

Tarea 02/A6

Implementación de mejoras-Mejora técnica de la herramienta de Realidad Virtual (VR)



This work is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-sa/4.0/)



"The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein."

INDEX

1. INTRODUCCIÓN	4
1.1. Metodología y funcionamiento de la aplicación	4
2. DESARROLLO DE LA HERRAMIENTA	4
3. DISEÑO DE ESCENAS	5
3.1. Selección de escenas	5
3.1.1. Drones (vehículos aéreos no tripulados): preparación para vuelos en obras de construcción a la luz del día	5
3.1.2. Drones (vehículos aéreos no tripulados) – Volar en obras de construcción en condiciones climáticas favorables	6
3.1.3. Drones (vehículos aéreos no tripulados) – Volar en obras de construcción en condiciones climáticas adversas	6
3.1.4. Drones (vehículo aéreo no tripulado): preparación para vuelos en sitios de construcción por la noche	6
3.1.5. Vehículo de transporte de sitio autónomo: condiciones del sitio interior	7
3.1.6. Vehículo de transporte autónomo del sitio: condiciones externas y exteriores del sitio	7
3.1.7. Equipo de control remoto (robots de demolición) - Requisitos generales	7
3.1.8. Equipo controlado a distancia (robots de demolición) - Condiciones del sitio interior	8
3.1.9. Equipo controlado a distancia (robots de demolición): condiciones externas y exteriores del sitio	8
3.1.10. Equipo de control remoto (excavadoras/excavadoras) - Condiciones externas y exteriores del sitio	8
3.2. Instrucciones generales	9
3.3. Panel de error	10
3.4. Panel de éxito	10
3.5. Tarea (ejemplo)	10
4. Desarrollo del Entorno Virtual 3D	11
5. DESARROLLO DE FUNCIONALIDADES Y EXPERIENCIA INMERSIVA.	17
5.1. Diseño y arquitectura	17
5.1.1. Menú principal	17
5.2. Modelado de datos	19
5.3. Integración del SDK	20

5.4.	Gestión de eventos.....	20
5.4.1.	Prepara la escena para la interacción de realidad virtual	20
5.4.2.	Entrada del controlador	21
5.4.3.	Administrador de cuestionarios	23
5.4.4.	Translator	24
5.4.5.	Interacción con rayos	25
6.	Conclusión	26

1. INTRODUCCIÓN

El consorcio SafeCRobot ha producido una herramienta de capacitación inmersiva e interactiva basada en realidad virtual que consta de 10 escenarios de riesgo (cada escenario consiste en un tipo de robot de construcción / vehículo autónomo y los riesgos asociados con ellos) y las medidas preventivas necesarias para su mitigación. El objetivo principal de la herramienta inmersiva basada en realidad virtual desarrollada para este proyecto será utilizar la herramienta para capacitar a los trabajadores de la construcción para que puedan trabajar de manera más segura junto con robots de construcción y equipos autónomos en sitios de construcción.

Se mejoró la herramienta de entrenamiento inmersivo, por lo que se modificó la versión beta de la herramienta con las mejoras identificadas en la evaluación por parte de los colaboradores, así como las pruebas en los cursos de los participantes. Además, también se consideraron las conclusiones técnicas de los Seminarios Internacionales.

1.1. Metodología y funcionamiento de la aplicación.

La investigación y el desarrollo de SafeCRobot se han dividido en varias partes distintas correspondientes al desarrollo de los paquetes de trabajo planificados.

En las siguientes secciones se describe la metodología de evaluación en la que se ha basado el proyecto, así como su funcionamiento y las mejoras implementadas.

2. DESARROLLO DE LA HERRAMIENTA.

El SafeCRobot (en adelante denominado como "herramienta") se basa en la realidad virtual inmersiva. Actualmente, la forma más común de desarrollar aplicaciones con esta tecnología es mediante motores de desarrollo de juegos como Unity3D.

