Augmented Reality on mobile devices

Laboratorio di Realtà Virtuale

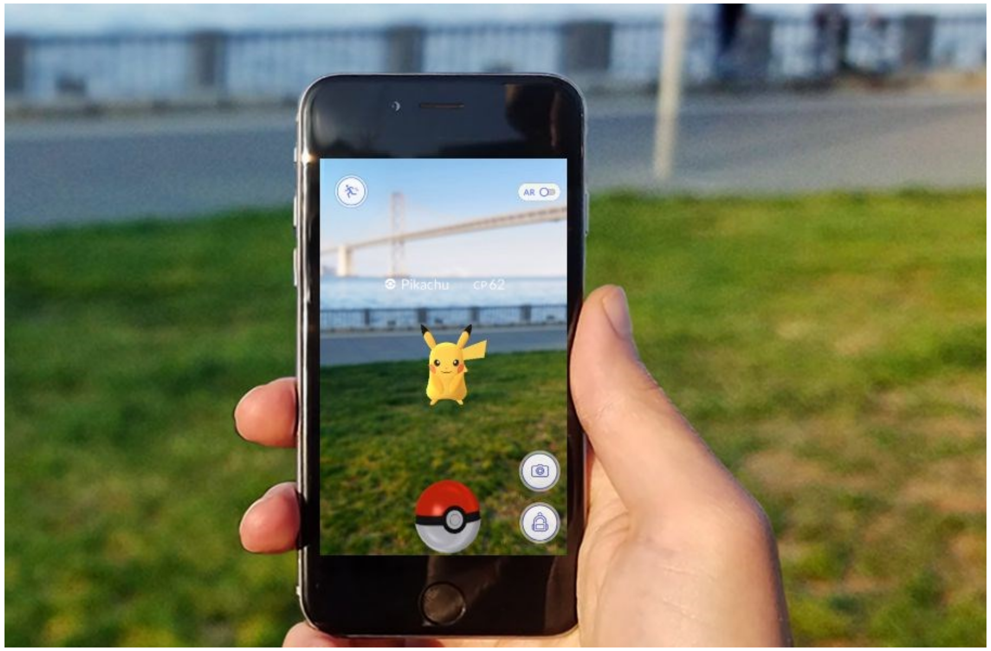
Manuel Pezzera – manuel.pezzera@unimi.it

What is augmented reality?

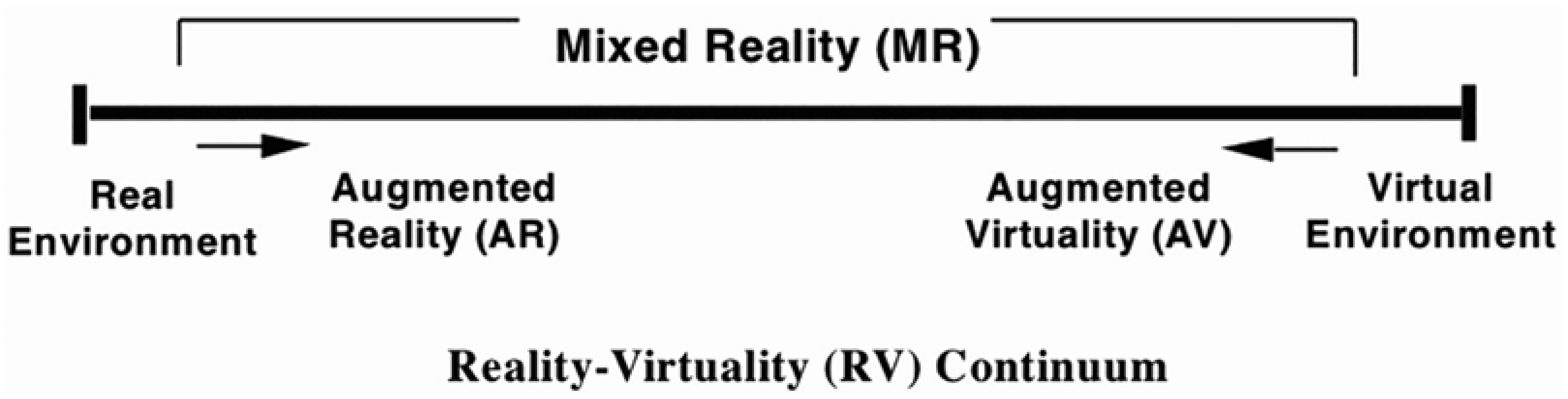
A combination of a real scene viewed by a user and a virtual scene generated by a computer the augments the scene with additional information

What is augmented reality?

An augmented reality device adds new virtual objects and audio to the real-world environment in real time



From reality to VR (passing by AR)



Nothing

Video through device HUD

HMD

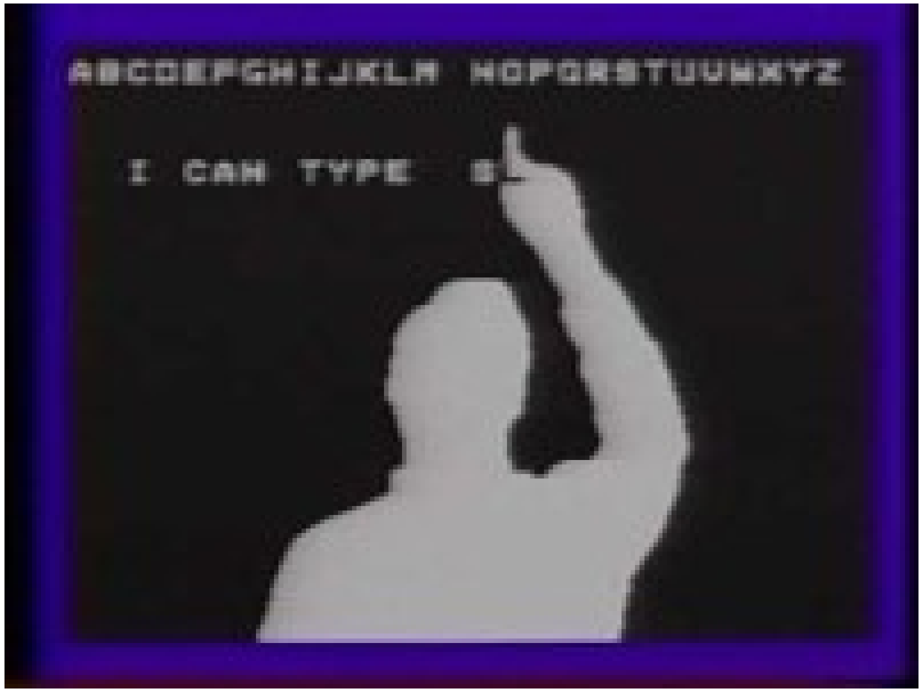
History

- In 1968, a Harvard professor and computer scientist by the name of Ivan Sutherland invented what he called The Sword of Damocles. He invented this first sort of augmented reality device with his student, Bob Sproull.

https://www.youtube.com/watch?v=eVUgfUvP4uk

History

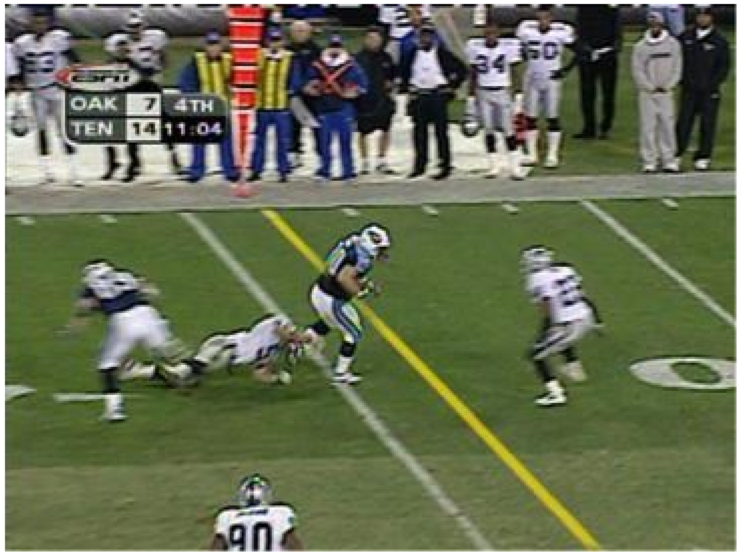
- One of the next big developments in augmented reality was in 1974 by Myron Krueger. The project was called, Videoplace, which combined a projection system and video cameras that produced shadows on the screen. This setup made the user feel as though they were in an interactive environment.



https://www.youtube.com/watch?v=d4DUIeXSEpk

History

- In 1998, Sportsvision uses the 1st and Ten line computer system. This system showed the original virtual yellow first down marker during a live NFL game. A variation of this virtual first down marker is now a norm in all televised football games today and is a big part of the augmented reality history.



ARToolKit

- An important advancement in the augmented reality technology happened in 2000 when Hirokazu Kato from the Nara Institute of Science and Technology in Japan created and released software called ARToolKit. Through this software, one could capture real-world actions and combine it with interactions of virtual objects.

- ARToolKit has been acquired by DAQRI in 2015, it is now called artoolkitX and is currently one of the most used open source library for augmented reality.

