UNIVERSIDAD DE GRANADA

Computación Ubicua e Inteligencia Ambiental

ARPI: Realidad aumentada directa aplicada a la enseñanza de instrumentos de tecla

---

Leire Requena Garcia, Clara María Romero Lara

28 de junio de 2022

Índice general

Parte 1:
Primer acercamiento a ARPI

1.1: Descripción

ARPI (Augmented Reality PIano) es un programa de ordenador basado en realidad aumentada de percepción directa. Mediante un proyector y una webcam, y dada una partitura en formato digital, ARPI proyecta sobre el piano un indicador de las teclas que corresponde tocar.

1.2: Motivación

En este proyecto, plantearemos ARPI como una herramienta de apoyo al aprendizaje del piano. Su uso está enfocado a pianistas noveles, con ciertas pero limitadas nociones de solfeo.

1.2.1: Planteamiento del problema

Problema

El piano es un instrumento en el cual es difícil iniciarse incluso si conoces solfeo y un instrumento previo: el trabajo paralelo independiente de las dos manos es un aspecto no se encuentra en casi ningún instrumento, y la lectura en acordes y dos claves se puede antojar muy difícil a músicos de otras disciplinas.

Solución ofrecida

A modo de apoyo, ARPI proyecta sobre el teclado las teclas a tocar, permitiendo al pianista centrarse en más aspectos de la partitura (tempo, matiz, pedal, indicaciones del profesor…) sin sacrificar tanto tiempo en la lectura y colocación de los dedos.

Así, el estudio y ensayo de las piezas será más ameno y se progresará más rápido, ayudando a crear la memoria muscular necesaria hasta que el músico se sienta cómodo con la pieza para tocarla sin la ayuda de ARPI.

1.2.2: Competencia

En percepción directa, nuestra competencia es prácticamente nula. Los proyectos de percepción directa suelen ser costosos y difíciles de implementar, pero creemos que podemos ofrecer esta herramienta a un precio asequible. Pensamos que las ventajas ante nuestros competidores superan a los inconvenientes.

Nuestros competidores se ubican en el campo de la realidad virtual o de realidad aumentada indirecta, pero creemos que un accesorio aparatoso como las gafas VR los aleja demasiado del modelo educativo junto al profesor que nos gustaría alentar.

1.2.3: Plazo de amortización

Modelo de negocio: licencia + hardware/estructura

- Licencia adaptada a las necesidades del cliente, de pago único.
- Hardware. Su precio de venta al público se calculará tras tener en cuenta los siguientes ítems:
  - Cámara (lote 100 webcam al por mayor: 56.7$/ud)
  - Proyector (lote 100 miniproyector al por mayor: 24.70$/ud)
  - Soporte: su precio incluirá diseño e impresión 3D
    - Modelado 3D: 24$/hora (único pago una vez se obtenga el modelo)
    - Precio impresión: aprox 5$/hora, depende de material

1.2.4: Opiniones

Expertos

- **Julián Redondo, director de la banda y la asociación musical de San Clemente (Cuenca):**
  - A largo plazo es una herramienta que podría ser funcional.
  - Es especialmente útil para principiantes, ya que aunque la técnica no sea la mejor, hacer sonar un piano es sencillo. También sirve en el caso de la gente que no se termina de acostumbrar a leer partituras, ya que en el piano es especialmente difícil porque cada mano trabaja en una clave distinta.
  - Aún con todo, se debería tener un profesor cerca para corregir fallos técnicos como la posición de las manos.
- **Elena, pianista:**
  - Es útil para gente empezando con el instrumento.
  - Para avanzar a largo plazo está un poco más limitado, porque aunque te diga que notas tocar se debería aprender a leer música porque empezarán a complicarse las canciones con combinaciones de teclas más raras y varias voces a la vez
  - Para melodías sencillas funcionaría muy bien.
- **José Mateos Jiménez, profesor de piano y jefe de estudios adjunto en el conservatorio Pablo Sorozábal (Puertollano, Ciudad Real):**
  - Interesante para principiantes, gente con muy poca noción del instrumento o de música.
  - Es una opción mucho más interesante que otros planteamientos anteriores, basados en pianos virtuales (como en aplicaciones móviles) o en gafas de realidad virtual. De esta forma es mucho más cómodo para alumno y profesor.
- **Jame Day, pianista y pedagogo musical con enfoque TIC. Autor de 'Apps para músicos':**
  - El aprendizaje del piano puede dividirse en una serie de hitos, los cuales incluyen la identificación de las notas sobre el teclado y su correcta digitación. ARpi tiene potencial para ser una herramienta útil en estos dos aspectos y, además, es única en su género: no existen otras alternativas que apoyen la enseñanza de estos hitos.
  - Maximizar el acierto "dedo-nota-tecla.[es] natural para un pianista avanzado, pero cuando se trata de principiantes que estudian de manera autónoma, una sesión de estudio puede ser ineficaz o incluso volverse contraproducente si se memorizan e interiorizan errores. Una herramienta que guíe al estudiante en estos pasos aceleraría el proceso de interiorización y reduciría la posibilidad de fallo, haciendo el estudio más eficaz.
  - Es importante tener en cuenta que en las escuelas y academias de música el presupuesto suele ser muy reducido.

