults (the binarization threshold in this case) and starts grabbing frames. It records its settings into 3 variables which are passed in as parameters. In this case, these parameters are stored in global variables. The next piece of code opens up a window to draw into. This code uses GLUT to open the window.

```
// Library setup. //  // Set up GL context(s) for OpenGL to draw into.
glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGBA | GLUT_DEPTH);
if (!prefWindowed)
{ if (prefRefresh) sprintf(glutGamemode, "%ix%i:%i@%i", prefWidth, prefHeight, prefDepth, prefRefresh);
    else
        sprintf(glutGamemode, "%ix%i:%i", prefWidth, prefHeight, prefDepth);
        glutGameModeString(glutGamemode);
        glutEnterGameMode(); }
        else
        {
        glutInitWindowSize(prefWidth, prefHeight);
        glutCreateWindow(argv[0]);
}
```

The code uses the value of a variable "prefWindowed" to decide whether to open a window, or whether to use fullscreen mode. Other variables prefWidth, prefHeight, prefDepth and prefRefresh - to decide how many pixels wide and tall, what colour bit depth to use, and whether to change the refresh rate of the display are simply held in static variables defined near the top of main.c in simpleLite. Next, with a window from GLUT, initialise the OpenGL part of the application. In this case, the ARgsub_lite library is used to manage the interaction between the ARToolKit video capture and tracking, and OpenGL.

```
// Setup ARgsub_lite library for current OpenGL context.
if ((gArglSettings = arglSetupForCurrentContext(gARHandle)) == NULL)
{ fprintf(stderr, "main(): arglSetupForCurrentContext() returned error.\n");
Quit(); }
```

```
debugReportMode(gARHandle, gArglSettings);
glEnable(GL_DEPTH_TEST);
arUtilTimerReset();
```

The third major part of ARToolKit initialisation is to load one or more markers which the camera should track. Information about the markers has previously been recorded into marker pattern files using the mk_patt utility (called "marker training"), so now these files can be loaded. In simpleLite, one marker is used, the default Hiro marker. The task of loading this marker and telling ARToolKit to track it - is performed by the function called setupMarker().
setupCamera and setupMarker

Before entering a real-time tracking and drawing state, the ARToolKit application parameters should be initialised. The key parameters for an ARToolKit application are: the patterns that will be used for the pattern template matching and the virtual objects these patterns correspond to. the camera characteristics of the video camera being used.

setupCamera begins by opening a connection to the video camera from which images for tracking will be acquired, using arVideoOpen(). The parameter vconf, passed to arVideoOpen is a string which can be used to request some video configuration other than the default. The contents of the vconf string are dependent on the video library being used. At this point it is found out from the video camera library how big the images it will supply will be, and what pixel format will be used:

```
static int setupCamera(const char *cparam_name, char *vconf, int threshhold,
ARParam *cparam, ARHandle **arhandle, AR3DHandle **ar3dhandle)
{ ARParam wparam; int xsize, ysize;
int pixFormat;   // Open the video path.
 if (arVideoOpen(vconf) < 0)
{ fprintf(stderr, "setupCamera(): Unable to open connection to camera.\n");
return (FALSE); }   // Find the size of the window.
if (arVideoGetSize(&xsize, &ysize) < 0)
return (FALSE);
fprintf(stdout, "Camera image size (x,y) = (%d,%d)\n", xsize, ysize);   // Get the
format in which the camera is returning pixels.
pixFormat = arVideoGetPixelFormat();
if (pixFormat < 0 )
{ fprintf(stderr, "setupCamera(): Camera is using unsupported pixel format.\n");
return (FALSE); }
```

Next – is to deal with the structures that ARToolKit uses to hold its model of the camera's parameters. These parameters are generated by the camera calibration process. The camera parameter file is loaded with the call to arParamLoad, with the path to the file being passed in a c-string as a parameter. Once the camera parameters are loaded, we adjust them to match the actual video image size being supplied by the video library, and then initialise a few necessary ARToolKit structures which depend on the camera parameters:

```
// Load the camera parameters, resize for the window and init.
if (arParamLoad(cparam_name, 1, &wparam) < 0)
{ fprintf(stderr, "setupCamera(): Error loading parameter file %s for camera.\n",
cparam_name);
return (FALSE); }
```

```
arParamChangeSize(&wparam, xsize, ysize, cparam); fprintf(stdout, "*** Camera Parameter ***\n");
arParamDisp(cparam);
if ((*arhandle = arCreateHandle(cparam)) == NULL)
{ fprintf(stderr, "setupCamera(): Error: arCreateHandle.\n");
return (FALSE); }
if (arSetPixelFormat(*arhandle, pixFormat) < 0)
{ fprintf(stderr, "setupCamera(): Error: arSetPixelFormat.\n");
return (FALSE); }
```

