

TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE TLAXIACO

REPORTE DE PRÁCTICA:
RECONOCIMIENTO DE IMÁGENES MÚLTIPLES.

MATERIA:
DESARROLLO CON REALIDAD AUMENTADA

PRESENTA:
Darío Sánchez Linton
Edwin Ortiz Cruz
Diego Alexis Carlos Cruz
Irving Fernando Reyes Pacheco

DOCENTE:
Ing. José Alfredo Román Cruz

CARRERA:
Ingeniera en Sistemas Computacionales
Grupo: 8US

Tlaxiaco, Oax. abril de 2024.



"Educación, ciencia y tecnología, progreso día con día"®

ÍNDICE

INTRODUCCIÓN	4
DESARROLLO	5
Objetivo de la práctica.	5
Descripción.....	5
Material.....	5
Procedimiento.....	5
INSTALACIÓN DE AFRAME.....	5
NPM.....	6
INSTALACIÓN DE THREE.js.....	7
Compilar imágenes de destino.....	9
Paso 3: descargue el archivo	11
Construir la página	11
Preparación.	11
Ejemplo mínimo.	11
Cosas HTML estándar.....	12
Ejecutarlo.	14
Servidor web.	14
Activos 3D.....	15
Agregar activos.....	15
Construye la escena.	16
Objetivos múltiples.	16
Código fuente:.....	17
Pistas múltiples	19
CONCLUSIÓN.....	22

ÍNDICE DE FIGURAS

Figura 1 Seguimiento de imágenes AFrame.....	5
Figura 2 Seguimiento facial AFrame.....	6
Figura 3 Instalación NPM MindAR	6
Figura 4 Instalación NPM AFrame	6
Figura 5 Seguimiento de imágenes NPM	6
Figura 6 Seguimiento facial NPM	6
Figura 7 Seguimiento de imágenes THREE.js	7
Figura 8 Aplicación de seguimiento de imágenes THREE.js	7
Figura 9 Seguimiento facial THREE.js	8
Figura 10 Aplicación de seguimeinto facial THREE.js.....	8
Figura 11 Instalación NPM con THREE.js.....	8
Figura 12 Seguimiento de imágenes NPM con THREE.js.....	8
Figura 13 Seguimiento facial NPM con THREE.js	9
Figura 14 SElección de imágenes compilador MindAR	9
Figura 15 Imagen de demostración.....	10
Figura 16 Funciones de visualización.	10
Figura 17 Carpeta del proyecto.....	11
Figura 18 Archivos de la carpeta del proyecto.....	11
Figura 19 Archivo index.html	12
Figura 20 Ejecución del proyecto.	14
Figura 21 Imagen de ejecución.	15
Figura 22 Animación RA en ejecución objetivos múltiples alternados.	19
Figura 23 Animación con objetivos múltiples simultáneos.....	21

INTRODUCCIÓN

El siguiente documento explica de manera secuencial el desarrollo de realidad aumentada utilizando múltiples objetivos y pistas. Inicialmente se hace mención de las tecnologías con las cuales se puede realizar dicha actividad, los materiales necesarios, se ofrece una breve explicación y se complementa con imágenes que puedan brindar una comprensión más profunda dentro de las limitaciones del proceso de desarrollo. Posteriormente se despliega el código del que se hace uso según se va avanzando en el desarrollo para finalmente obtener la reproducción de animaciones simultáneas al ejecutar el proyecto. Es importante recalcar que como buen investigador, estudiante o profesional; cualquier concepto que se aborde en el presente documento y que sea desconocido o poco familiar para el lector queda a su consideración indagar e informarse para aclarar las dudas y tener una comprensión más exacta.

DESARROLLO

Objetivo de la práctica.

Compilar múltiples imágenes y mostrar diferentes efectos individuales de realidad aumentada mediante el uso de MindAR.

Descripción.

MindAR es una biblioteca web de realidad aumentada de código abierto. Admite seguimiento de imágenes y seguimiento facial.

El proyecto MindAR se puede ejecutar directamente en un archivo HTML estático simple.

MindAR viene con diferentes tipos de capacidades de seguimiento, incluido el seguimiento de imágenes y el seguimiento facial. Para minimizar el tamaño de la biblioteca, cada una de ellas se construye de forma independiente. Además, MindAR proporciona soporte nativo para three.js o AFRAME.

Material.

Para realizar la práctica se deberá contar con los siguientes materiales:

- Computadora con cámara
- Teléfono móvil con cámara
- Conexión a Internet
- Visual Studio Code
- Node.js para instalación de repositorios vía npm
- Navegador web

Procedimiento.

Generalmente hay dos formas de instalar la biblioteca, ya sea a través HTML script o npm.

INSTALACIÓN DE AFRAME.