El desarrollo de la herramienta se puede dividir en tres tareas principales que se cubrirán ampliamente en las siguientes secciones:

- **Diseño de las escenas.**
- **Desarrollo de los entornos 3D.**
- **Desarrollo de la funcionalidad y experiencia inmersiva.**

Para llevar a cabo el desarrollo de manera eficiente y efectiva, cabe destacar que se han utilizado diferentes tecnologías:

- **Autodesk Revit:** Autodesk Revit es una herramienta de software de modelado de información de construcción para arquitectos, ingenieros estructurales, ingenieros mecánicos, eléctricos y de plomería, diseñadores y contratistas.

- **Blender:** Software multiplataforma dedicado al procesamiento, modelado, renderizado y otras tareas relacionadas con gráficos 3D. Es de código abierto y tiene una gran comunidad con videos y documentación, lo que hace que sea muy fácil de aprender.
- **Unity3D:** Este es otro software de código abierto, un motor de desarrollo de juegos que ayudó en el desarrollo de un entorno virtual inmersivo. Es altamente escalable, permitiendo a los usuarios añadir módulos mediante scripts de código, y además cuenta con una interesante comunidad y documentación.
- **Oculus Quest:** Este es un dispositivo de realidad virtual muy asequible y accesible. Desarrollado por Meta como uno de los pilares de su nueva filosofía. No requieren cables para su uso y tienen una fácil integración con Unity.

3. DISEÑO DE ESCENAS.

Este es el proceso mediante el cual se diseñan cada uno de los pasos a seguir desde el punto de entrada en la herramienta hasta completar la inmersión. En otras palabras, generar el script de desarrollo para la aplicación.

El primer paso fue identificar varios tipos de peligros asociados con la aplicación de robots / vehículos autónomos en sitios de construcción a través de una rigurosa revisión de la literatura. Una vez que los peligros están identificados y categorizados (peligro para uno mismo y peligro para otros); Se modelaron 10 escenarios de riesgo basados en los peligros identificados utilizando las herramientas mencionadas anteriormente. Los escenarios seleccionados son los siguientes:

3.1. Selección de escenas

3.1.1. Drones (vehículos aéreos no tripulados): preparación para vuelos en obras de construcción a la luz del día

Durante este escenario, el usuario asumirá el papel de piloto de un vehículo aéreo no tripulado (UAV). La tarea del usuario será prepararse adecuadamente para el vuelo durante el día en el sitio de construcción. El usuario se encuentra en la oficina de construcción. El contenedor contiene el siguiente equipo: un dron, un conjunto de baterías cargadas, una tableta y documentación de construcción. En la pared el usuario encontrará el desarrollo del sitio de construcción. Teniendo en cuenta algunos de los problemas de seguridad más importantes para

volar drones en sitios de construcción, los usuarios deben responder preguntas en el cuestionario dentro de la escena.

3.1.2. Drones (vehículos aéreos no tripulados) – Volar en obras de construcción en condiciones climáticas favorables

Durante este escenario, el usuario asumirá el papel de piloto de un vehículo aéreo no tripulado (UAV). La tarea será llevar a cabo una misión (incursión en el sitio de construcción) utilizando un dron en un día brillante y soleado. Durante la incursión, tendrán que prestar atención a su entorno y a los mensajes que aparecen en la pantalla del controlador con respecto a los parámetros técnicos del vuelo. Teniendo en cuenta algunos de los problemas de seguridad más importantes para volar drones en sitios de construcción, los usuarios deben responder preguntas en el cuestionario dentro de la escena.

3.1.3. Drones (vehículos aéreos no tripulados) – Volar en obras de construcción en condiciones climáticas adversas

Durante este escenario, los usuarios asumirán el papel de piloto de un vehículo aéreo no tripulado (BSP). La tarea será llevar a cabo una misión (incursión en el sitio de construcción) utilizando un dron durante condiciones climáticas adversas, incluidos el viento y la lluvia. Durante la incursión, los usuarios deberán prestar atención al entorno y a los mensajes que aparecen en la pantalla del controlador con respecto a los parámetros técnicos del vuelo. Teniendo en cuenta algunos de los problemas de seguridad más importantes para volar drones en sitios de construcción, los usuarios deben responder preguntas en el cuestionario dentro de la escena.