setupCamera is completed by setting up some defaults related to the tracking portion of ARToolKit. These include debug mode, the labelling threshold, and the structure used to hold positions of detected patterns. Finally, the video library capturing frames are started and is ready to process -

```
if (arSetDebugMode(*arhandle, AR_DEBUG_DISABLE) < 0)
{ fprintf(stderr, "setupCamera(): Error: arSetDebugMode.\n");
return (FALSE); }
if (arSetLabelingThresh(*arhandle, threshhold) < 0)
{ fprintf(stderr, "setupCamera(): Error: arSetLabelingThresh.\n");
return (FALSE); }
if ((*ar3dhandle = ar3DCreateHandle(cparam)) == NULL)
{ fprintf(stderr, "setupCamera(): Error: ar3DCreateHandle.\n");
return (FALSE); }
if (arVideoCapStart() != 0)
{ fprintf(stderr, "setupCamera(): Unable to begin camera data capture.\n"); return (FALSE);
}
return (TRUE); }
```

The second major part of ARToolKit setup is to load pattern files for each of the patterns to be detected. In simpleLite, only one pattern is tracked, the basic "Hiro" pattern. setupMarker creates a list of patterns for ARToolKit to track, and loads the Hiro pattern into it. Loading multiple patterns can be seen in the simpleVRML example,

```
static int setupMarker(const char *patt_name, int *patt_id, ARHandle *arhandle, ARPattHandle **pattHandle)
{ if ((*pattHandle = arPattCreateHandle()) == NULL)
{ fprintf(stderr, "setupCamera(): Error: arPattCreateHandle.\n");
return (FALSE); }
// Loading only 1 pattern in this example.
if ((*patt_id = arPattLoad(*pattHandle, patt_name)) < 0)
{ fprintf(stderr, "setupMarker(): pattern load error !!\n");
arPattDeleteHandle(*pattHandle);
return (FALSE); }
arPattAttach(arhandle, *pattHandle);
return (TRUE); }
```

mainLoop

This is the routine where the bulk of the ARToolKit function calls are made and it contains code corresponding to steps 2 through 5 of the required application steps. First a new

video frame is requested using the function arVideoGetImage. If the function returns non-NULL, a new frame has been captured, and the return value points to the buffer containing the frame's pixel data, so it is saved in a global variable.

```
// Grab a video frame.
if ((image = arVideoGetImage()) != NULL)
{ gARTImage = image; // Save the fetched image.
 gCallCountMarkerDetect++; // Increment ARToolKit FPS counter.
```

Every time a new frame has been acquired, it needs to be searched for markers. This is accomplished by a call to the function arDetectMarker(), passing in the pointer to the new frame, and an ARHandle. The ARHandle holds the ARToolKit marker detection settings and also stores the results of the marker detection.

```
// Detect the markers in the video frame.
if (arDetectMarker(gARHandle, gARTImage) < 0)
{ exit(-1); }
```

The results of the marker detection process can now be examined - to check whether they match the IDs of the marker(s) loaded earlier. Of course, in simpleLite - need to check for one marker, the Hiro marker. A value known as the marker confidence is used to make sure that the Hiro marker is obtained and not a marker with a different pattern.

```
// Check through the marker_info array for highest confidence //
 visible marker matching our preferred pattern.
k = -1; for (j = 0; j < gARHandle->marker_num; j++)
{ if (gARHandle->markerInfo[j].id == gPatt_id)
{ if (k == -1) k = j; // First marker detected.
else
 if (gARHandle->markerInfo[j].cf > gARHandle->markerInfo[k].cf) k = j;
// Higher confidence marker detected.
 } }
```

At the end of this loop, if k has been modified, then the marker containing the Hiro pattern is found, so the last task to be performed by ARToolKit on the marker is to retrieve its position and orientation (its "pose") relative to the camera. The pose is stored in an AR3DHandle structure. If the marker is not found, note that fact- because if no markers are found, 3D objects should not be drawn in the frame.

```
  if (k != -1)
{ // Get the transformation between the marker and the real camera into gPatt_trans.
err = arGetTransMatSquare(gAR3DHandle, &(gARHandle->markerInfo[k]),
gPatt_width, gPatt_trans);
gPatt_found = TRUE;
}
else
{ gPatt_found = FALSE;
}
```