Seguimiento de imágenes:

```
<script src="https://aframe.io/releases/1.5.0/aframe.min.js"></script>  
<script src="https://cdn.jsdelivr.net/npm/mind-ar@1.2.5/dist/mindar-image-aframe.prod.js"></script>
```

Figura 1 Seguimiento de imágenes AFrame

Seguimiento facial:

```
<script src="https://aframe.io/releases/1.5.0/aframe.min.js"></script>  
<script src="https://cdn.jsdelivr.net/npm/mind-ar@1.2.5/dist/mindar-face-aframe.prod.js"></script>
```

Figura 2 Seguimiento facial AFrame

NPM

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
PS C:\Users\Fox\Desktop\Desarrollo con Realidad Aumentada\UNIDAD 3\RAA> npm i mind-ar --save
```

Figura 3 Instalación NPM MindAR

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
PS C:\Users\Fox\Desktop\Desarrollo con Realidad Aumentada\UNIDAD 3\RAA> npm i aframe --save
```

Figura 4 Instalación NPM AFrame

Seguimiento de imágenes:

```
import 'aframe';  
import 'mind-ar/dist/mindar-image-aframe.prod.js';
```

Figura 5 Seguimiento de imágenes NPM

Seguimiento facial:

```
import 'aframe';  
import 'mind-ar/dist/mindar-face-aframe.prod.js';
```

Figura 6 Seguimiento facial NPM

INSTALACIÓN DE THREE.js

Para alinearse con la instalación oficial de ThreeJS del uso del módulo ES e importmaps desde v137, la versión MindAR threeJS también sigue un patrón similar.

Desde MindAR v1.2.0, ThreeJS se convierte en una dependencia externa, por lo que puede elegir su propia versión de ThreeJS, pero la versión mínima admitida es la v137.

Seguimiento de imágenes:

```
<script type="importmap">
{
  "imports": {
    "three": "https://unpkg.com/three@0.160.0/build/three.module.js",
    "three/addons/": "https://unpkg.com/three@0.160.0/examples/jsm/",
    "mindar-image-three": "https://cdn.jsdelivr.net/npm/mind-ar@1.2.5/dist/mindar-image-three.prod.js"
  }
}
</script>
```

Figura 7 Seguimiento de imágenes THREE.js

Y luego en su aplicación:

```
<script type="module">
import * as THREE from 'three';
import { MindARThree } from 'mindar-image-three';
</script>
```

Figura 8 Aplicación de seguimiento de imágenes THREE.js

Seguimiento facial:

```
<script type="importmap">
{
  "imports": {
    "three": "https://unpkg.com/three@0.160.0/build/three.module.js",
    "three/addons/": "https://unpkg.com/three@0.160.0/examples/jsm/",
    "mindar-face-three": "https://cdn.jsdelivr.net/npm/mind-ar@1.2.5/dist/mindar-face-three.prod.js"
  }
}
</script>
```

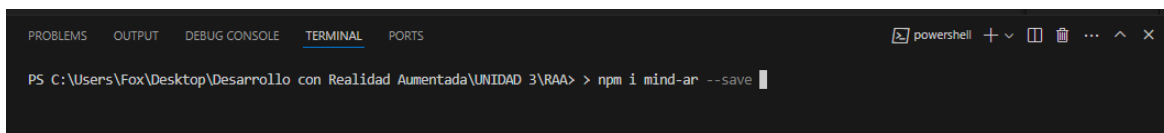
Figura 9 Seguimiento facial THREE.js

Y luego en su aplicación:

```
<script type="module">
import * as THREE from 'three';
import { MindARThree } from 'mindar-face-three';
</script>
```

Figura 10 Aplicación de seguimeinto facial THREE.js

NPM



```
PS C:\Users\Fox\Desktop\Desarrollo con Realidad Aumentada\UNIDAD 3\RAA> npm i mind-ar --save
```

Figura 11 Instalación NPM con THREE.js

Seguimiento de imágenes:

```
import {MindARThree} from 'mind-ar/dist/mindar-image-three.prod.js';|
```

Figura 12 Seguimiento de imágenes NPM con THREE.js

Seguimiento facial:

```
import {MindARThree} from 'mind-ar/dist/mindar-face-three.prod.js';
```

Figura 13 Seguimiento facial NPM con THREE.js

Compilar imágenes de destino.

Antes de trabajar en la página web, primero debemos preprocesar (también conocido como compilar) las imágenes. Necesitamos escanear las imágenes y extraer ubicaciones interesantes (también conocidas como puntos característicos) para poder detectar y rastrear las imágenes más tarde.

Este paso de preprocesamiento lleva tiempo, por lo tanto, queremos hacerlo de antemano para reducir el tiempo de carga cuando los usuarios realmente usen su aplicación AR más adelante.

MindAR viene con una herramienta de compilación súper amigable para hacer esto. Compilador de objetivos de imagen.

Paso 1. Selección de imágenes. Nos dirigimos al compilador y veremos lo siguiente:

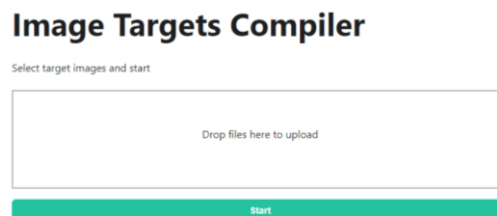


Figura 14 Selección de imágenes compilador MindAR

En esta demostración de inicio rápido, usaremos esta imagen.