3.1.4. Drones (vehículo aéreo no tripulado): preparación para vuelos en sitios de construcción por la noche

Durante este escenario, los usuarios asumirán el papel de piloto de un vehículo aéreo no tripulado (UAV). La tarea será prepararse adecuadamente para volar de noche en un sitio de construcción. El usuario se encuentra en la oficina de construcción. En el contenedor, el usuario encontrará el siguiente equipo: un dron, un conjunto de baterías cargadas, una tableta, documentación de construcción e iluminación adicional. Mire alrededor de la oficina y prepárese para volar. Teniendo en cuenta algunos de los problemas de seguridad más importantes para volar drones en sitios de construcción, los usuarios deben responder preguntas en el cuestionario dentro de la escena.

3.1.5. Vehículo de transporte de sitio autónomo: condiciones del sitio interior

Este módulo presenta a los usuarios los requisitos de salud y seguridad para operar **vehículos de transporte autónomos (ATV)** en condiciones de construcción en interiores. ATV es un vehículo capaz de operar por sí mismo. Por lo general, son con ruedas u orugas y varían en tamaño. Para el transporte en interiores, los ATV son normalmente de tamaño pequeño-mediano. Supongamos que el ATV utilizado en este módulo es eléctrico y tiene las siguientes dimensiones: L-1200mm, H-600mm y W-700mm y puede transportar cargas de hasta 500kg. Los usuarios deben observar el medio ambiente y el equipo para identificar cualquier problema que consideren riesgos para la salud y la seguridad. Teniendo en cuenta algunos de los problemas de seguridad más importantes para administrar un ATV en un sitio de construcción, los usuarios deben responder preguntas en el cuestionario dentro de la escena.

3.1.6. Vehículo de transporte autónomo del sitio: condiciones externas y exteriores del sitio

Este módulo presenta al usuario los requisitos de salud y seguridad para operar **vehículos de transporte autónomos (ATV)** en condiciones de construcción externas o al aire libre. Para actividades externas o al aire libre, los ATV varían de equipos pequeños a muy grandes. El ATV utilizado en esta escena es un gran volquete utilizado para transportar material en el sitio. Los materiales típicos incluyen material agregado, excavado o demolido. Observe el entorno y el equipo para identificar cualquier problema que considere como riesgo para la salud y la seguridad. Teniendo en cuenta algunos de los problemas de seguridad más importantes para administrar un ATV en un sitio de construcción, los usuarios deben responder preguntas en el cuestionario dentro de la escena.

3.1.7. Equipo de control remoto (robots de demolición) - Requisitos generales

Este módulo presenta a los usuarios los requisitos de salud y seguridad para operar un robot de demolición controlado a distancia en condiciones de sitio de construcción (interiores y exteriores). El contratista Smith encarga a dos de sus empleados, Marc y Gordon, demoler varias paredes dentro / fuera de una gran fábrica industrial. Se utilizará un robot de demolición a control remoto. El martillo hidráulico ya ha sido montado. Marc y Gordon nunca han trabajado con un robot de demolición antes, pero están deseando que llegue. Únase a Marc y Gordon en su trabajo y descubra los riesgos para la salud y la seguridad al trabajar con el robot de demolición. Luego, los usuarios verificarán y consolidarán sus conocimientos trabajando a través del cuestionario.

3.1.8. Equipo controlado a distancia (robots de demolición) - Condiciones del sitio interior

Este módulo presenta a los usuarios los requisitos de salud y seguridad para operar un robot de demolición controlado a distancia en condiciones de sitio de construcción (interiores y exteriores). El contratista Smith encarga a dos de sus empleados, Marc y Gordon, demoler varias paredes dentro / fuera de una gran fábrica industrial. Se utilizará un robot de demolición a control remoto. El martillo hidráulico ya ha sido montado. Marc y Gordon nunca han trabajado con un robot de demolición antes, pero están deseando que llegue. Únase a Marc y Gordon en su trabajo y descubra los riesgos para la salud y la seguridad al trabajar con el robot de demolición. Luego, los usuarios verificarán y consolidarán sus conocimientos trabajando a través del cuestionario.