Finally.. since there is a new video frame, request the operating system call our Display function: glutPostRedisplay();

Display

This program run two loops in parallel.. one (in mainLoop()) grabs images from the camera and looks for markers in them. The other loop displays images and 3D objects over

the top of detected marker positions. or other AR-related content that is to be drawn. These two loops run separately, because the operating system separates drawing from other regular tasks, to work more efficiently. In simpleLite, the drawing all happens in the function named Display(). (This gets called by the operating system via GLUT).
In the display function, several steps are done:

- Clear the screen and draw the most recent frame from the camera as a video background.
- Set up the OpenGL camera projection to match the calibrated ARToolKit camera parameters.
- Check whether any active marker is present, and if so, position the OpenGL camera view for each one to place the coordinate system origin onto the marker.
- Draw objects on top of any active markers (using the OpenGL camera).

Step 1: Clear the screen and draw the most recent frame from the camera as a video background:

```
// Select correct buffer for this context.
glDrawBuffer(GL_BACK);
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
// Clear the buffers for new frame.
 arglDispImage(gARTImage, &gARTCparam, 1.0, gArglSettings); // zoom = 1.0.
gARTImage = NULL;
```

The video image is this displayed on screen. This can either be an unwarped image, or an image warped to correct for camera distortions. Unwarping the camera's distorted image helps the virtual 3D objects appear in the correct place on the video frame.

Step 2: Set up the OpenGL camera projection to match the calibrated ARToolKit camera parameters.

```
// Projection transformation.
arglCameraFrustumRH(&gARTCparam, VIEW_DISTANCE_MIN,
VIEW_DISTANCE_MAX, p); glMatrixMode(GL_PROJECTION);
 glLoadMatrixd(p);
glMatrixMode(GL_MODELVIEW);   // Viewing transformation. glLoadIdentity();
```

The call to arglCameraFrustumRH converts the camera parameters stored in gARTCparam into an OpenGL projection matrix p, which is then loaded directly, setting the OpenGL camera projection. With this, the field-of-view, etc. of the real camera will be exactly matched in the scene.

Step 3: Check for any active markers, and if so, position the OpenGL camera view for each one to place the coordinate system origin onto the marker.

```
if (gPatt_found)
{
// Calculate the camera position relative to the marker.
// Replace VIEW_SCALEFACTOR with 1.0 to make one drawing unit equal to 1.0
ARToolKit units (usually millimeters).
arglCameraViewRH(gPatt_trans, m, VIEW_SCALEFACTOR);
glLoadMatrixd(m);
```

arglCameraViewRH converts the marker transformation (saved in mainLoop) into an OpenGL modelview matrix. These sixteen values are the position and orientation values of the real camera, so using them to set the position of the virtual camera causes any graphical objects to be drawn to appear exactly aligned with the corresponding physical marker. The virtual camera position is set using the OpenGL function glLoadMatrixd(m).

Step 4: Draw objects on top of any active markers (using the OpenGL camera).

Finally, The last part of the code is the rendering of 3D object, in this example the OpenGL colour cube. This function is simply an example and can be replaced with any drawing code. Simply, at the time the draw function is called, the OpenGL coordinate system origin is exactly in the middle of the marker, with the marker lying in the x-y plane (x to the right, y upwards) and with the z axis pointing towards the viewer. If you are drawing directly onto a marker, remember not to draw in the -z part of the OpenGL coordinate system, or else your drawing will look odd, as it will be drawn "behind" the marker.

cleanup

The cleanup function is called to stop ARToolKit and release resources used by it, in a clean manner:

```
static void cleanup(void)
{ arglCleanup(gArglSettings);
arPattDetach(gARHandle);
arPattDeleteHandle(gARPattHandle);
arVideoCapStop();
ar3DDeleteHandle(gAR3DHandle);
arDeleteHandle(gARHandle); arVideoClose(); }
```

Cleanup steps are generally performed in reverse order to setup steps.

Steps in cleaning up ARToolKit.

Cleanup step	!Functions to be called.
remove ARToolKit's connections to OpenGL	arglCleanup
release resources used in the marker tracking	arPattDetach, arPattDeleteHandle, ar3DDeleteHandle, arDeleteHandle
stop the video capture and close down the video path to free it up for other applications	arVideoCapStop, arVideoClose