Figura 15 Imagen de demostración.

La arrastramos y saltamos en el compilador y hacemos click en **Start**.

Paso 2. Visualizar las características.

Una vez realizada la compilación, veremos algunas funciones de visualización.

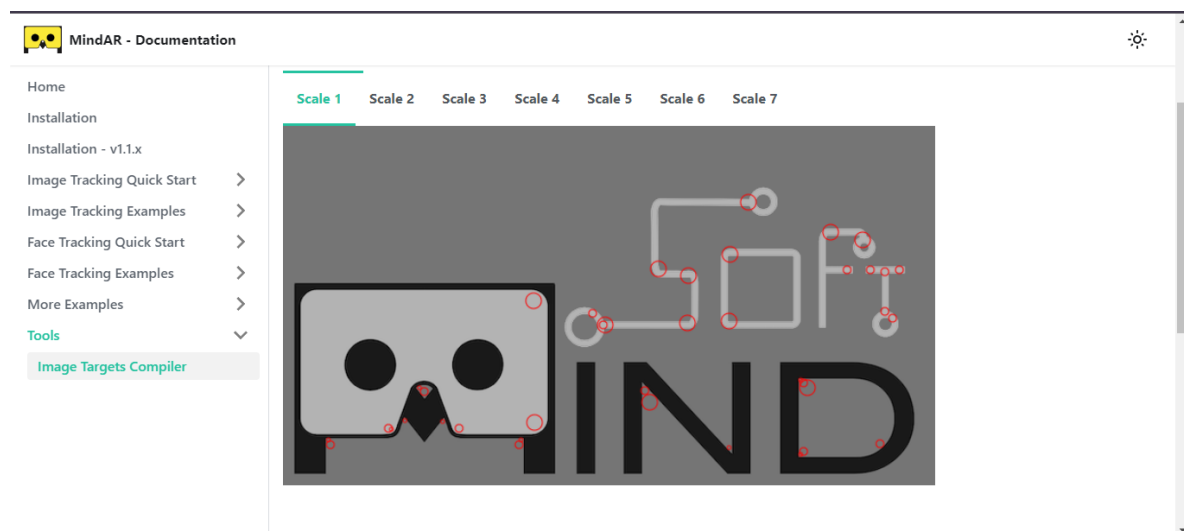


Figura 16 Funciones de visualización.

Esta herramienta de visualización le da una idea de la distribución de características de sus imágenes. En general, desea tener una buena cantidad de funciones con una buena distribución.

Paso 3: descargue el archivo

En la parte inferior de la visualización, verá un botón **Download**. Esto le dará un archivo **targets.mind**. Almacena los datos de las funciones en formato compacto y los necesitaremos más adelante al crear la página web.

Construir la página

Ahora que tiene el archivo **targets.mind** listo, podemos comenzar a construir la página web.

Preparación.

Primero, creamos una carpeta limpia para su proyecto, para este caso la llamaremos **RealidadAR**. Coloque el archivo **targets.mind** allí y creamos un archivo html en blanco, por ejemplo **index.html**. Entonces la carpeta debería tener dos archivos como este:

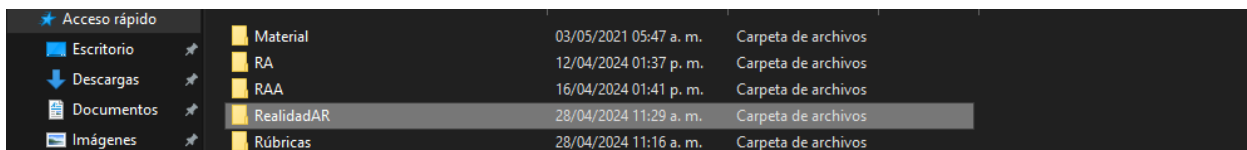


Figura 17 Carpeta del proyecto.

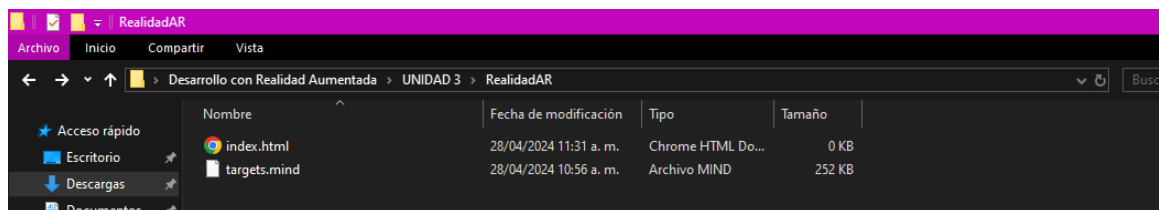


Figura 18 Archivos de la carpeta del proyecto.

Ejemplo mínimo.