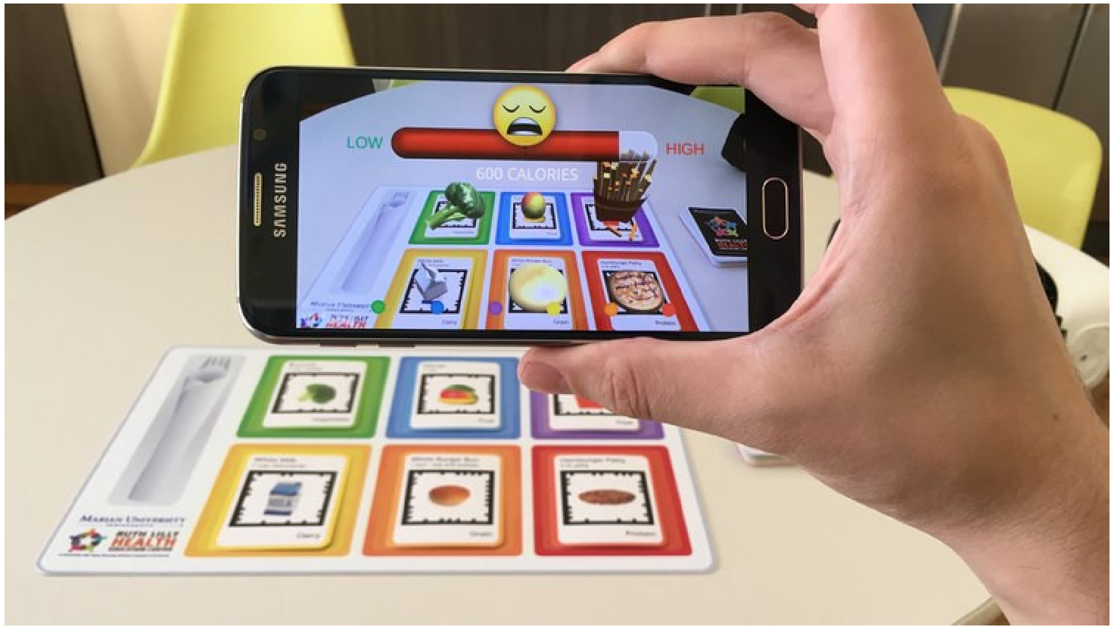
Augmented reality vs virtual reality

- Augmented reality:
  - System augments the real-world scene
  - User maintains a sense of presence in the real world (he/she still sees the real environment)
  - Needs a mechanism to combine virtual and real world

- Virtual reality:
  - Totally immersive environment
  - Visual sense are under control of the system

AR: how it works

- Pick a real-world scene.
- Add your virtual objects in it.
- Delete real world objects (if necessary).
- It's not virtual reality since environment is real.



Applications

- Medical
- Entertainment
- Military training
- Engineering design
- Robotics and telerobotics
- Manufacturing, maintenance and repair
- Consumer design
- Hazard detection

Application

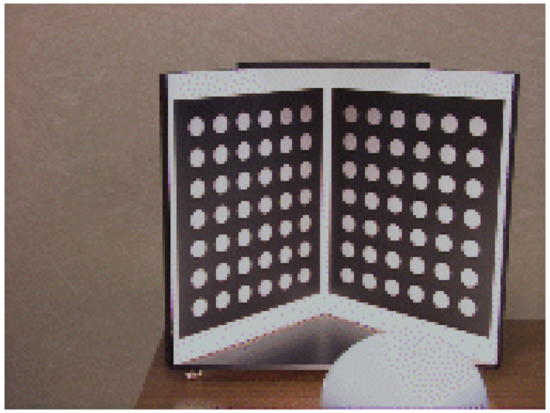
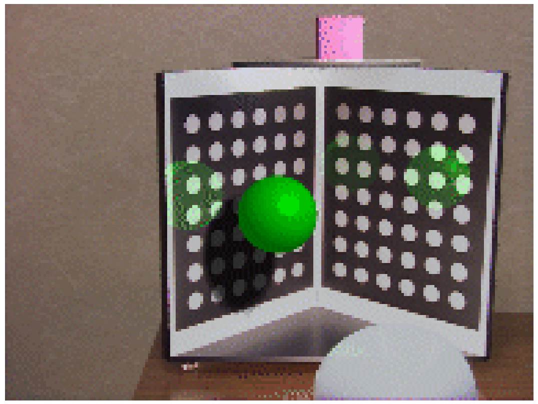
https://www.youtube.com/watch?v=RnEzrChVhEI&ab_channel=esimplestudios

Combining the real and the virtual world

- We need:
  - Locations and optical properties of the viewer (or camera) and the display.
  - Calibration of all devices.
  - To combine all local coordinate system centered on the devices and the objects in the scene in a global coordinate system.

Realistic merging

- Realistic merging requires:
  - Objects to behave in physically plausible manners when manipulated
  - Occlusion
  - Collision detection
  - Shadows

Performance issue

- Two performance criteria are placed on the system:
  - Update rate for generating the augmenting image
  - Accuracy of the registration of the real and virtual image
    - Update rate can limit registration accuracy as well

Failures in registration

- Failures in registration due to:
  - Noise
  - Image distortions
  - Time delays
    - In calculating the camera position

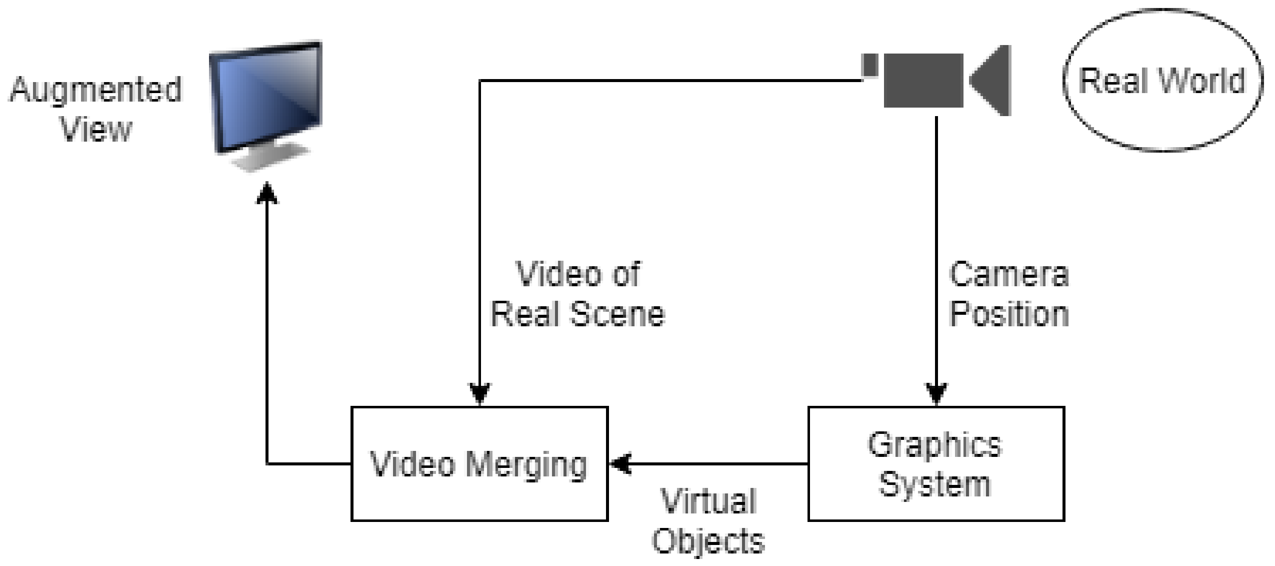
Display technologies

- Monitor based
  - Laptops
  - Cell phones
  - Projectors

- Head mounted displays:
  - Video see-through
  - Optical see-through

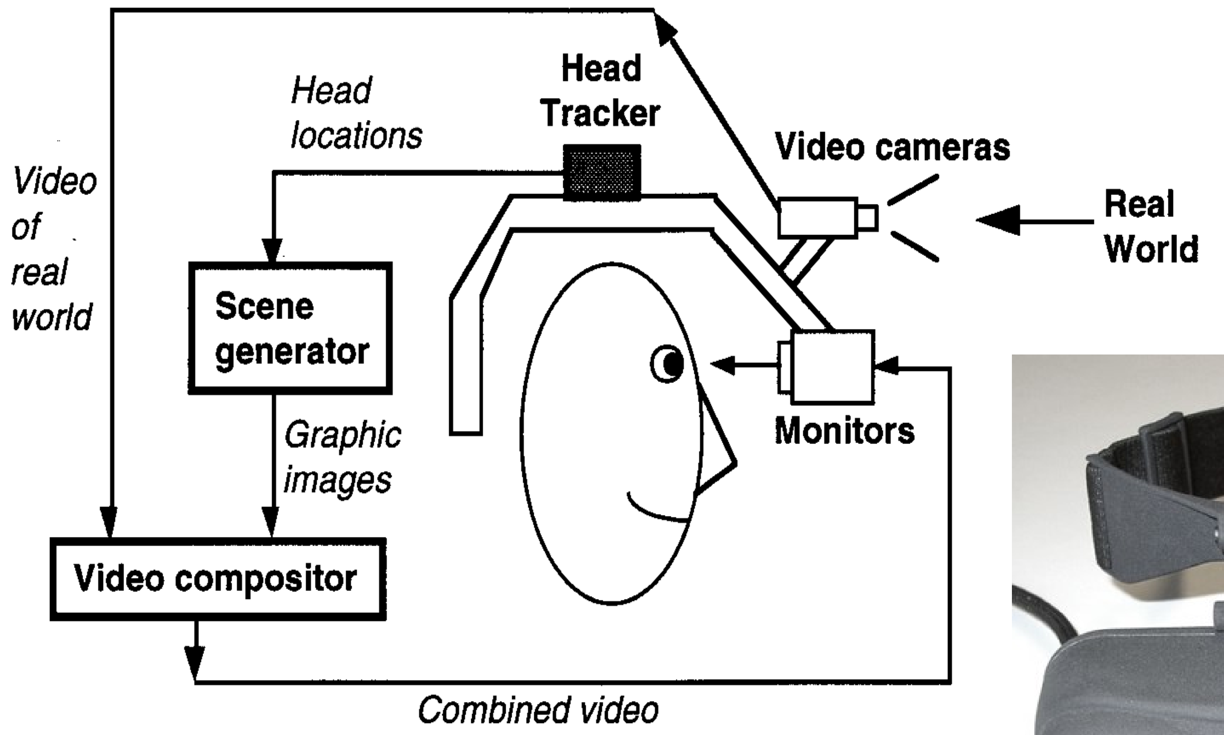
Monitor based augmented reality

- Simplest available
- Treat laptop/PDA/cell phone as a window through which you can see AR world.



