CHANGELOG BRANCH DEVELOPMENT RV:

V1.1:

ADDED:

- Los guantes se ven correctamente en RV.

- No hay errores de compilacion.

CHANGED:

- Agregado SDK de Noitom HI5 (no el de Interaction ya que ese provoca errores al no usar sus objetos interactivos).

- Comentada linea de mHumanButtons.setVisible(value); para evitar que los botones de magnetizacion y ayuda se muestren en las manos de RV.

Script: HI5\_VIVEInstance.cs

- Cambiado el color del material de las manos para ponerlas de color carne.

V1.2:

ADDED:

- Soporte para capturar cuando el guante toca un cubo( es decir, capturar el primer estado de cada gesto).

CHANGED:

- Para realizar la captura del collider:

- Agrego un collider a cada cubo.

- Creacion del Script StateController.cs para capturar el collider del cubo 1 respecto al guante.

V1.3:

ADDED:

- Soporte para capturar el gesto del guante entero (es decir, capturar todos los estados).

- Todavia no tienen tiempo predefinido.

CHANGED:

- Script StateController.cs

V1.4:

ADDED:

- Primera version funcional de escena tutorial tempo 2/4.

- La informacion se saca por logs.

- Feedback sonoro que tambien habrá que cambiar.

CHANGED:

- StateController.cs

V1.7:

ADDED:

- Todos los cubos de los gestos de tempo (compas de 2/4, 3/4, 4/4) colocados correctamente.

- Scripts de deteccion de gestos de tempo: EN PROCESO.

CHANGED:

- Posicion de los cubos.

V1.9:

ADDED:

- Agregado HI5\_INTERACTION\_SDK.

- Reconocimiento de gestos agregado.

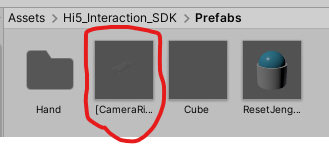
- Collider con las manos.

CHANGED:

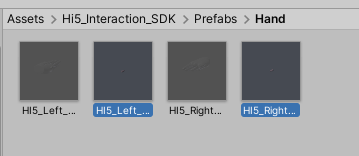
* 1. Seguir los pasos de las páginas 3, 4 y 5 del documento adjunto (documento 1) para preparar el entorno:



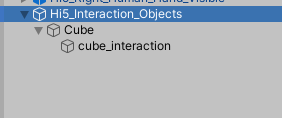
* 1. Arrastrar a la escena el prefab de "[CameraRig]\_HI5\_Interaction" que controla toda la interacción de los guantes. El prefab esta localizado en la siguiente ruta:



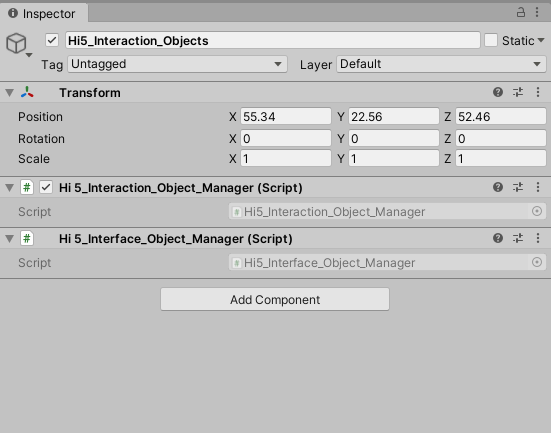
* 1. Agregar a la escena los prefabs de las manos visibles para poder verlas en VR. Ambos prefabs estan localizados en:



* 1. Seguir los pasos del apéndice del documento adjunto en el paso 1. (Hi5\_Unity\_Interaction\_SDK\_UserGuide\_1.1.0.35.pdf). Páginas 16 a 29. Estos pasos se encargan de configurar todos los prefabs necesarios para la interacción correcta de los scripts del SDK.
  2. El siguiente paso es agregar objetos de interaccion usados por los guantes. Si este paso no se lleva a cabo se produce un error constante ya que los scripts del SDK utilizan el objeto HI5\_Glove\_Interaction\_Item, inicializado solo sobre estos objetos.
  3. Para agregar los objetos del paso 5 se siguen los pasos del documento adjunto en el paso 1. (Hi5\_Unity\_Interaction\_SDK\_UserGuide\_1.1.0.35.pdf) desde la página 7 y a la 10 en la sección "make an object an interactive object".
  4. Para que los scripts de interacción con los objetos puedan comunicarse correctamente con ellos hay que agregar una capa por encima como Empty Object que SE DEBE llamar con el siguiente nombre "Hi5\_Interaction\_Objects". Las características de este empty object que hace de padre sobre todos los objetos interactivos que hemos creado en el paso 6 se representa en la siguiente imagen:



Jerarquía en Unity de los objetos interactivos. (HI5\_Interaction\_Objects representa la capa OBLIGATORIA agregada en el paso 7 y el resto de hijos son los objetos interactivos que hemos creado en el paso 6).



Características obligatorias de la capa HI5\_Interaction\_Objects ( La propiedad Transform puede tener cualquier valor).