Ahora, comencemos con algo simple: mostrar un plano rectangular justo encima de la imagen de destino. Abra **index.html** con el editor de su elección y pegue el siguiente contenido:

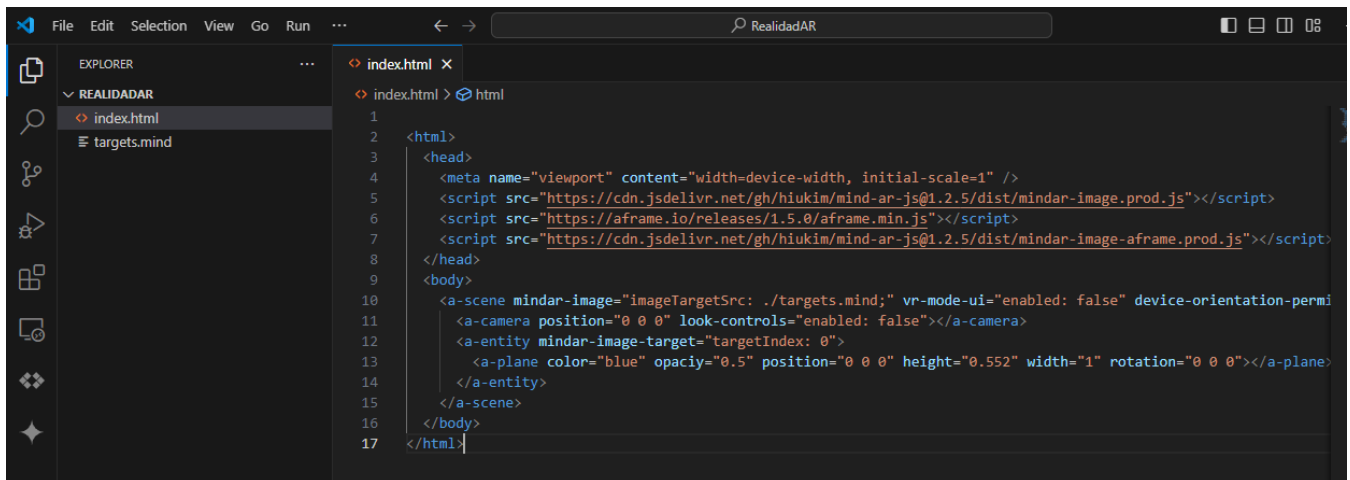


Figura 19 Archivo index.html

Cosas HTML estándar.

html, heady son solo elementos HTML estándar metay bodylos omitiremos.

Biblioteca **mind-ar-js** y **aframe**.

```
<script      src="https://cdn.jsdelivr.net/gh/hiukim/mind-ar-
js@1.2.5/dist/mindar-image.prod.js"></script>
```

```
<script
src="https://aframe.io/releases/1.5.0/aframe.min.js"></script
>
```

```
<script      src="https://cdn.jsdelivr.net/gh/donmccurdy/aframe-
extras@v7.0.0/dist/aframe-extras.min.js"></script>
```

```
<script      src="https://cdn.jsdelivr.net/gh/hiukim/mind-ar-
js@1.2.5/dist/mindar-image.aframe.js"></script>
```

La biblioteca mindar aframe alojada en cdn son las únicas cosas que necesitas para crear una aplicación web AR.

MindAR viene con una extensión AFRAME que te permite construir una escena 3D fácilmente. No entraremos en detalles de AFRAME en este documento.

En resumen, puedes ver el <a-scene>bloque por dentro body. Esta es la parte principal de la aplicación. Si no tienes experiencia en AFRAME, no te preocupes. La mayoría de las veces puedes copiar este bloque de código como plantilla para comenzar.

Destacaremos aquí dos cosas relacionadas con MindAR.

1. Dentro de **<a-scene>** se puede ver una propiedad **mindar-image="imageTargetSrc: ./targets.mind;** "que le dice al motor dónde está el archivo **mind** compilado que se creó anteriormente.
2. Hay un **<a-entity>**, con una propiedad **mindar-image-target="targetIndex: 0"**. Esto le indica al motor que detecte y rastree un objetivo de imagen en particular. Siempre **targetIndexes 0**, si **targets.mind** contiene solo una imagen. Sin embargo, puedes compilar varias imágenes juntas y la sentencia **targetIndex** seguirá el orden de las imágenes. Hablaremos más de esto más adelante cuando tengamos múltiples objetivos de imagen.

El motor AR consume la transmisión de su cámara, luego detecta, rastrea las imágenes objetivo y actualiza la visibilidad y las posiciones de esta entidad. Significa que todo lo que esté adjunto a la entidad se mostrará mágicamente en consecuencia. Una vez que tenga esta configuración correctamente, lo que normalmente debe hacer es construir el contenido dentro de ella **a-entity** de acuerdo con las necesidades de su aplicación.

Este es un caso mínimo, se verá un archivo **<a-plane color="blue" opacity="0.5" position="0 0 0" height="0.552" width="1" rotation="0 0 0"></a-plane>**. Este es el objeto que queremos mostrar encima de la imagen de destino. Obviamente, es sólo un plano azul.