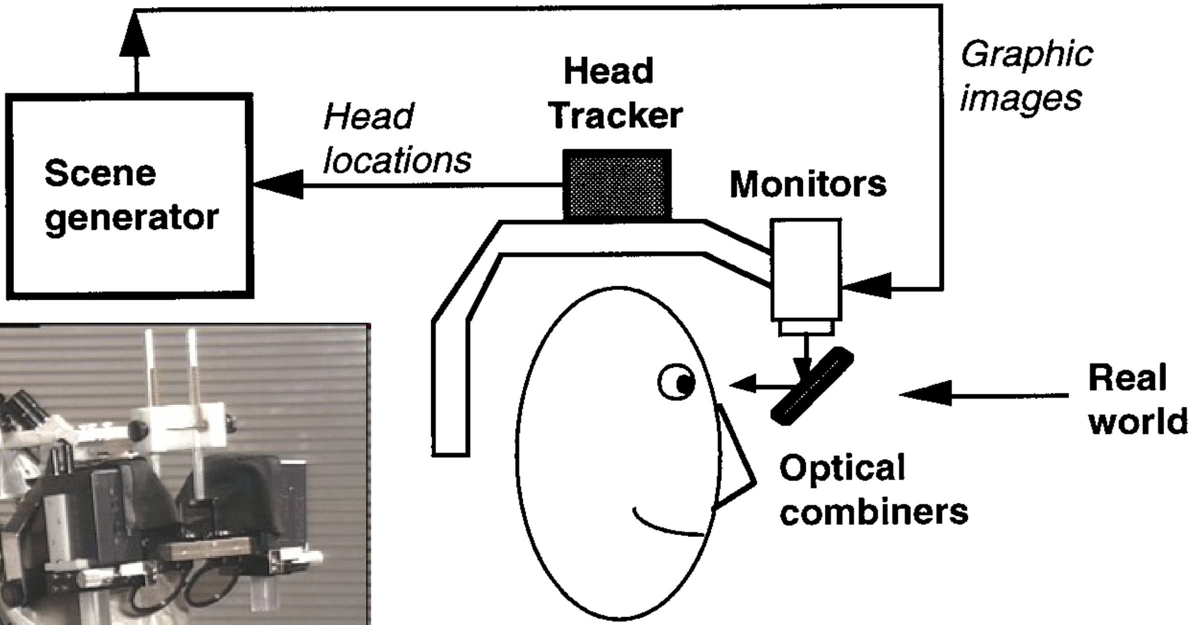
- Consumer-level equipment
- Most practical
- A lot of current research aimed here

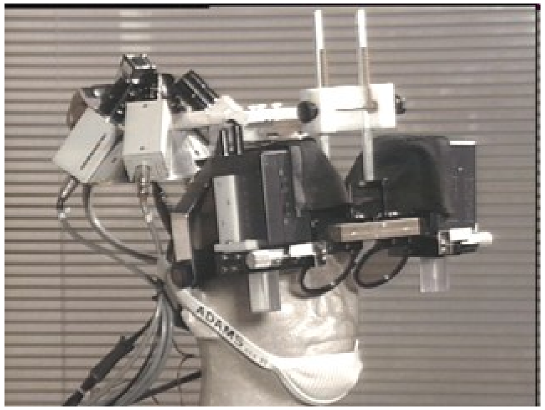
Video see-through HMD



- Advantages:
  - Flexibility in composition strategies
  - Real and virtual view delays can be matched

Optical see-through HMD





- Advantages
  - Simplicity
  - Resolution

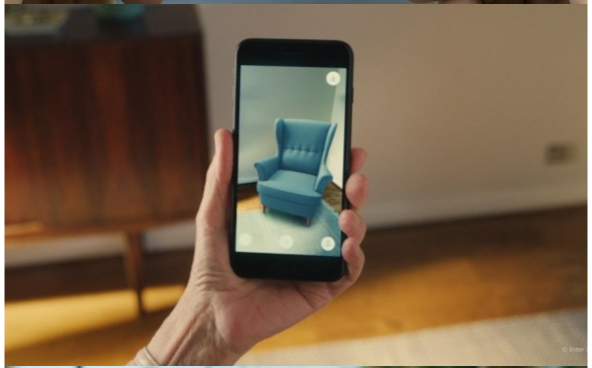
Mobile augmented reality

What is mobile Augmented Reality?

- Overlaying virtual information onto the real world
  - Using mobile devices
  - On the local surroundings

- Smartphones & tables are suitable platforms for AR applications
  - Growing availability and computing power
  - Integration of sensors that are necessary to realize AR (compass, GPS, camera…)

- Fields of applications
  - Marketing and advertisement (e.g. IKEA app)
  - Tourism (display of points of interests)
  - Games



AR types

- What content do we display on the live camera view? And where exactly should we put that content within the user's view?
- The answer to these questions depends on which application of AR you choose:
  - Marker based
  - Markerless/Location based

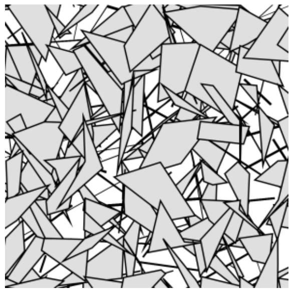
Marker based AR

- In some cases, we need to know exactly what the user is looking at, this is called "marker-based AR". In this mode the digital world is anchored to the real world.

- Therefore, the device must first recognize which page you're looking at from the live camera view. This can be achieved by placing a distinctive picture or shape on the page. That picture will be recognized and the animation can start immediately, tracked to the appropriate place on the page.

- We call the distinctive picture that can be recognized by the device, the marker. A marker can be anything, as long as it has enough unique visual points. Images with lots of corners and edges work especially well.

    - Marker generator: http://www.brosvision.com/ar-marker-generator/



Location based AR

- Imagine an AR application that can place virtual furniture inside your living room. This allows you to try different combinations of objects, styles and location. For this application, the user needs to decide where to place the virtual object. This is called “location based” or “markerless” AR.

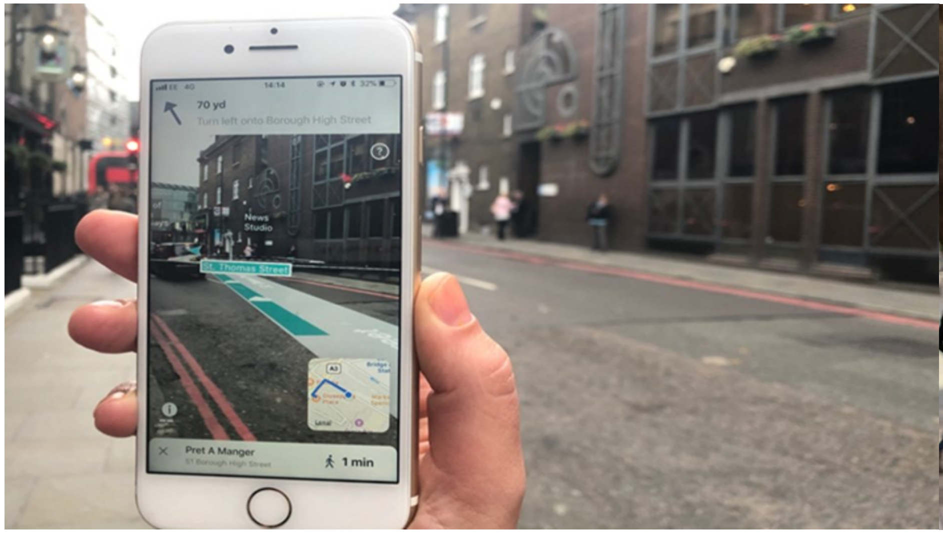
- Sometimes, all we want is to place a virtual video-game character right in front of the user. The interaction here is the game-play, and the user does not need to worry about the location of the virtual objects. Use markerless AR for those applications that do not require an “anchor” to the real world.

- Typically, this means virtual objects will appear to “float” in mid-air. But it is also possible to automatically place a 3D augmented reality object onto a flat surface to increase realism, for instance placing a lamp onto a table.

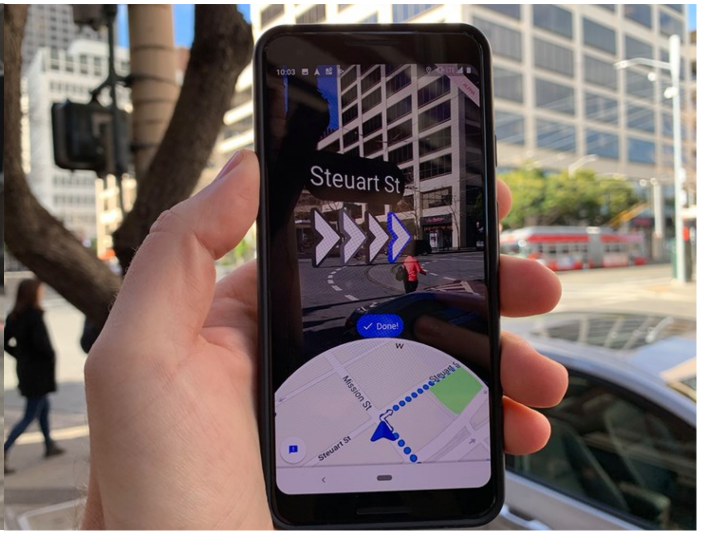
Location based AR

- Location based AR ties augmented reality content to a specific location. Imagine walking in a city street you're not familiar with and through your phone's camera seeing a virtual road sign displaying the street name, this is location based AR. Example: [AR City app](#).

- To make sure the digital AR content appears in exactly the right place, your device must accurately figure out its location. The AR City app relies on a combination of GPS, the compass sensor on your phone, and a computer vision system, to place virtual objects at the right location.