3.1.9. Equipo controlado a distancia (robots de demolición): condiciones externas y exteriores del sitio

Este módulo presenta a los usuarios los requisitos de salud y seguridad para operar un robot de demolición controlado a distancia en condiciones de sitio de construcción (interiores y exteriores). El contratista Smith encarga a dos de sus empleados, Marc y Gordon, demoler varias paredes dentro / fuera de una gran fábrica industrial. Se utilizará un robot de demolición a control remoto. El martillo hidráulico ya ha sido montado. Marc y Gordon nunca han trabajado con un robot de demolición antes, pero están deseando que llegue. Únase a Marc y Gordon en su trabajo y descubra los riesgos para la salud y la seguridad al trabajar con el robot de demolición. Luego, los usuarios verificarán y consolidarán sus conocimientos trabajando a través del cuestionario.

3.1.10. Equipo de control remoto (excavadoras/excavadoras) - Condiciones externas y exteriores del sitio

Este módulo presenta a los usuarios los requisitos de salud y seguridad para operar equipos controlados a distancia utilizando excavadoras / excavadoras como ejemplo. El escenario representa la construcción en condiciones externas o exteriores del sitio. Una excavadora o excavadora es un equipo que consiste en una pluma, un cucharón, un cucharón y una cabina en una plataforma giratoria. Se utilizan principalmente para excavar y excavar material. Por lo general, son con ruedas u orugas y varían en tamaño. La excavadora utilizada en esta escena es de tamaño mediano. Los usuarios deben observar el entorno y el equipo para identificar cualquier problema que consideren riesgos para la salud y la seguridad. Teniendo en cuenta

algunos de los problemas de seguridad más importantes para la gestión de una excavadora / excavadora a control remoto en un sitio de construcción, los usuarios deben responder preguntas en el cuestionario para la escena.

Sobre la base del guión de escena detallado anteriormente, el modelado 3D del entorno de construcción virtual se llevó a cabo utilizando herramientas de modelado previamente detalladas.

Para simular cada uno de los pasos mientras los usuarios están en la escena, se han generado diferentes paneles con información que el usuario tendrá que leer para actuar. Los paneles que se encuentran en cada escena se pueden clasificar en los siguientes tipos.

3.2. Instrucciones generales

Estos son paneles que aparecen al principio de cada escena. Explican el procedimiento a seguir para llegar a las misiones, así como información general sobre ciertas herramientas. Cada misión comenzará con un panel de bienvenida y dos paneles con instrucciones básicas.