Como se puede ver, configuramos el ancho en 1, para que en realidad tenga el mismo ancho que la imagen de destino. ¿Por qué fijamos la altura en 0,552? Porque la imagen de destino tiene una proporción de 0,552/1. Si se establece la altura en 1, significa que también es igual al ancho de la imagen de destino. Resultará ser un cuadrado. Ahora este plano rectangular se superpondrá perfectamente a la imagen de destino.

Además, se debe tener en cuenta que el punto de anclaje de la entidad es el centro de la imagen de destino.

Ejecutarlo.

Servidor web.

Aunque es una página HTML simple, probablemente no pueda ejecutarla simplemente abriendo el archivo en el navegador. El motivo es que la página requiere acceso a la cámara.

Creo que hay muchas soluciones posibles a esos problemas, como configurar la política del navegador o algo así. Una forma de solucionar este problema es configurar un servidor localhost que pueda servir una página web.

Para este caso utilizaremos el editor Visual Studio Code y decargaremos una extensión Live Server para ejecutar localmente el servidor y a ver las pruebas.

Y para lanzar el proyecto se presiona el botón **Go Live** ubicado en la esquina inferior izquierda del editor.

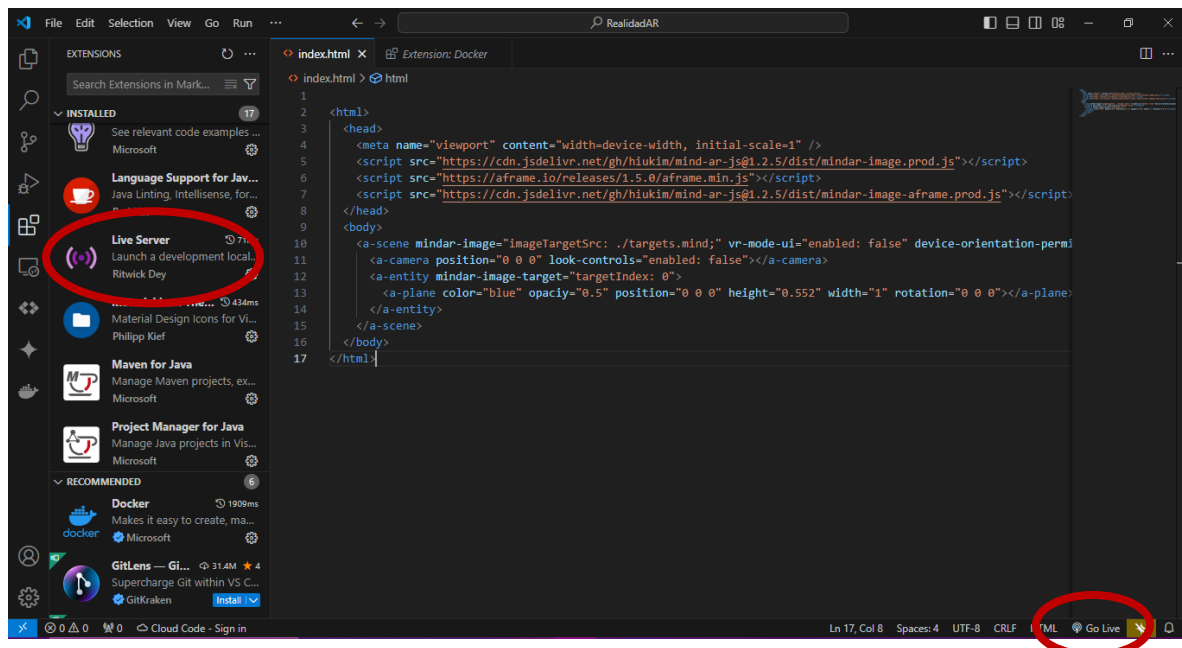


Figura 20 Ejecución del proyecto.

Si puede iniciar la página con éxito, la cámara se iniciará. Después de apuntar al objetivo de la imagen, verá un rectángulo azul pegado en la parte superior.

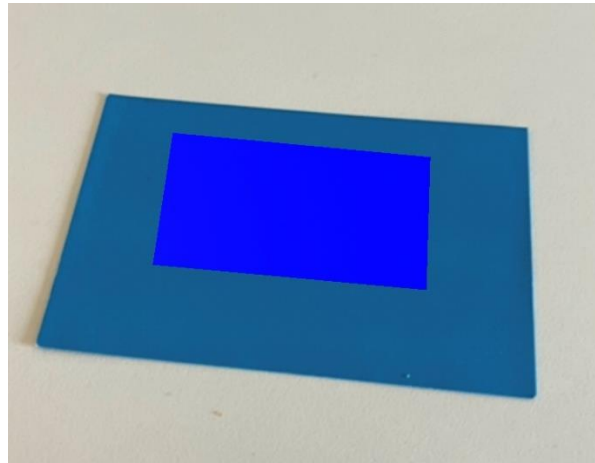


Figura 21 Imagen de ejecución.