Clientes

- **Anuncia Martínez Serrano, alcaldesa de Vara de Rey (Cuenca):**
  - Si llegara como una proposición oficial se intentaría poner en práctica a través de la banda de música municipal.
  - Lo considera una buena idea para fomentar el estudio de la música en niños pequeños y adolescentes y podría expandirse también a personas mayores.
  - El ofrecer en conjunto la aplicación, el proyector, soporte, cámara y micro, hace más accesible a una posible compra.

Usuarios

- **Atanasio, 4 años en grado elemental de conservatorio (violín):**
  - Aunque ARPI no sea imprescindible para el aprendizaje del piano, cualquier herramienta que asista en los primeros pasos es bienvenida.
  - Facilita y apoya la labor del profesor y del alumno.
  - En la realidad virtual se pierde la fisicalidad del piano, la realidad aumentada permite mantener el contacto con el teclado, con las propias manos...
- **Miguel Ángel, 8 años tocando el piano y estudiante DGIIM:**
  - Una herramienta muy interesante, sobre todo para la gente que tenga poca formación musical, y tenga que comenzar a colocar las notas sobre el piano.
  - Para los que ya sepan tocar el piano podría ser un complemento artístico. Hay muchos vídeos en internet en el que la gente usa pianos que tienen luces en las teclas, se podrían conseguir efectos muy chulos.
  - Una de las preocupaciones que tendría al colocar un proyector arriba sería que el cuerpo podría tapar fácilmente la imagen del proyector, pero aún así sería preferible a VR.

1.3: Software y Hardware a utilizar

1.3.1: Software

Usaremos la última versión de Unity, un software privativo que nos ofrece una versión gratis siempre que no moneticemos nuestro programa.

Para las tareas específicas de realidad aumentada tenemos varias opciones de software libre como ARtoolkit, ARcore y OpenCV y otras privativas que ofrecen una versión gratiuta limitada como Vuforia; al empezar el proyecto habremos investigado lo suficiente de cada una de ellas para escoger una.

1.3.2: Hardware

Necesitaremos un proyector, una cámara, un soporte y por supuesto un piano o teclado electrónico.

En este momento del desarrollo, contamos con un teclado electrónico de tres escalas y media, sobre el cual montamos un brazo ajustable en el que se coloca el móvil, que actúa como cámara y micrófono.

Para el proyector, disponemos de un trípode ajustable que se coloca estratégicamente detrás del pianista. En una versión final del producto, el proyector iría montado en un soporte montado encima de las teclas, a una altura suficiente para alcanzar el rango necesario y no dificulte al intérprete su labor.

Se adjunta un prototipo del diseño que podría tener el soporte completo, que incluye el proyector, la cámara y el micrófono.