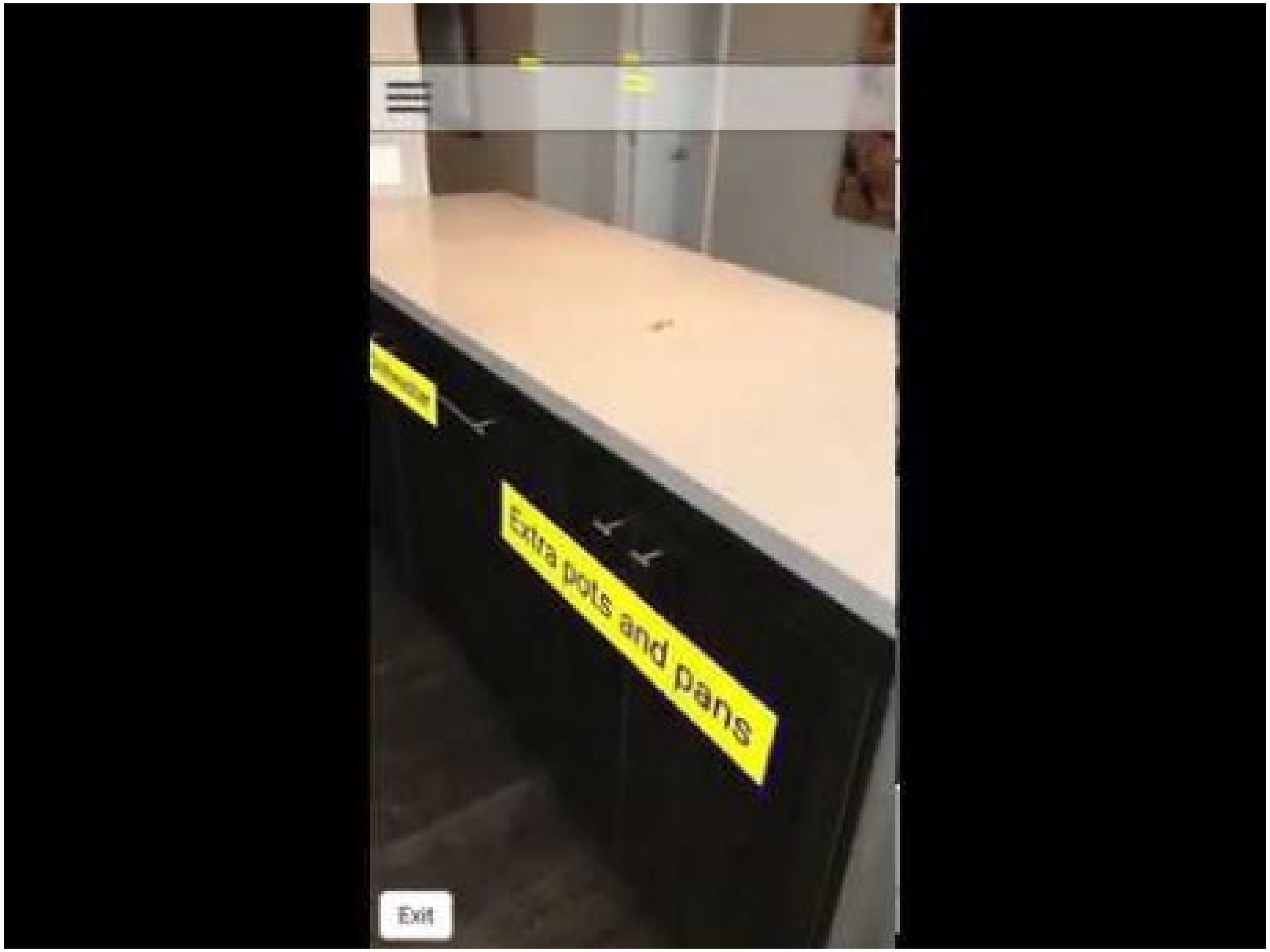


AR City App



Google Maps AR Preview

Location-based AR



https://www.youtube.com/watch?v=evAtusdRJjY&ab_channel=PlacenoteSDK

Location-based AR vs Marker-based AR

Location-based AR

Technical realization
- GPS sensor for determination of position
- IMU sensors (compass, accelerometer, gyroscope) for determination of orientation

Pro and cons
- Robust, modest and fast technology
- Suitable in large-scale environments
- Imprecise

Marker-based AR

Technical realization
- Analysis of camera image for determination of pose & orientation (feature tracking)
- Registration within virtual 3D model

Pros and cons
- Very precise and realistic rendering possible
- Complex, error-prone technology (light condition, moving objects…)
- Not suitable in large-scale environments

Google ARCore

- ARCore is Google's platform for building augmented reality experiences. Using different APIs, ARCore enables your phone to sense its environment, understand the world and interact with information. Some of the APIs are available across Android and iOS to enable shared AR experiences.

- ARCore uses three key capabilities to integrate virtual content with the real world as seen through your phone's camera:
  - **Motion tracking** allows the phone to understand and track its position relative to the world.
  - **Environmental understanding** allows the phone to detect the size and location of all type of surfaces: horizontal, vertical and angled surfaces like the ground, a coffee table or walls.
  - **Light estimation** allows the phone to estimate the environment's current lighting conditions.

- Supported devices: https://developers.google.com/ar/discover/supported-devices

How does AR Core work?

- ARCore is doing two things: tracking the position of the mobile device as it moves and building its own understanding of the real world.

- ARCore's motion tracking technology uses the phone's camera to identify interesting points, called **features**, and tracks how those points move over time. With a combination of the movement of these points and readings from the phone's inertial sensors, ARCore determines both the position and orientation of the phone as it moves through space.

- ARCore can also detect flat **surfaces**, like a table or the floor, and can also estimate the average lighting in the area around it. These capabilities combine to enable ARCore to build its own understanding of the world around it.

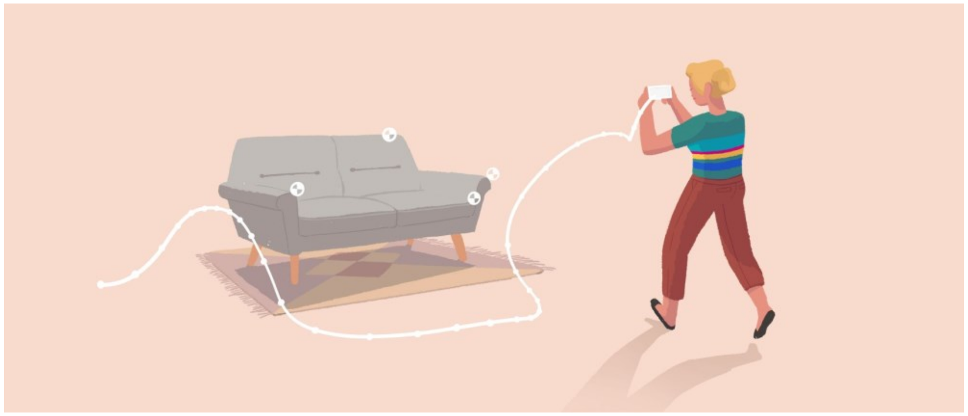
- ARCore's understanding of the real world lets you place objects, annotations, or other information in a way that integrates seamlessly with the real world.

AR Core: Motion Tracking

- As your phone moves through the world, ARCore uses a process called concurrent odometry and mapping, to understand where the phone is relative to the world around it.

- ARCore detects visually distinct features in the captured camera image called **feature points** and uses these points to compute its change in location. The visual information is combined with inertial measurements from the device's IMU to estimate the **pose** (position and orientation) of the camera relative to the world over time.

- By aligning the pose of the virtual camera that renders your 3D content with the pose of the device's camera provided by ARCore, developers are able to render virtual content from the correct perspective. The rendered virtual image can be overlayed on top of the image obtained from the device's camera, making it appear as if the virtual content is part of the real world.



Light estimation

- ARCore can detect information about the lighting of its environment and provide you with the average intensity and color correction of a given camera image. This information lets you light your virtual objects under the same conditions as the environment around them, increasing the sense of realism.

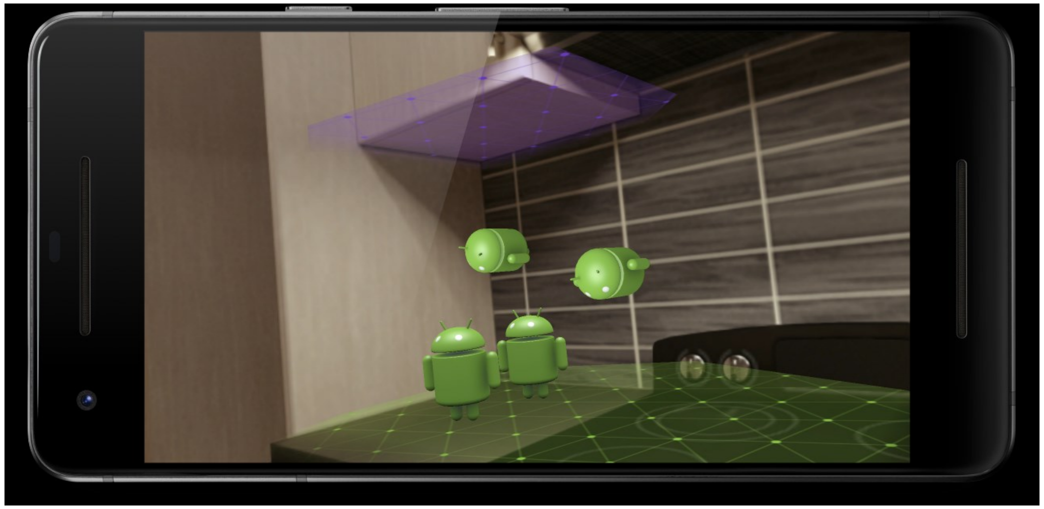
User interaction and oriented points

- **User interaction**: ARCore uses hit testing to take an (x,y) coordinate corresponding to the phone's screen (provided by a tap or whatever other interaction you want your app to support) and projects a ray into the camera's view of the world, returning any planes or feature points that the ray intersects, along with the pose of that intersection in world space. This allows users to select or otherwise interact with objects in the environment.