Activos 3D

Agregar activos.

Lo primero que debemos hacer es agregar algunos activos a la escena. En AFRAME, hacemos esto por **a-assets**. Agrega este bloque de código dentro del **<a-scene/>** elemento.

```
<a-assets>

  <a-asset-item                             id="avatarModel"
src="https://cdn.jsdelivr.net/gh/hiukim/mind-ar-
js@1.2.5/examples/image-tracking/assets/card-
example/softmind/scene.glTF"></a-asset-item>

</a-assets>
```

La primera es en realidad nuestra imagen de destino. El segundo es un modelo en gltf formato 3D. AFRAME básicamente admite todos los formatos 3D estándar, por lo que probablemente puedas reemplazarlo con los modelos que elijas más adelante.

Construye la escena.

Ahora podemos reemplazar el plano rectangular opaco del ejemplo anterior con un recurso de imagen.

```
<a-plane src="#card" position="0 0 0" height="0.552" width="1"
rotation="0 0 0"></a-plane>
```

Además, agregaremos un modelo 3D animado encima de la imagen.

```
<a-gltf-model rotation="0 0 0" position="0 0 0.1" scale="0.005
0.005 0.005" src="#avatarModel" animation="property: position;
to: 0 0.1 0.1; dur: 1000; easing: easeInOutQuad; loop: true;
dir: alternate">
```

La escala del modelo 3D que utilizamos aquí se normalizó a -1 a 1, por lo que establecimos una escala apropiada de 0,005. También tenemos una animación para hacer que el modelo oscile entre 0 y 0,1 en el eje z. No entraremos en detalles de la animación, pero son AFRAME cosas estándar.

Objetivos múltiples.

MindAR le permite compilar múltiples imágenes de destino y mostrar diferentes efectos AR individualmente, como esta demostración:

Puede utilizar las siguientes imágenes de destino para realizar pruebas:



Código fuente:

```
<html>

  <head>

    <meta      name="viewport"      content="width=device-width,
initial-scale=1" />

    <script
src="https://aframe.io/releases/1.5.0/aframe.min.js"></script
>

    <script
src="https://cdn.jsdelivr.net/gh/donmccurdy/aframe-
extras@v7.0.0/dist/aframe-extras.min.js"></script>

    <script      src="https://cdn.jsdelivr.net/npm/mind-
ar@1.2.5/dist/mindar-image-aframe.prod.js"></script>

  </head>

  <body>

    <a-scene      mindar-image="imageTargetSrc:
https://cdn.jsdelivr.net/gh/hiukim/mind-ar-
js@1.2.5/examples/image-tracking/assets/band-
example/band.mind;"      color-space="sRGB"
renderer="colorManagement: true, physicallyCorrectLights" vr-
mode-ui="enabled: false"      device-orientation-permission-
ui="enabled: false">

      <a-assets>

        <a-asset-item      id="bearModel"
src="https://cdn.jsdelivr.net/gh/hiukim/mind-ar-
js@1.2.5/examples/image-tracking/assets/band-
example/bear/scene.glTF"></a-asset-item>

        <a-asset-item      id="raccoonModel"
src="https://cdn.jsdelivr.net/gh/hiukim/mind-ar-
js@1.2.5/examples/image-tracking/assets/band-
example/raccoon/scene.glTF"></a-asset-item>

      </a-assets>
```

```
<a-camera position="0 0 0" look-controls="enabled:
false"></a-camera>
```

```
<a-entity mindar-image-target="targetIndex: 0">
```

```
<a-gltf-model rotation="0 0 0 " position="0 -0.25 0"
scale="0.05 0.05 0.05" src="#raccoonModel" animation-mixer>
```

```
</a-entity>
```

```
<a-entity mindar-image-target="targetIndex: 1">
```