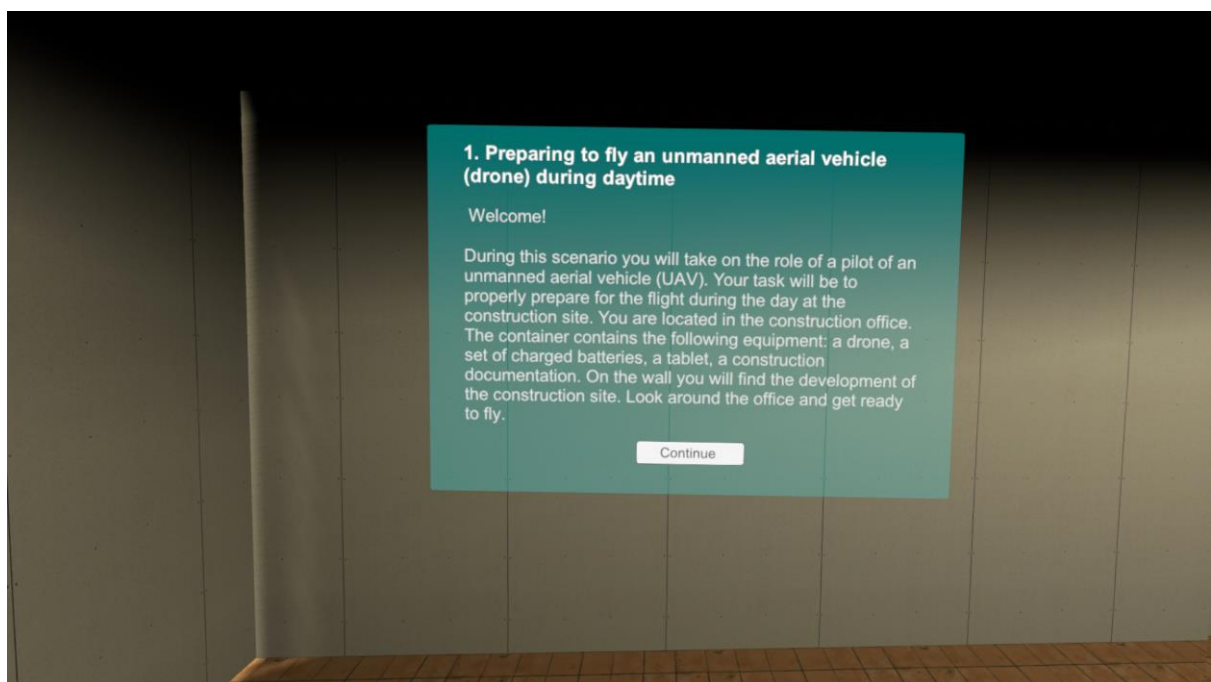


Figura 1. Pantalla de bienvenida con información e instrucciones de la escena

Panel de prueba: Este panel presentará un conjunto de preguntas relacionadas con el peligro asociadas con la escena relevante y los usuarios pueden usar el puntero láser y el botón de disparo en el controlador para responder las preguntas que se registrarán y los resultados se mostrarán al final del cuestionario.

3.3. Panel de error

Cada vez que se falla una misión, aparecerá el siguiente panel:

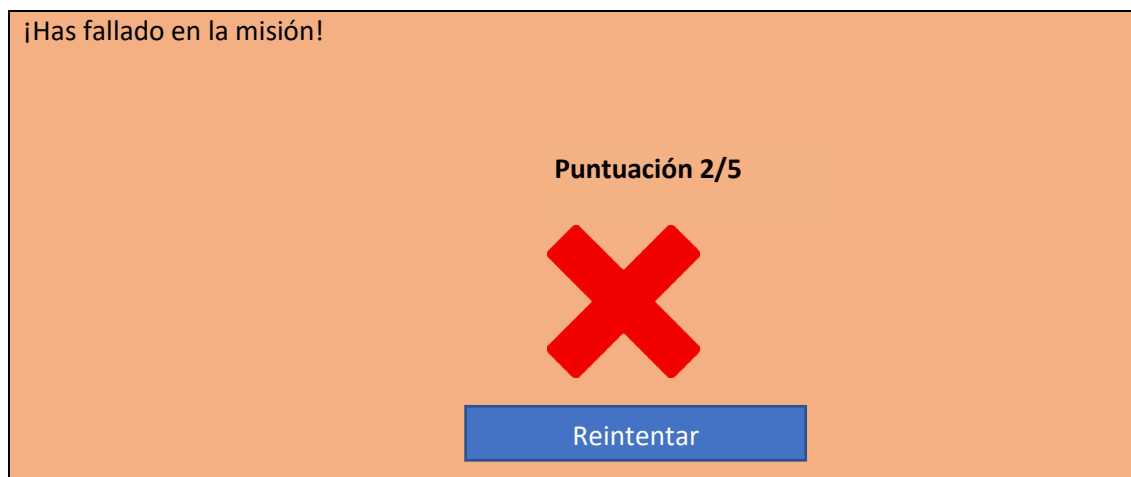


Figura 2. Panel de fallos.

3.4. Panel de éxito

Cada vez que se complete una misión, aparecerá el siguiente panel:



Figura 3. Panel de éxito.

3.5. Tarea (ejemplo)

Estos paneles variarán según la tarea a realizar, conteniendo información, preguntas, diálogos u órdenes a completar.

Tarea 5:

Vehículo de transporte autónomo – Condiciones del sitio interior (consulte los detalles en la sección 3):