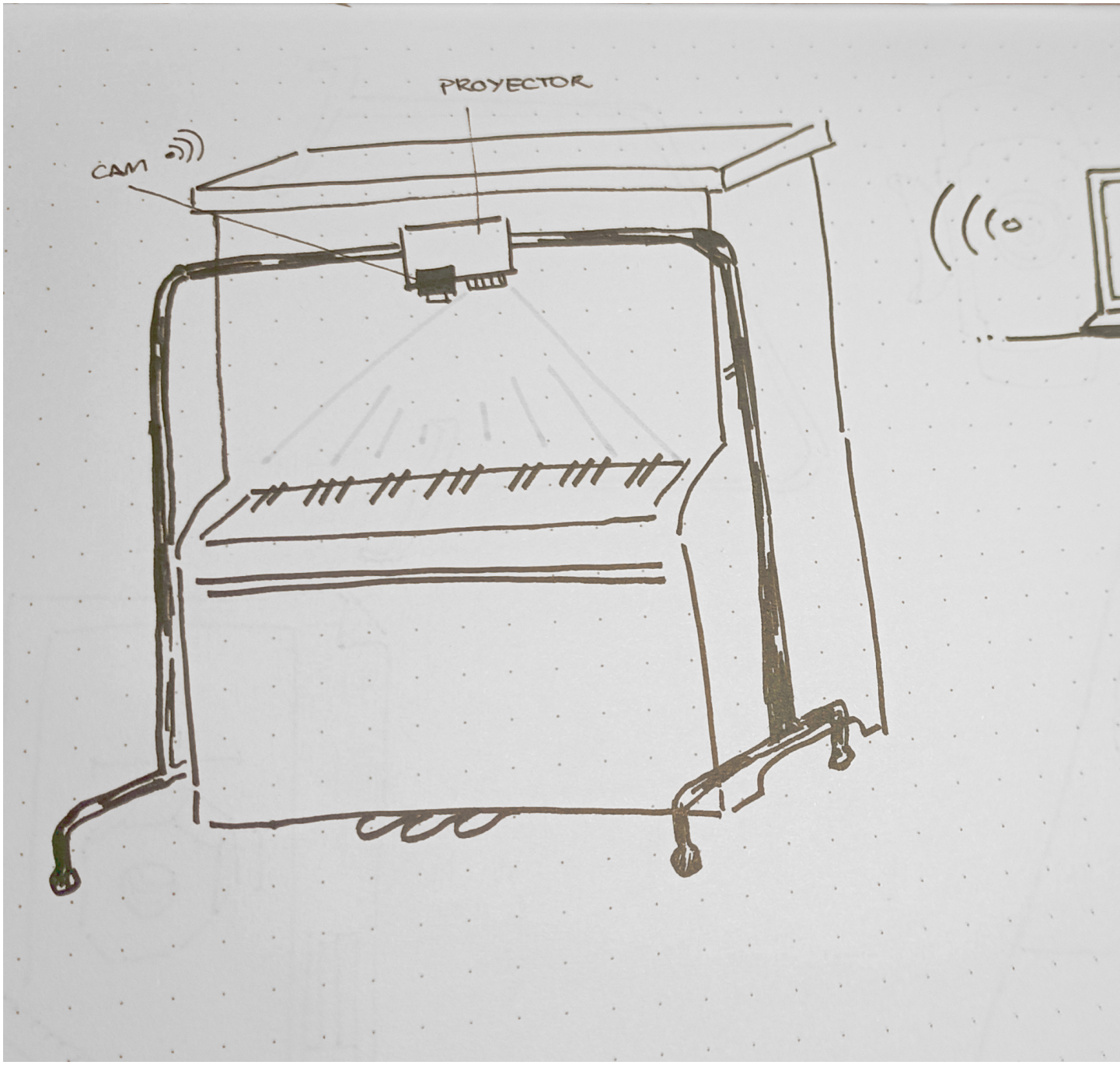

Figura 1.1: Prototipo de la versión final

Parte 2:
Impementación en Unity

2.1: Reconocimiento del piano

2.1.1: Instalación de Vuforia

Como hemos mencionado, la herramienta que decidimos usar para la realidad aumentada en este proyecto es Vuforia, un software de pago que ofrece licencias gratuitas bajo una serie de condiciones:

- 20 model targets
- 20 area targets
- No superar las 100 imágenes en los servicios en la nube

Para poder usar la versión gratis debemos crearnos una cuenta en el portal de Vuforia y crear en el apartado developing una licencia desde *Develop*. Además necesitamos importar la biblioteca a Unity, podemos descargarla desde la página oficial y un asistente nos indicará los pasos en el entorno de desarrollo de Unity. Cuando creemos nuestro primer objeto Vuforia, como `ARCamera`, podremos acceder a su configuración y añadir nuestra licencia.