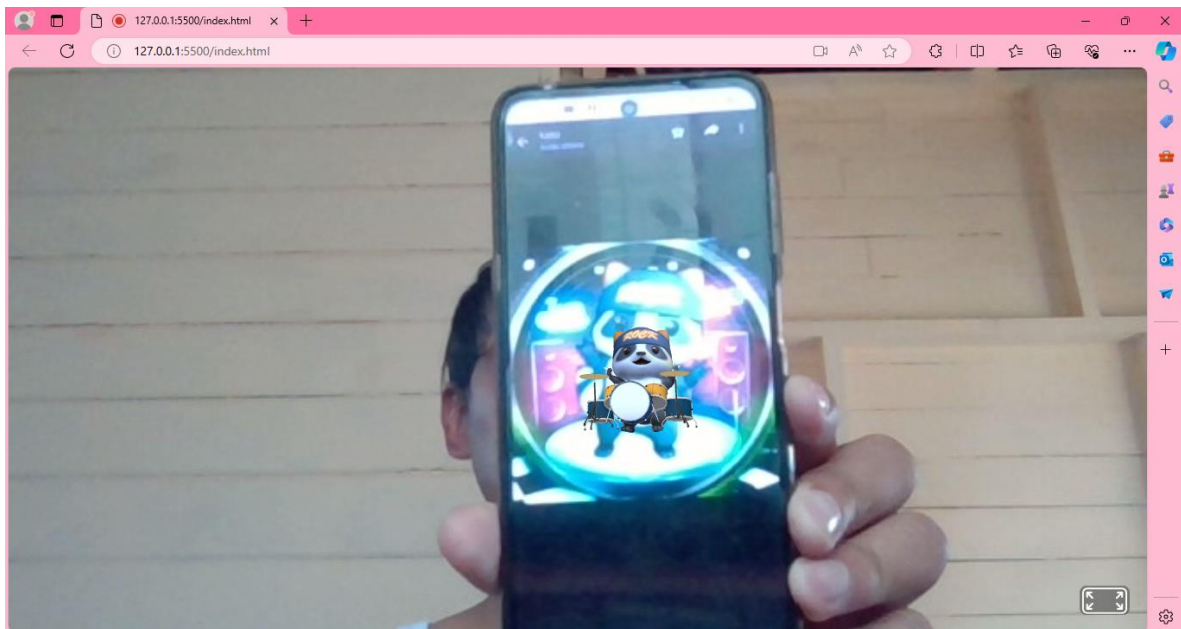
```
<a-gltf-model rotation="0 0 0 " position="0 -0.25 0"
scale="0.05 0.05 0.05" src="#bearModel" animation-mixer>
```

```
</a-entity>
```

```
</a-scene>
```

```
</body>
```

```
</html>
```



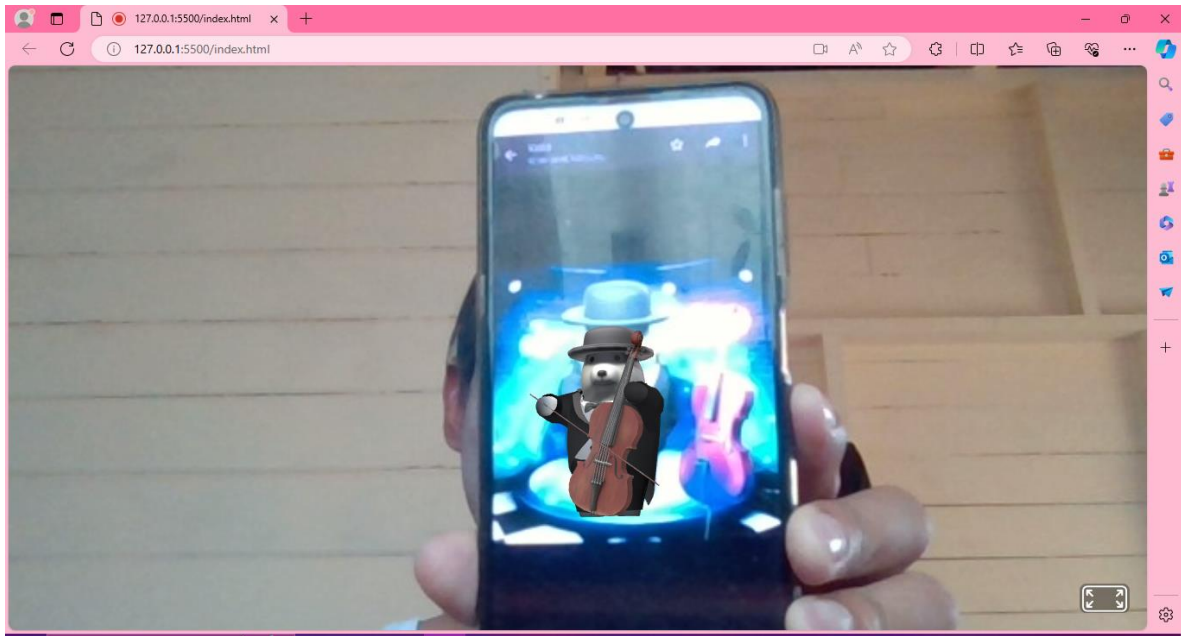


Figura 22 Animación RA en ejecución objetivos múltiples alternados.

Pistas múltiples

A diferencia del ejemplo anterior, MindAR te permite rastrear múltiples objetivos simultáneamente.