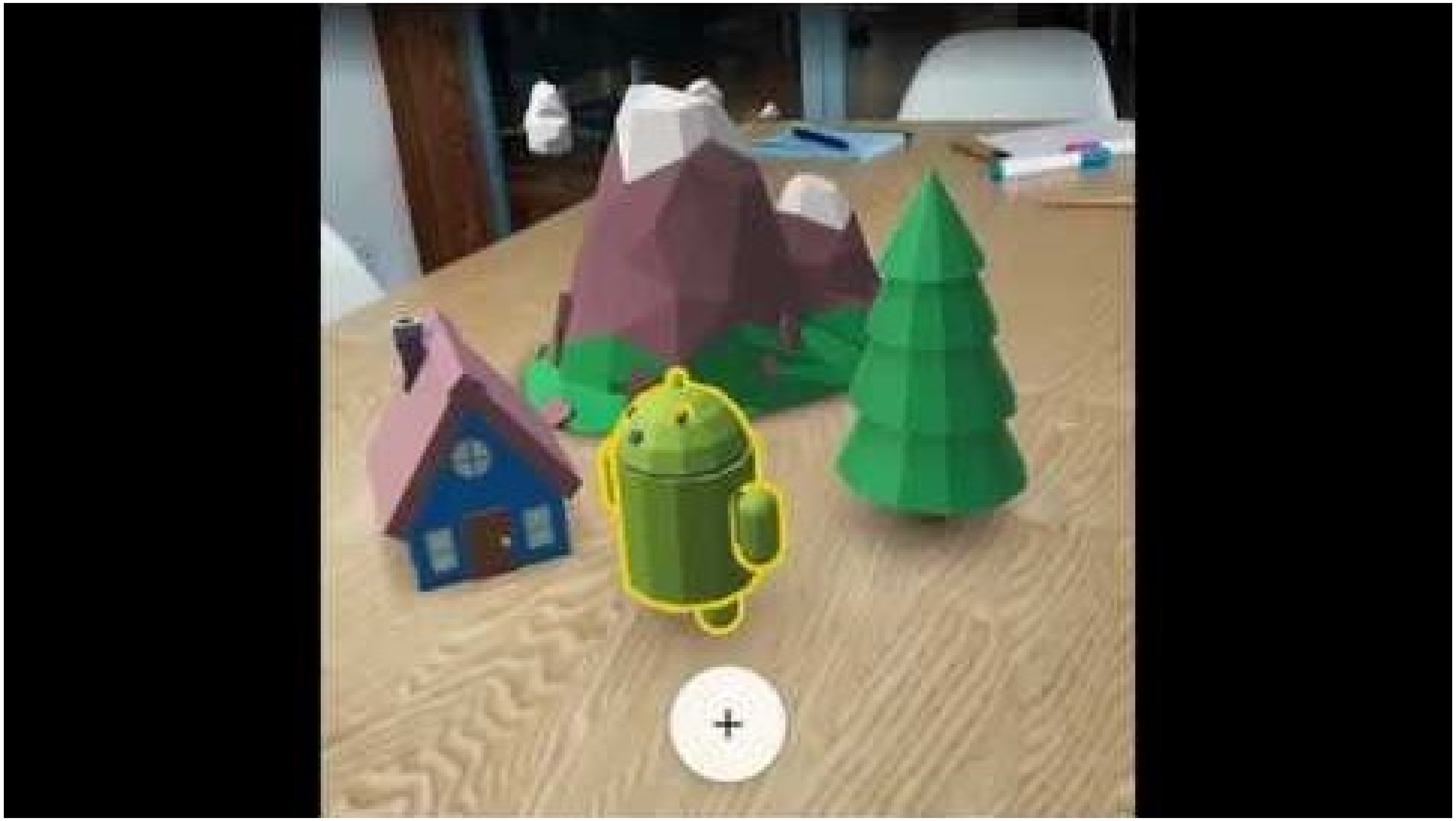
- **Oriented points**: Oriented points lets you place virtual objects on angled surfaces. When you perform a hit test that returns a feature point, ARCore will look at nearby feature points and use those to attempt to estimate the angle of the surface at the given feature point. ARCore will then return a pose that takes that angle into account. Because ARCore uses clusters of feature points to detect the surface's angle, surfaces without texture, such as a white wall, may not be detected properly.

Anchors and trackables

- When you want to place a virtual object, you need to define an **anchor** to ensure that ARCore tracks the object's position over time. Often you create an anchor based on the pose returned by a hit test, as described in user interaction.
- The fact that poses can change means that ARCore may update the position of environmental objects like planes and feature points over time. Planes and points are a special type of object called a **trackable**.
- These are objects that ARCore will track over time. You can anchor virtual objects to specific trackables to ensure that the relationship between your virtual object and the trackable remains stable even as the device moves around.
    - This means that if you place a virtual Android figurine on your desk, if ARCore later adjusts the pose of the plane associated with the desk, the Android figurine will still appear to stay on top of the table.



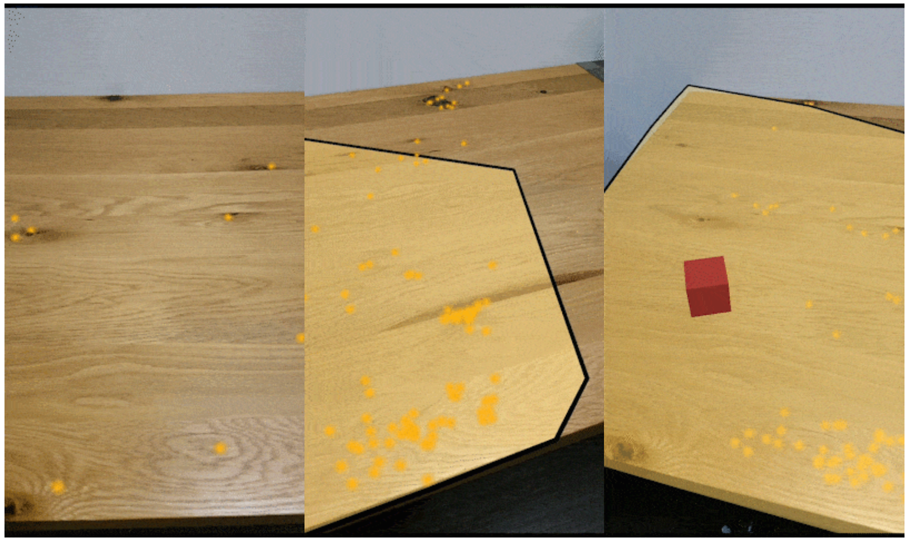
AR Core Demo



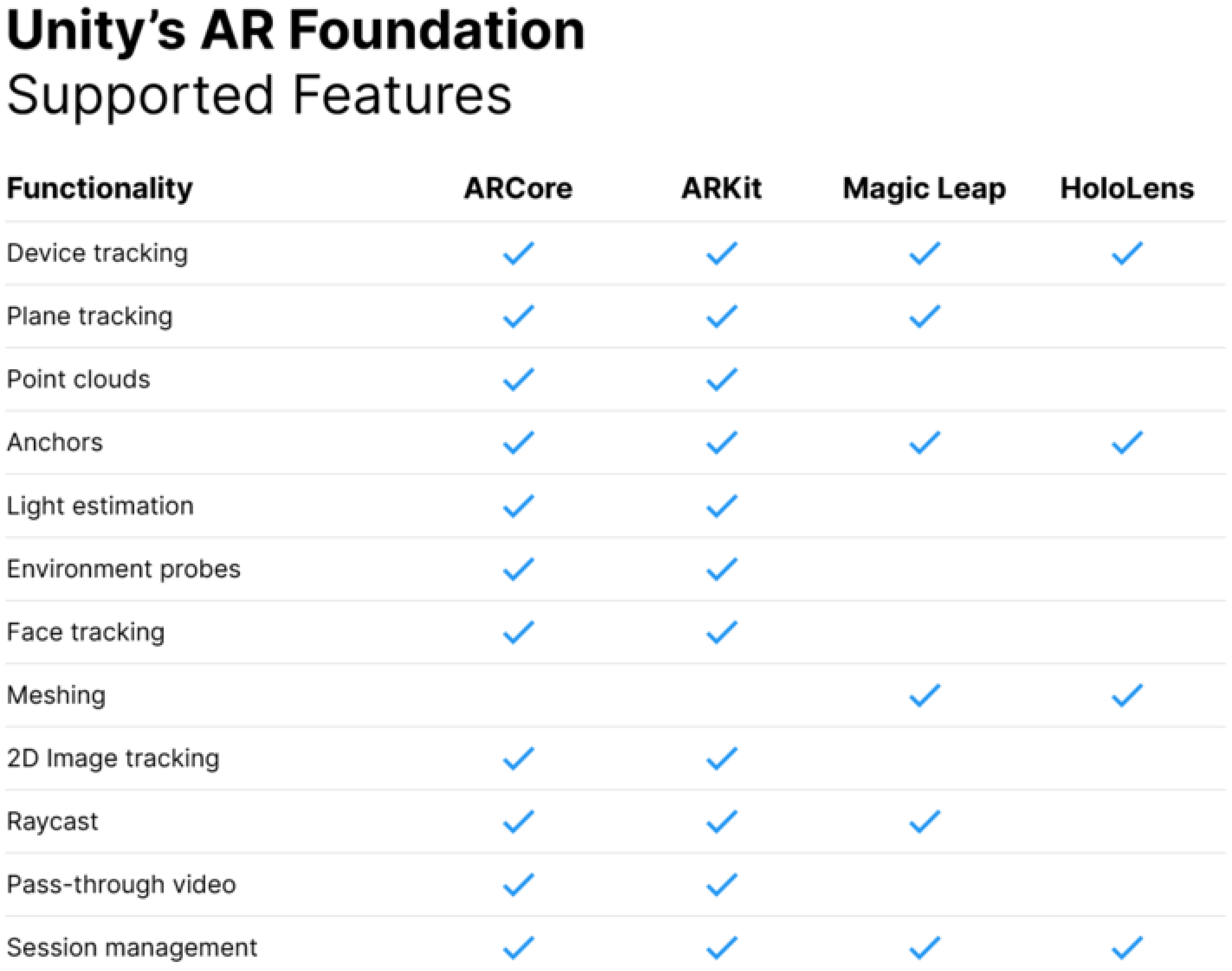
https://www.youtube.com/watch?v=cape_Af9j7w

Unity AR Foundation

- Unity AR Foundation offers a common API which supports core functionality for ARCore, ARKit and future platforms.
- AR Foundation provides support for the core functionality of most AR apps:
  - Planar surface detection
  - Depth data represented as point clouds
  - Performant pass-thru rendering
  - Reference points to aid in anchoring virtual objects to physical world
  - Estimated for average color temperature and brightness
  - Tracking device position and orientation in physical space
  - Utilities for scaling content properly in AR
  - Raycasting against plane and depth data



