Documentación GeoKids

Descripción del proyecto

Esta aplicación, por si no se recuerda la propuesta, va dirigida a niños ya que esta es una aplicación la cual hace cuestiones a los niños sobre figuras geométricas en la que consta de varios niveles sobre estas.

La aplicación ha sido desarrollada con Python y OpenCV, ya que este último es muy necesario ya sea por el reconocimiento facial o el mismo reconocimiento de voz.

Estructura del código

Para realizar la aplicación se han realizado varios ficheros, los cuales tienen la siguiente estructura:

Encontramos dos directorios hermanos y cuatro ficheros al comienzo.

En el primer directorio,datos, encontramos donde se guardan en **usuarios_vectores.json** los usuarios registrados al sistema, en donde se registra el nombre del usuario,el vector de puntos de su cara, y las distintas preferencias de ese usuario, además del nivel y las estadísticas de sus preguntas.

Por otra parte **preguntas.json** guarda las preguntas las cuales se van a usar en la aplicación.

En el segundo directorio nos encontramos con cuatro ficheros con tres funcionalidades distintas.

- **detector_marcadores.py:** Se encarga de detectar marcadores ArUco en tiempo real usando OpenCV. También permite estimar su pose 3D y dibujar información sobre ellos en la imagen de la cámara.

- **figura_visual**.**py**: Permite dibujar figuras geométricas tanto en 2D como en 3D (cubo,pirámide,hexagono,...) en la imagen proyectadas sobre el marcador, en base a las preguntas del nivel.

- **reconocedo_cara.py**: Módulo de reconocimiento facial. Identifica al usuario si ya está registrado mediante su vector de puntos faciales. Si no se reconoce, permite registrar un nuevo usuario junto con su progreso y preferencias.

- **reconocedor_voz.py**:Utiliza un sistema de reconocimiento de voz para permitir al usuario responder preguntas habladas. Detecta la respuesta y evalúa su corrección.

Terminando con los directorios tenemos que fuera de ellos tenemos dos ficheros:

- **calibrar_camara.py** Este script realiza la calibración de una cámara utilizando marcadores ArUco para determinar:

    - Matriz de parámetros intrínsecos de la cámara (cameraMatrix)
    - Coeficientes de distorsión (distCoeffs)
    - Error de reproyección

- **camara.py** Archivo que almacena los parámetros intrínsecos y de distorsión de la cámara obtenidos mediante el proceso de calibración.

- **cuia.py** En donde encontramos algunas utilidades que usamos en el resto de códigos, como:

    - Mostrar ventanas emergentes (popup)
    - Proyección de puntos 3D a 2D (proyeccion)
    - Funciones auxiliares de dibujo o manejo de imágenes

- **main.py** Es donde se define y se gestiona el flujo de toda la aplicación:

    - Muestra el menú inicial (iniciar sesión o registrarse)
    - Inicia la cámara y el reconocimiento facial
    - Carga el nivel correspondiente del usuario
    - Gestiona las preguntas, las respuestas, el feedback y el avance de nivel
    - Coordina la interacción entre módulos de cara, voz, marcadores y figuras

Flujo de ejecución

1. Inicialización de la aplicación

- La aplicación arranca ejecutando la función main(). En esta etapa inicial:
- Se invoca inicializar_aplicacion() para configurar la cámara y cargar la base de datos con las preguntas.
- Se crea la ventana principal de visualización llamada GeoKids AR.

2. Menú inicial y autenticación

- Se muestra un menú con opciones para que el usuario pueda iniciar sesión, registrarse o salir.
- En caso de iniciar sesión, el sistema captura un frame de la cámara y utiliza el módulo de reconocimiento facial para identificar al usuario.Si la identificación falla, se ofrece la opción de registrarse.

- En la opción de registro, se captura un frame para almacenar una nueva identidad facial en el sistema.

3. Preparación del entorno de juego

- Una vez autenticado, se recupera el nivel de progreso guardado para el usuario.
- Se muestra un resumen de estadísticas en pantalla con datos personalizados.

4. Bucle principal de juego

- La cámara captura frames de vídeo de forma continua.

- Se detecta la presencia de un marcador AR específico en el entorno (ID 10).

- Al detectar el marcador y si no hay preguntas cargadas, se cargan las preguntas correspondientes al nivel actual.

- Se muestra la figura visual asociada a la pregunta sobre el marcador detectado.

- Se despliega la pregunta y sus opciones de respuesta.

- El usuario puede responder usando el teclado o mediante reconocimiento por voz.

- El sistema evalúa la respuesta, muestra feedback visual y avanza a la siguiente pregunta.

- Cuando se terminan las preguntas del nivel, se muestra un resumen y se ofrece la opción de repetir el nivel, avanzar o salir.

5. Finalización

- El juego termina cuando el usuario decide salir o se detecta un fallo en la captura.

- Se liberan todos los recursos (cámara y ventanas).

Diseño del juego

Gestión de niveles

- Cada usuario tiene asignado un nivel actual, que se recupera al iniciar sesión.
- La estructura de preguntas está organizada por niveles dentro de la base de datos.
- Al completar un nivel con éxito, el sistema ofrece avanzar al siguiente nivel o repetir el actual.
- El nivel máximo permitido en esta versión es el nivel 2.

Carga de preguntas

- Las preguntas se cargan dinámicamente desde la base de datos al detectar el marcador AR.
- Cada pregunta incluye:
    - Texto de la pregunta.
    - Opciones de respuesta.
    - Figura visual asociada para mostrar en la interfaz.

Gestión de respuestas

- El jugador responde con las teclas numéricas (1 a 4) o mediante comandos de voz.
- La respuesta se compara con la correcta y se registra si fue acertada o no.
- Se proporciona retroalimentación visual durante un breve periodo tras la respuesta.
- Si la respuesta es correcta, se incrementa el contador de aciertos y se avanza a la siguiente pregunta.
- Si la respuesta es incorrecta, se avanza igualmente, pero sin sumar puntos.

Almacenamiento del progreso

- El progreso del usuario, incluyendo el nivel actual y estadísticas, se guarda en un sistema de usuarios persistente.
- Al terminar un nivel, el progreso se actualiza y queda disponible para futuras sesiones.
- La gestión y almacenamiento de los datos de usuario se realiza mediante el módulo reconocedor_cara.