Continuar

<p>Examen:</p> <p>¿Cuál de los siguientes riesgos para la salud existe en el sitio que acaba de revisar?</p> <ol style="list-style-type: none"> 1. Colisión (v) 2. Trastornos musculoesqueléticos 3. Captura (v) 4. Trituración (v) 5. Hematomas 	
<p>Examen</p> <p>¿Cuál de las siguientes afirmaciones sobre la escena es cierta?</p> <ol style="list-style-type: none"> 1. La planificación de la ruta para el vehículo fue deficiente (v) 2. Los sistemas de advertencia de vehículos eran muy buenos 3. Las señales de advertencia y seguridad son inadecuadas (v) 4. Existe el riesgo de volcar 5. El operador o el banquero no es identificable (v) 6. Los trabajadores mantuvieron una distancia adecuada entre ellos y el vehículo 7. El vehículo debe estar separado de los trabajadores 8. Mala demarcación de los pasos de peatones (v) 	
<p>Examen</p> <p>¿Cuál de los siguientes factores puede contribuir al riesgo de que una máquina se caiga?</p> <ol style="list-style-type: none"> 1. Ruta del vehículo demasiado cerca de las aberturas (v) 2. El sitio es demasiado pequeño para el vehículo 3. Carga desequilibrada y sobrecarga (v) 4. Falta de uso de barricadas 5. Ruido 	
(Panel de éxito)	(Panel de error)

Tabla 1. Ejemplo de cuestionario de tareas

4. Desarrollo del Entorno Virtual 3D

Unity 3D se utilizó como motor de desarrollo de juegos y los modelos 3D desarrollados para la creación de escenas se denominan objetos de juego. Así, la segunda gran tarea consiste en generar cada uno de los objetos o activos que se definen directa e indirectamente en los scripts generados en el apartado anterior.

Este proceso se divide a su vez en tres tareas:

- **Modelado:** Una representación matemática de un objeto tridimensional se realiza utilizando un software específico. En este caso, se han utilizado Blender y Autodesk

Revit. Algunos de los objetos del juego se adquirieron de la tienda de activos y están listos para usar.

- **Rigging:** Este es el proceso por el cual las diferentes partes de un objeto se separan para crear un sistema controlable que permite animaciones efectivas y eficientes. La funcionalidad de animación de Unity3D solo se utilizó para esto.
- **Texturizado:** Este es el proceso por el cual los objetos 3D reciben color y detalle. Blender y el editor de material incorporado de unidades se han utilizado para lograr esto.

La lista de objetos desarrollados para este proyecto es muy amplia y diversa, desde objetos muy simples hasta máquinas complejas con mucho detalle. Los objetos generados se pueden clasificar en las siguientes categorías.

- Contenedores, edificios, tiendas, humanoides y otros elementos como (no limitado a) polvo, lluvia, etc. para recrear un entorno de construcción realista.

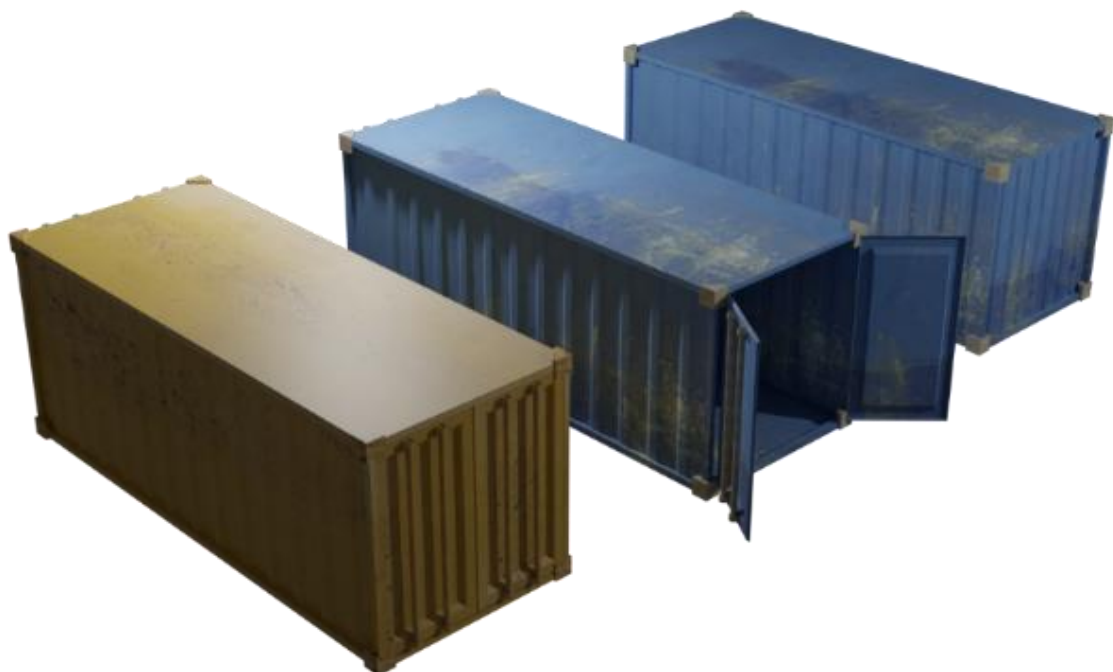


Figura 4. Contenedores (Ejemplos de objetos de juego utilizados para el desarrollo de escenas)