2.1.2: Algoritmo de reconocimiento

El algoritmo de reconocimiento del teclado actúa de esta forma:

1. Colocar un marcador aproximadamente en el centro del teclado
2. Al ser reconocido se mostrará un modelo 3D de un teclado
3. Calibraremos el teclado para colocarlo encima del nuestro y ajustarlo a su medida
4. Confirmamos que la calibración está hecha y el modelo desaparece de la pantalla

De esta forma no tenemos que gestionar muchos marcadores de reducido tamaño y será más eficiente y fácil de reconocer.

Marcador

El marcador es una imagen que se asocia al objeto `ImageTarget` que nos ofrece Vuforia, le debemos asignar el modelo del piano creando un nodo hijo en unity con el. Sin mayor configuración, al encender el simulador de programa y enfocar con la cámara se nos muestra el modelo justo encima de nuestro marcador.

Cámara

Cuando creamos una nueva escena aparece con dos componentes: una cámara y un foco de luz; para la realidad aumentada el primero de ellos no nos sirve, y hay que sustituirlo por la `ARCamera` que nos ofrece Vuforia.

Para que use nuestra webcam debemos configurar el `PlayMode` en la configuración del paquete.

Además, especificaremos un poco más el comportamiento del marcador y la `ARCamera` creando un script asociado a esta última con dos eventos personalizados de Unity que indican qué hará cuando encuente o pierda el marcador. Este script es solo una modificación del ejemplo que viene el el pack de realidad aumentada de Vuforia para Unity, cuyo único añadido es mostrar un aviso de recalibración.

Ajuste del modelo

Para ajustar el modelo 3D a nuestro piano real, hemos usado el paquete `LeanTouch` en su versión gratuita de la *Unity Asset Store*. Esto nos proporcionará una herramienta para redimensionar y mover el modelo que, aunque esté pensada para pantallas táctiles, incluye también una forma de simular la interacción táctil en el ordenador pulsando la tecla `ctrl izquierdo`.

2.2: Reconocimiento del audio

2.2.1: Captura del micrófono en tiempo real

Para la captura, reproducción y análisis en tiempo real del audio, hemos hecho uso de las clases `Microphone` y `AudioClip` de Unity. Creando una entidad de tipo `AudioSource`, podemos hacer que su contenido sea guardado en un `AudioClip`, que extraemos de un dispositivo `Microphone`.

2.2.2: Procesamiento del audio y notación musical

Para el desarrollo de esta funcionalidad, nos hemos apoyado en la biblioteca `ExtractDataFromCapturedSoundByMicrophone` (Jorge Gutiérrez Segovia, 2021). Cuando extraemos un clip de audio, podemos emplear el método `GetData` de `AudioClip` para obtener información sobre, entre otras cosas, su frecuencia. La información del sonido se reparte en un espectrograma de n elementos con valores del 0 al 1, dependiendo de la amplitud relativa a el tramo de la frecuencia dado.

Con este espectrograma, podemos extraer la frecuencia máxima relativa en un momento dado, y la podemos correlacionar con la frecuencia de una nota musical conocida. La3 tiene una frecuencia de 440Hz, y es el estándar de afinación mundial. Usando los 12 semitonos existentes (siete notas musicales separadas por dos semitonos, menos Mi-Fa y Si-Do, que se separan por uno solo), podemos calcular la diferencia con el La3 y obtener una nota y octava a partir de una frecuencia en herzios. Todo este procesamiento lo lleva a cabo la función `GetSpectrumData` de la biblioteca anteriormente mencionada.

2.3: Implementación de los menús

La aplicación consta de dos escenas:

- Menú principal.
  - Play
  - Salir
- Escena de canción.
  - Calibración 3D
  - Selector de canciones
  - Juego
  - Resultados
  - Salida al menú principal

El cambio entre escenas se realiza mediante la función `LoadScene` de la clase `SceneManager` que nos ofrece Unity. El script se asocia a un objeto que no se destruye entre escenas, ya que tiene la orden `DontDestroyOnLoad` porque en Unity los objetos se destruyen al cambio de escena, si no lo indicamos de forma explícita tendríamos que tener una copia del objeto en cada escena o dejaría de funcionar el programa.

La escena para el menú consta de un `canvas` y varios `botones` (*Play*, y *Salir*). En esta versión del programa solo hay dos escenas, la calibración se realiza justo antes de escoger la canción que se desea tocar.