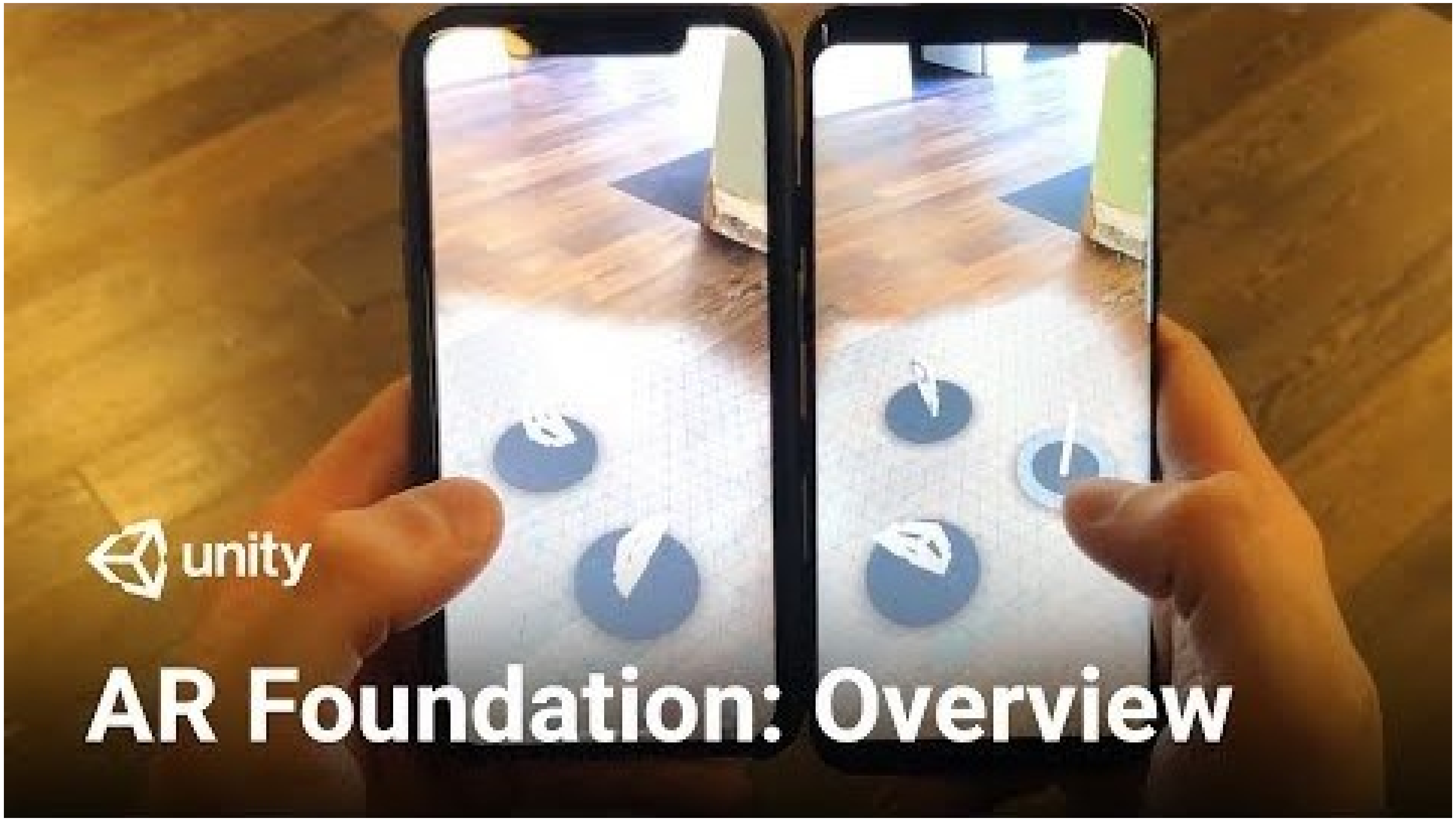
Unity AR Foundation



Unity's AR Foundation
Supported Features

Functionality	ARCore	ARKit	Magic Leap	HoloLens
Device tracking	✓	✓	✓	✓
Plane tracking	✓	✓	✓	
Point clouds	✓	✓		
Anchors	✓	✓	✓	✓
Light estimation	✓	✓		
Environment probes	✓	✓		
Face tracking	✓	✓		
Meshing			✓	✓
2D Image tracking	✓	✓		
Raycast	✓	✓	✓	
Pass-through video	✓	✓		
Session management	✓	✓	✓	✓

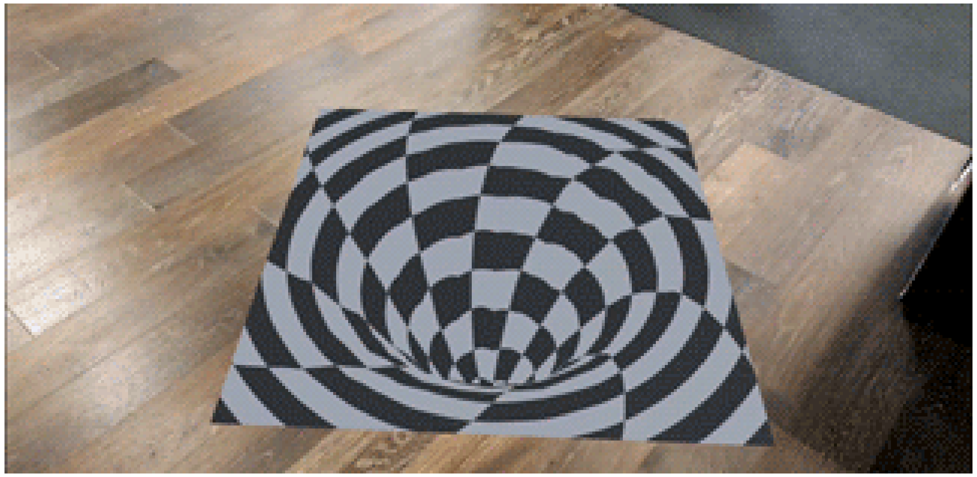
AR Foundation



https://www.youtube.com/watch?v=ml9qVRdEH4k&ab_channel=Unity

AR Foundation

- Compatible with URP
- Camera Image APIs
- Face Tracking (ARKit)



Vuforia

- One of the most used augmented reality software development kit is Vuforia.
- It uses computer vision technology to recognize and track planar images and simple 3D objects, such as boxes, in real time. This enables developers to position and orient virtual objects, such as 3D models and other media, in relation to real world images when they are viewed through the camera of a mobile device.
- Also available for Unity.

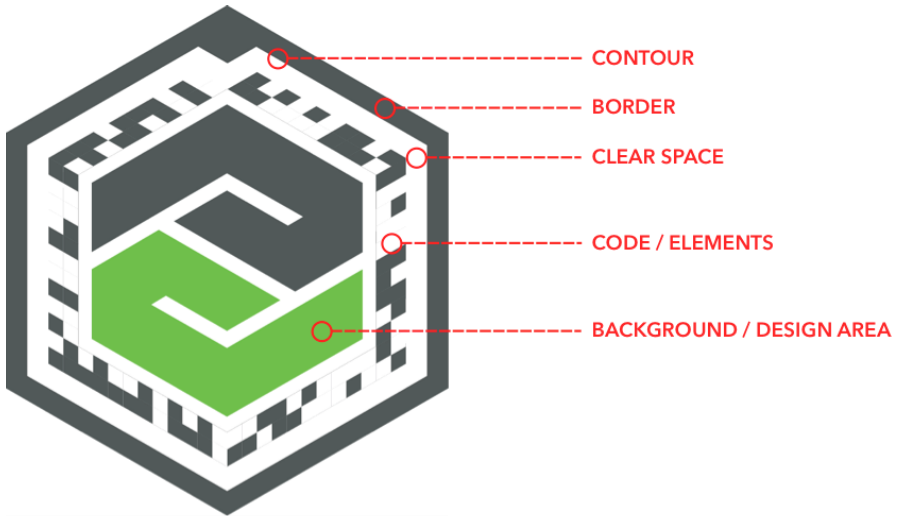
Unity & AR: Vuforia

- Vuforia uses marker to analyze the real scene and to decide if and where put virtual objects.
- Vuforia detects “feature points” in your target image, and then uses the data to compare the features in target image and the receiving frame from camera. Vuforia is not open source, implementation details are unknown.
- Usually, a marker is something similar to the following image.

VuMark

- VuMark is the next generation bar code. It allows the freedom for a customized and brand-conscious design while simultaneously encoding data and acting as an AR target. VuMark designs are completely customizable, so you can have a unique VuMark for every unique object.

- The VuMark provides a universal solution for delivering unique AR experiences on any object, while allowing the design freedom for a custom look and feel. The VuMark also provides a simple method for encoding data such as an URL or a product serial number, and overcomes the limitations of existing matrix bar code solutions that do not support AR experiences and can detract from a product's appearance.



VuMarks vs Image Targets

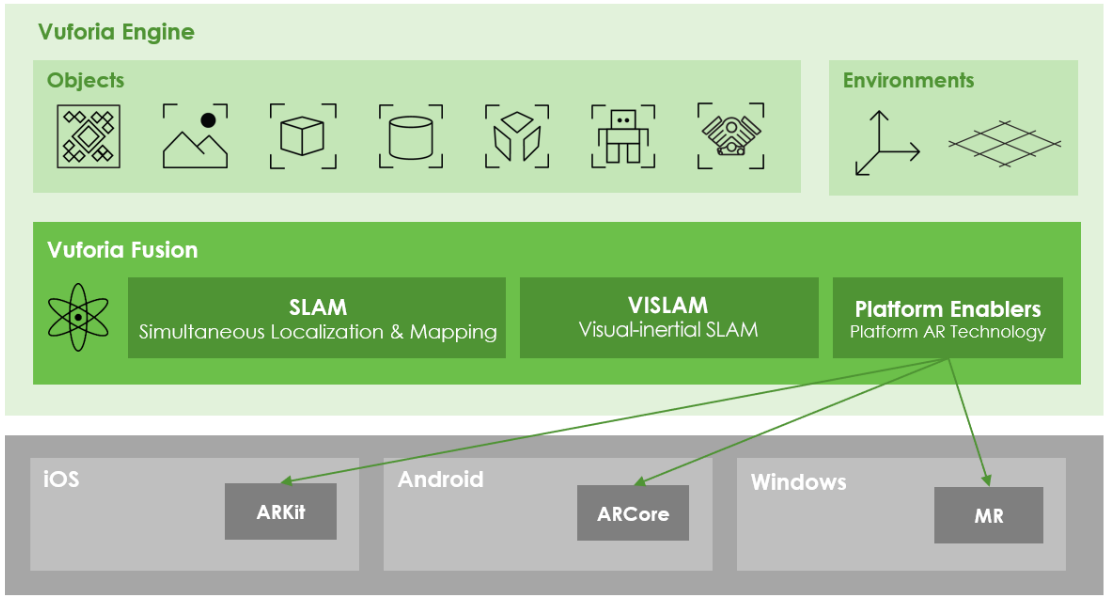
- VuMarks provide some of the same capabilities as Vuforia ImageTargets, they can be individually recognized and tracked by the Vuforia SDK. Both can be used to create rich Augmented Reality experiences.
- But there are also significant differences that make VuMarks especially useful for many enterprise and consumer applications.
    - VuMarks can present millions of uniquely identifiable instances
    - VuMarks can encode a variety of data formats
    - VuMarks enable you to differentiate among identical looking products based on their Instance ID

Vuforia Ground Plane

- Ground plane is a Vuforia script that allows developers to spawn objects without the use of a marker, the Vuforia library scans the environment looking for a ground plane and when it finds it, it spawn your content.