V1.10:

ADDED:

* Escena ¾ por con el Script de gestos de tempo funcionando correctamente.
* Existe un problema con los colliders de cada cubo que por mucho que se reducen siguen siendo demasiado grandes.
* Reconocimiento de gestos activo. Gestos: Puño, Palma, Señalar dedo índice.

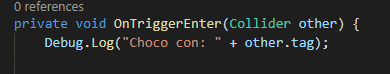
CHANGED:

COLLIDERS:

* Para detectar el collider correspondiente de cada mano podemos cambiar el tag de cada uno de los colliders de cada mano. Asi por ejemplo todos los colliders usados en la mano izquierda se tagean de la siguiente forma:



* Esto nos permite poder diferenciar en cualquier script el collider de cada mano y por lo tanto saber si el objeto esta siendo tocado por la mano izquierda o la mano derecha:

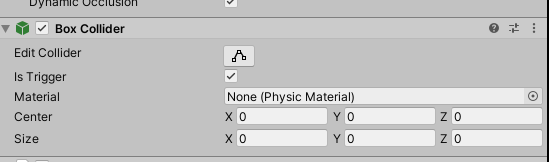


Other.tag permite consultar el tag del collider que choca contra el objeto que invoca este script y por ejemplo una de sus utilidades es diferenciar si ese objeto ha sido tocado por la mano izquierda o la mano derecha. Tambien agrego collider a las cajas que deben ser tocadas con la mano derecha solo.

* Reducir el collider del cubo para tocarlo solo cuando realmente lo tocas.

EVITAR QUE EL CUBO ROTE O SE MUEVA CUANDO LO TOCAMOS:

* En el componente BoxCollider isTrigger activado en el padre e hijo:



CAMBIOS A NIVEL DE SCRIPTS:

* El Script StateController3\_4.cs y StateController.cs quedan obsoletos. En su lugar se utiliza el Script StateControllerv2.cs

V2:

A partir del 26/05 donde presentamos una pequeña parte de la parte técnica desarrollada.

V2.1:

ADDED:

* Detección de paradas de la mano derecha para realizar un gesto asíncrono.

CHANGED:

* Script StateControllerv2.cs

V2.2:

ADDED:

[EN PROCESO] Reconocimiento de los gestos asíncronos con la detección de la orientación de la mano.

CHANGED:

* La idea es hacerlo usando un raycast que interactue con objetos invisibles que sirvan para determinar la orientación que tiene la mano teniendo en cuenta el tag o layer del objeto.
* Scripts cambiados:
* Ficheros modificados:

Las lineas de codigo añadidas en HI5 Interaction SDK irán encerradas entre un bloque con la directiva:

/\* MATR \*/ : Para abrir el bloque.

/\* EMATR \*/: Para cerrar el bloque.

* Scripts modificados:
* Hi5\_Glove\_Gesture\_Recognition.cs
* Hi5\_Object\_Judgement.cs
* Hi5\_Glove\_Collider\_Palm.cs
* Hi5\_Hand\_Visible\_Hand.cs
* Agrego AudioSource a PalmCollider.

V2.3:

demo para el Congreso.

Escenas tutoriales completas, escena avanzada sin soporte de reconocimiento de gestos.

V2.4:

Demo Congreso finalizado.

ADDED:

* Escena Tutorial 2/4: Aparentemente funciona bien. Hace falta revisión.
* Escena Tutorial ¾: Aparentemente funciona bien. Hace falta revisión.
* Escena Tutorial 4/4: No funciona.
* Escena Avanzado: Por hacer entera.

CHANGED:

* Ficheros de configuración de los gestos de tempo en las 4 escenas:

1. StateController2\_4v2.cs (Controlar los gestos de tempo de la escena Tutorial 2/4).
2. StateControllerv2.cs (Controlar los gestos de tempo de la escena Tutorial ¾).
3. StateController4\_4v2.cs (Controlar los gestos de tempo de la escena Tutorial 4/4).
4. StateControllerv2.cs (Incluye en la función Update una función para fakear por entrada de teclado el reconocimiento de gestos para el video del Congreso).

* Posición de las cajas y escala de las mismas en todas las escenas.

V3:

* Version parcialmente funcional para el Congreso EduLearn2020 finalizada.
* Al final de la V3 el objetivo es tener toda la aplicación totalmente funcional, sin preocuparse por ahora en estética o detalles adicionales como limpieza de código (V4).

V3.1:

ADDED:

* Escena 2/4 totalmente funcional y revisada.
* Detalles adicionales (como áreas de control para que el personaje no pueda moverse libremente por el mapa se añadirán en la versión V4).

CHANGED:

* Script controlador de toda la escena 2/4: StateController2\_4v2.cs.
* Añadido un GameObject que agrega la posibilidad de añadir una segunda línea de texto para cuando el usuario se equivoca. En este caso se utiliza para cuando el usuario tras haberse equivocado (primera línea de texto) toca cubos que no son el 1 (que es por el que tiene que empezar).

V3.2:

ADDED:

* Escenas Minijuego de Tempo funcionales completamente.
* Escena Avanzada funcional completamente. Necesario repaso y agregación del gesto inicial.

CHANGED:

* Escenas Minijuego de Tempo solucionados problemas a nivel de Scripts StateController.
* Escena Avanzada: Reestructuración completa para poder agregar la funcionalidad de cambio de tono.