2.3.1: Escena para tocar

Después de confirmar la calibración, se mostrarán 6 botones, cada uno de ellos con la demo de una canción. Inmediatamente después comenzará la ejecución del juego.

2.4: Implementación de la lógica de juego

La lógica de nuestro programa es, en esencia, una serie de cubos de Unity cuya posición en el eje Y varía descendientemente de forma constante a lo largo de la ejecución; y cuya altura se corresponde con la duración de la nota. Su posición en el eje X dependerá de la calibración previamente realizada con el modelo 3D del piano, del cual obtiene el cubo su ancho.

2.4.1: PianoObject

`PianoObject` es el comportamiento asociado al nodo que contiene el modelo 3D del piano. Está pensada como una clase *singleton*, de la cual sólo puede existir una instancia. Sus métodos más relevantes son los siguientes:

- `GetChildren(int i)`: este método devuelve un objeto del conjunto del modelo del piano; es decir, un `GameObject` que contiene todas las partes que componen una tecla, además del cubo que será la representación gráfica en pantalla de la nota.
- `GetNoteObject(int i)`: devuelve la representación gráfica de la nota.

2.4.2: NoteObject

Esta clase se encarga de manejar las colisiones, que usamos para conocer el momento en el que se debe tocar una nota.

Los Objetos de Unity cuentan con un colisionador por defecto `BoxCollider`, del cual haremos uso para determinar el momento en el que la nota debe de ser tocada, y en tal caso, compararla con la nota esperada. Para implementarlo, hemos indicado que la representación gráfica de las notas es un *Trigger*, y además tenemos un plano con una etiqueta específica de colisionador.

Hemos sobrecargado los métodos asociados al colisionador por *Trigger*:

- `OnTriggerEnter(Collider other)`: método de evento para cuando entra por primera vez en contacto la nota con el plano colisionador. Activa un atributo privado (`can_be_pressed`).
- `OnTriggerStay(Collider other)`: mientras esté en contacto con el plano colisionador, llama a la función que compara la nota reconocida y la nota esperada. En caso de coincidir, añade 1 al contador de notas correctas.
- `OnTriggerExit(Collider other)`: método de evento para cuando la nota abandona el área del colisionador. Desactiva el atributo privado (`can_be_pressed`) y, si no se ha reconocido ninguna nota, añade un error al contador de fallos.

2.4.3: BeatScroller

`BeatScroller` se encarga de la representación gráfica de la canción sobre la pantalla, así como de recorrer la canción. Alguna de la información que contiene es:

- Lista estática con los nombres de las notas existentes
- Lista contenedora de los `NoteObject`
- Lista contenedora de las notas de la canción de forma secuencial

Sus métodos más relevantes son:

- `PlaySong(List<Note> notes)`: recibe la lista de objetos de la clase Nota y añade la representación gráfica correspondiente a la lista de `NoteObject` y los ordena.
- `AddChildren(string pitch)`: busca en el el `PianoObject` el objeto cuyo nombre corresponde al indicado. Crea una copia de la representación gráfia de la nota, la activa y la añade a la lista. En caso de recibir como argumento un silencio, añade una nota de igual manera pero inactiva.

- `OrderNotes()`: modifica la altura en base a la duración asociada a la Nota, y la posición de forma relativa a la posición de la nota anterior.
- `CompareNotes()`: si el juego ha empezado, obtiene la información del micrófono tras procesarla como se explica en la sección 2.22.2. De esta información nos interesa el *pitch*, el cual compara con el correspondiente en la lista de nombres, y devuelve si la nota escuchada se corresponde con la nota esperada, con margen de error de un semitono.

2.4.4: SongSelector

Esta clase se encarga de inicializar la lista de canciones. Obtiene todas las canciones del directorio Music, añade un evento *onClick* desde la lista de botones, el cual crea un objeto `Song`, cuyo funcionamiento describimos a continuación.

Song

Tiene una lista de objetos `Note`. Su constructor recibe la ruta a un documento XML del cual extraemos la información con funcionalidades de Unity para este tipo de archivo. Por cada nodo que encuentra, crea un objeto Note y lo añade a la lista.

Note

Objeto unitario que compone las canciones. Cuenta con pitch y duration.