Vuforia Fusion

- Fusion solves the problem of fragmentation in AR-enabling technologies, including cameras, sensors, chipsets, and software frameworks such as ARKit and ARCore.
- It senses the capabilities of the underlying device and fuses them with Vuforia Engine features, allowing developers to rely on a single Vuforia Engine API for an optimal AR experience.
- Vuforia Fusion brings advanced Vuforia Engine features to devices that support ARKit and ARCore, in addition to other Android and iOS device models.



Cloud Recognition

- The Vuforia Cloud Recognition service is an enterprise class Image Recognition solution that enables developers to host and manage Image Targets online.
- The Vuforia Cloud Recognition service is ideally suited for apps that use many targets, or targets that need to be updated frequently. Benefits:
  - Scale: More than one million targets can be used in an app.
  - Flexibility: Integrate with existing content management systems.
  - Time to Market: Deliver real-time, dynamically changing content and accelerate time to market

Objects Recognition

- Until now we described different types of 2D markers, but it is also possible to use 3D objects as markers.
- Object Recognition allows you to detect and track intricate 3D objects. It has been designed to work with toys (such as action figures and vehicles) and other consumer products.
- 3D objects should be:
  - Opaque, they should not be transparent or semi-transparent
  - Rigid
  - Minimum moving parts

Unity Demo!

Let's try Vuforia with Unity

Unity & Vuforia

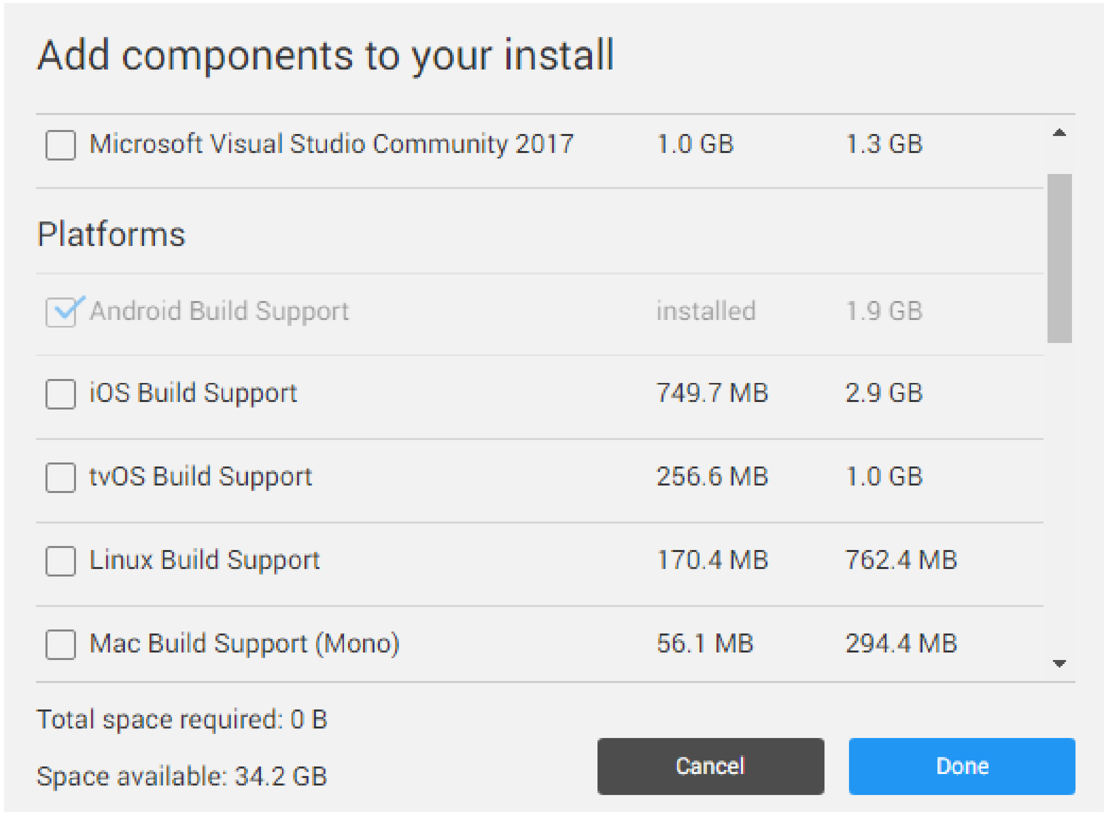
- To install Vuforia, go to:
  - https://developer.vuforia.com/
- Register an account (you need it both to download the Vuforia package for Unity, and to obtain a license)
- Once you have registered an account, click on “Downloads”, and then on “Add Vuforia Engine to a Unity Project….”

- Download the package, create a new empty project in Unity, import the package and wait for its importing.

- Download the Vuforia Core Samples
  - https://assetstore.unity.com/packages/templates/packs/vuforia-core-samples-99026

Unity & Vuforia: Camera

- If you want to deploy to Android you need to install also the Android component.
- From Unity Hub, click on the "…" button on the right, corresponding to the version of Unity you want to use
  - Add Component
  - Select "Android Build Support"



Add components to your install

☐ Microsoft Visual Studio Community 2017	1.0 GB	1.3 GB
Platforms		
✓ Android Build Support	installed	1.9 GB
☐ iOS Build Support	749.7 MB	2.9 GB
☐ tvOS Build Support	256.6 MB	1.0 GB
☐ Linux Build Support	170.4 MB	762.4 MB
☐ Mac Build Support (Mono)	56.1 MB	294.4 MB

Total space required: 0 B

Space available: 34.2 GB

Unity & Vuforia

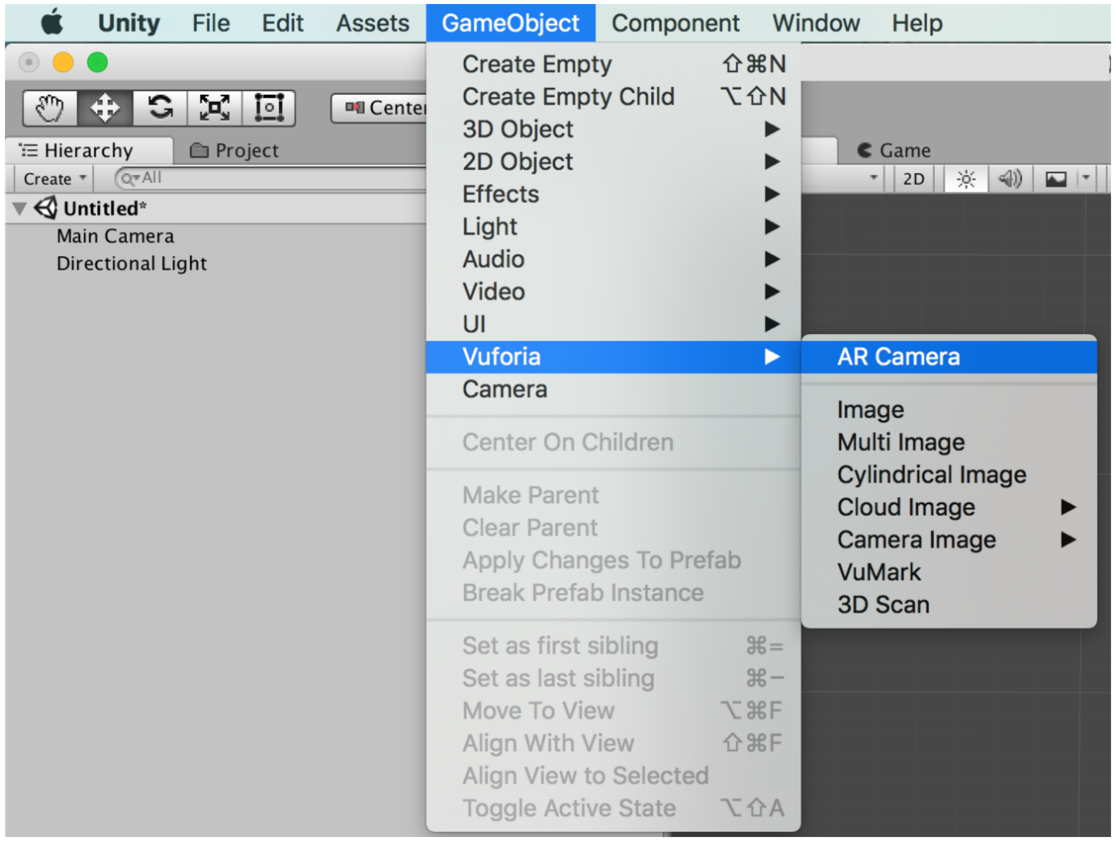
- Register a new Vuforia account: https://developer.vuforia.com
- If you want to distribute a Vuforia application, you need to pay the Vuforia license which is quite expensive. For testing you don’t need a license.
- Get a development key: https://developer.vuforia.com/vui/develop/licenses
- Clicking on the development key you will be redirect to a page containing the license key. Copy and paste it in the “App License Key” field in the “VuforiaConfiguration.asset” file.

Vuforia: create new target manager

- Open Vuforia Target Manager: https://developer.vuforia.com/vui/develop/databases
- Click "Add Database"
    - Type "Device"
- Select the just created DB and create two new targets
    - "Add Target" button
    - Select the image, the width and a name.
    - Your image will be ranked from 0 to 5
        - Images with rank 0 cannot be use.
        - To generate a 5-star image you can use the marker generator: http://www.brosvision.com/ar-marker-generator/

Unity & Vuforia: Camera

- When you use Vuforia you need to use a new type of camera: the **AR Camera**.