Código fuente:

```
<html>

  <head>

    <meta      name="viewport"      content="width=device-width,
initial-scale=1" />

    <script
src="https://aframe.io/releases/1.5.0/aframe.min.js"></script
>

    <script
src="https://cdn.jsdelivr.net/gh/donmccurdy/aframe-
extras@v7.0.0/dist/aframe-extras.min.js"></script>
```

```
<script src="https://cdn.jsdelivr.net/npm/mind-ar@1.2.5/dist/mindar-image-aframe.prod.js"></script>
```

```
</head>
```

```
<body>
```

```
<a-scene mindar-image="imageTargetSrc: https://cdn.jsdelivr.net/gh/hiukim/mind-ar-js@1.2.5/examples/image-tracking/assets/band-example/band.mind; maxTrack: 2" color-space="sRGB" renderer="colorManagement: true, physicallyCorrectLights" vr-mode-ui="enabled: false" device-orientation-permission-ui="enabled: false">
```

```
<a-assets>
```

```
<a-asset-item id="bearModel" src="https://cdn.jsdelivr.net/gh/hiukim/mind-ar-js@1.2.5/examples/image-tracking/assets/band-example/bear/scene.glTF"></a-asset-item>
```

```
<a-asset-item id="raccoonModel" src="https://cdn.jsdelivr.net/gh/hiukim/mind-ar-js@1.2.5/examples/image-tracking/assets/band-example/raccoon/scene.glTF"></a-asset-item>
```

```
</a-assets>
```

```
<a-camera position="0 0 0" look-controls="enabled: false"></a-camera>
```

```
<a-entity mindar-image-target="targetIndex: 0">
```

```
<a-gltf-model rotation="0 0 0" position="0 -0.25 0" scale="0.05 0.05 0.05" src="#raccoonModel" animation-mixer>
```

```
</a-entity>
```

```
<a-entity mindar-image-target="targetIndex: 1">
```

```
<a-gltf-model rotation="0 0 0" position="0 -0.25 0" scale="0.05 0.05 0.05" src="#bearModel" animation-mixer>
```

```
</a-entity>

</a-scene>

</body>

</html>
```

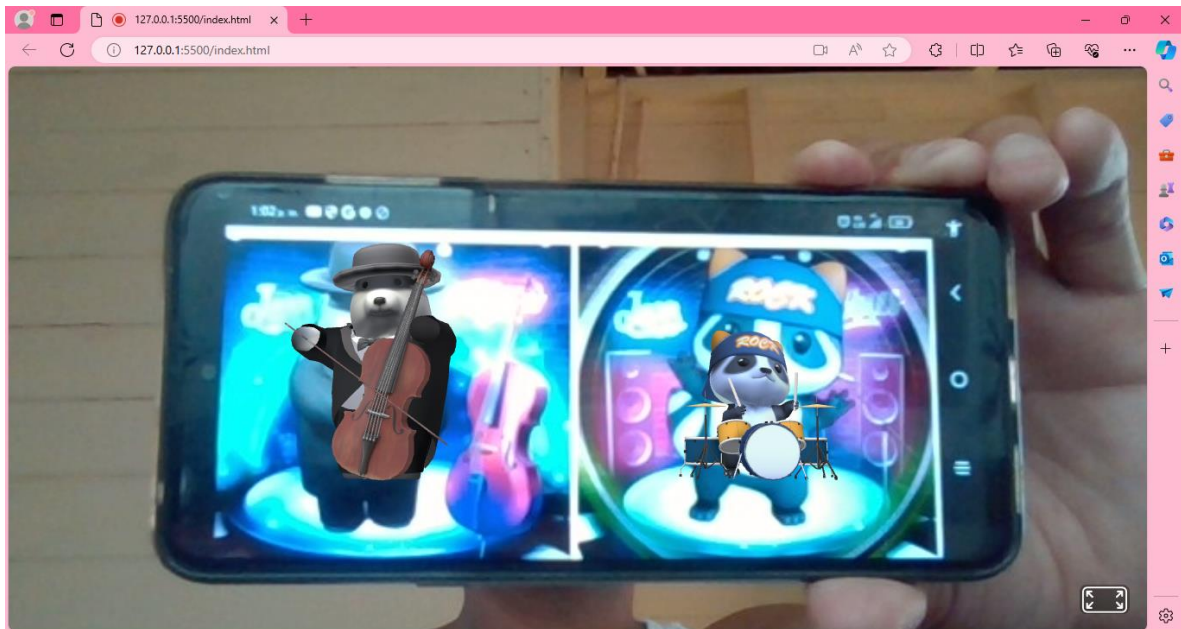


Figura 23 Animación con objetivos múltiples simultáneos.

Esto es exactamente igual que el ejemplo anterior, con una sola diferencia. Hay un **maxTrack: 2**; conjunto de parámetros dentro de **a-scene**.

CONCLUSIÓN.

La realidad aumentada ofrece la posibilidad de focalizar múltiples objetivos ya sea de manera alternada o simultánea, mediante la implementación de la tecnología AFRAME compatible con JavaScript, ThreeJS que es una tecnología también enfocada a la realidad virtual y algunas nociones de HTML, y cuya implementación es sencilla siguiendo la documentación que proporciona la plataforma MindAR. Esta práctica nos mostró un proceso sencillo para poder reproducir animaciones de realidad aumentada con el uso de las herramientas ya mencionadas y nos proporcionó un panorama introductorio que nos permitió conocer el potencial del desarrollo con estas tecnologías. Finalmente pudimos constatar que, con las herramientas y materiales necesarios, el desarrollo de realidad aumentada es cada vez más eficiente si se cuenta con el conocimiento preciso, incluso sea básico de los lenguajes requeridos y su eventual uso y aplicación en diversos sectores del mundo moderno.