Vuforia: create new target manager

- Right click in the Inspector:
    - Vuforia Engine -> Image Target

- Select the new object
- From the inspector, in the “Image Target Behaviour” script:
    - Assign your database in the “Database” drop-down options.
        - Your database will have the same name you assigned in the Vuforia website.
    - Select your image target

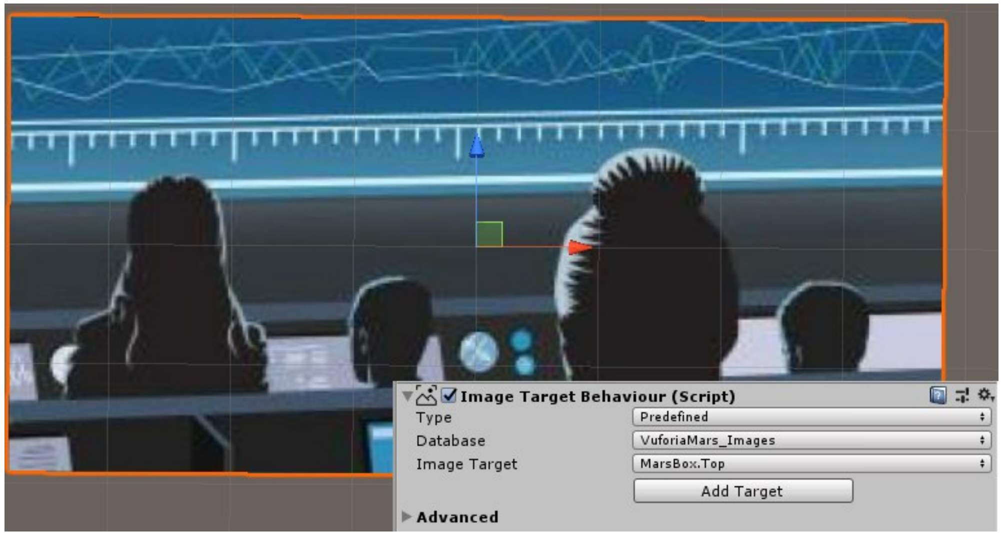


Image Target Behaviour (Script)

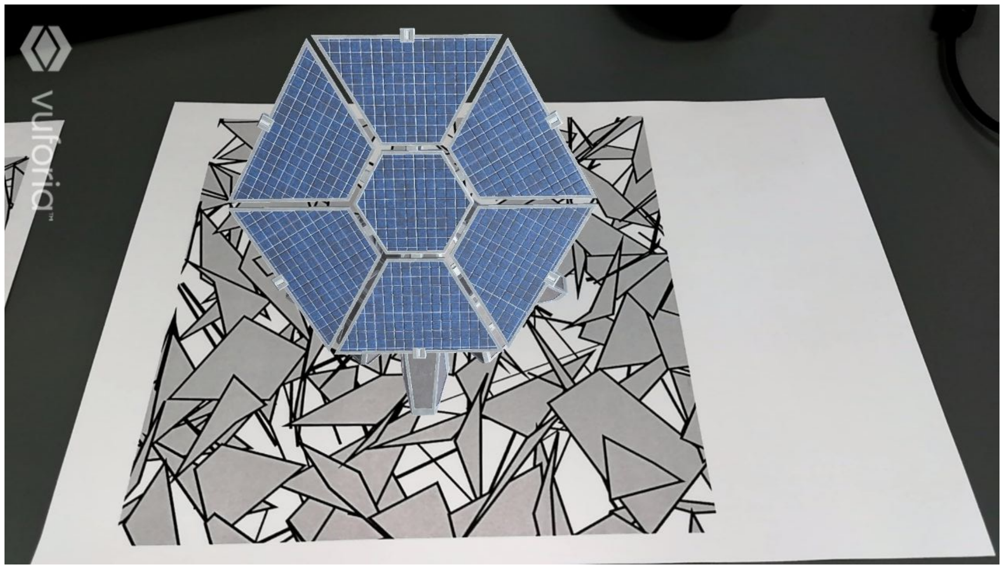
Type: Predefined

Database: VuforiaMars_Images

Image Target: MarsBox.Top

Unity: create new target manager

- Add the object you want to spawn when the target is visible, as child of the Image Target.
- The content will be automatically displayed, and it will move and rotate according to the real-world marker position and rotation.

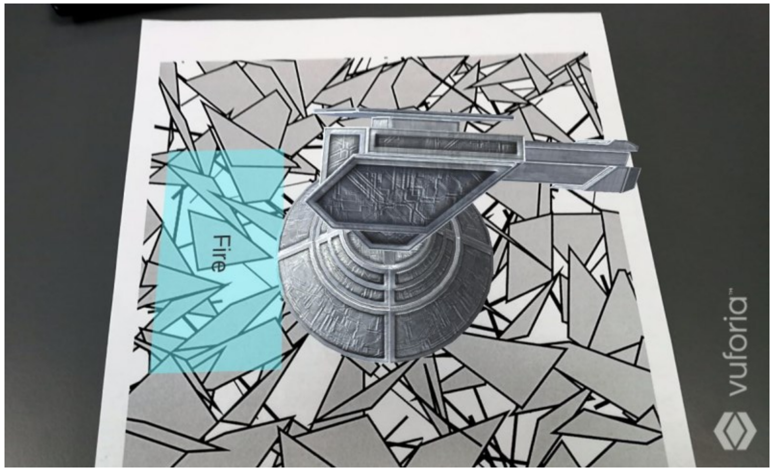


Unity: add Virtual Buttons

- You can interact with your target, one way to achieve it, it is to add a VirtualButton
  - See 3-VirtualButtons scene

- Select your ImageTargetBehaviour -> Advanced -> Add Virtual Button

- This will create a new child GameObject called "VirtualButton".
  - Select it, and remove the "TurnOffBehaviour" script, if present.



Unity: buttons events

```csharp
[SerializeField] private VirtualButtonBehaviour vbb;
private void Awake()
{
    vbb.RegisterOnButtonPressed(OnButtonPressed);
    vbb.RegisterOnButtonReleased(OnButtonReleased);
}

public void OnButtonPressed(VirtualButtonBehaviour vb)
{
    Debug.Log("Button pressed: " + vb.VirtualButtonName);
}

public void OnButtonReleased(VirtualButtonBehaviour vb)
{
    Debug.Log("Button released: " + vb.VirtualButtonName);
}
```

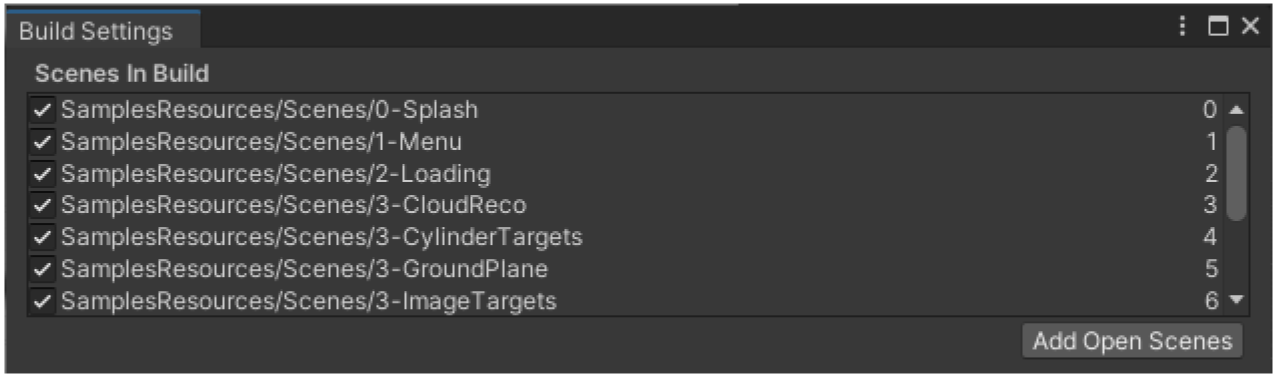
Unity: touch events

```
void Update ()
{
    if (Input.GetMouseButtonDown (0))
    {
        Ray ray = Camera.main.ScreenPointToRay (Input.mousePosition);
        ShootRay(ray);
    }
}

void ShootRay(Ray ray)
{
    RaycastHit rhit;

    bool objectHit = false;
    GameObject gObjectHit = null;

    if (Physics.Raycast(ray, out rhit, 1000.0f))
    {
        objectHit = true;
        gObjectHit = rhit.collider.gameObject;
        // Do whatever you want with the detected game object
    }
}
```

Unity: add Virtual Buttons

- In Assets/SampleResources/Scenes, you can find a few scene that you can test on your smartphone.

- I suggest to build for Android/iOS and test the whole application.
- Verify in “Build Settings” that the first scene is 0 – Splash. It will load the Main menu, from which you will be able to test all the scenes available.



Scenes In Build

✓	SamplesResources/Scenes/0-Splash	0
✓	SamplesResources/Scenes/1-Menu	1
✓	SamplesResources/Scenes/2-Loading	2
✓	SamplesResources/Scenes/3-CloudReco	3
✓	SamplesResources/Scenes/3-CylinderTargets	4
✓	SamplesResources/Scenes/3-GroundPlane	5
✓	SamplesResources/Scenes/3-ImageTargets	6

- If you want, you can test them also in the editor, using the webcam.
  - In this case, complex features, like GroundPlane will not be available.

Suggested activity

- If you want to practice with the basic functionality of the Vuforia, I suggest to:
  - Create an account on https://developer.vuforia.com/
  - Generate two markers using https://www.brosvision.com/ar-marker-generator/ and print them. Uncheck the “Color” option, to avoid unnecessary ink. Gray-scale images are fine.
  - Add the markers in the Vuforia website, create a database and import in in the Unity project.
  - Add two Image Targets with the markers of the database, and see if they work.
  - Try some basic interactions with them. Use virtual buttons and/or touch events.
  - Download a free package like this: https://assetstore.unity.com/packages/3d/environments/sci-fi/robo-s-turret-free-sample-147413
  - Associate the two markers with two turrets (contained in the previous package), and try some basic interactions. For example, touching them or their virtual buttons, they could shoot each other.

- This is just an example; you are free to test what you prefer.