Figura 5. Demoledor (Ejemplo de objetos de juego utilizados para el desarrollo de escenas)



Figura 6. Trabajadores de la construcción (Objetos de juego de ejemplo utilizados para el desarrollo de escenas)



Figura 7. Demoledor en acción



Figura 8. Sitio de construcción activo (desarrollado utilizando los objetos del juego)

Después del desarrollo de cada uno de los objetos a utilizar, se preparan los escenarios para que el desarrollo de la funcionalidad sea lo más preciso y cercano posible a la situación final. Este proceso no suele ser definitivo, ya que el proceso de prueba puede dar lugar a modificaciones. Este proceso se ha llevado a cabo con el motor gráfico Unity, ya que es el que se utilizará para generar la aplicación.

Si analizamos las descripciones de cada una de las situaciones descritas en el apartado anterior, podemos ver que existen tres escenarios diferentes:

- **Contenedor (oficina del sitio):** En el contenedor, el usuario encontrará el siguiente equipo: un dron, un conjunto de baterías cargadas, una tableta, documentación de construcción e iluminación adicional.

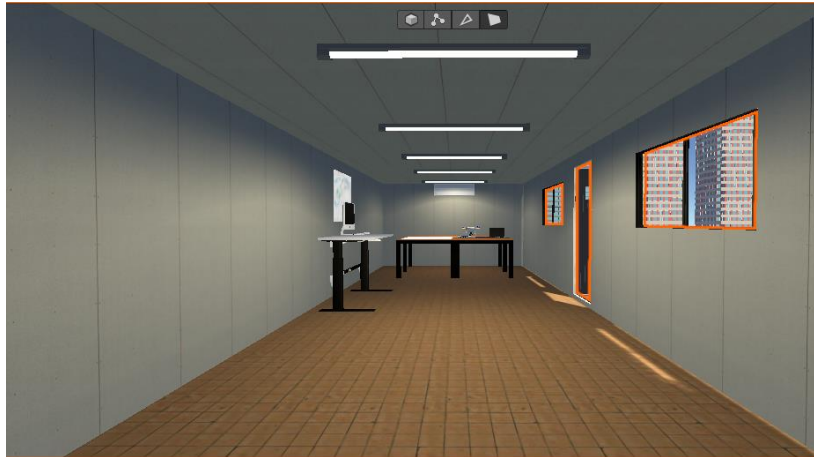


Figura 9. Oficina de contenedores



Figura 10. Drone y accesorios en la oficina de contenedores

- **Sitio de construcción:** Este modelo muestra un sitio de construcción con, andamios, el esqueleto de un edificio y vehículos autónomos relevantes.



Figura 11. Sitio de construcción activo

5. DESARROLLO DE FUNCIONALIDADES Y EXPERIENCIA INMERSIVA.

El desarrollo de la funcionalidad es el proceso por el cual toda la teoría recopilada toma forma. En otras palabras, el objetivo de esta tarea es hacer que la herramienta sea utilizable. El motor gráfico Unity se ha utilizado para desarrollarlo, ya que proporciona una integración simple de Realidad Virtual y tiene una gran comunidad.

Los proyectos en Unity se dividen en escenas, que se pueden entrelazar con eventos. Las escenas se componen de diferentes objetos que contendrán ciertos componentes, que dan diferentes características o comportamientos a los objetos.

El desarrollo de la funcionalidad es una tarea amplia, pero se puede subdividir en varios procesos: **Diseño y Arquitectura, Modelado de Datos, Integración de SDK y Gestión de Eventos.**

5.1. Diseño y arquitectura

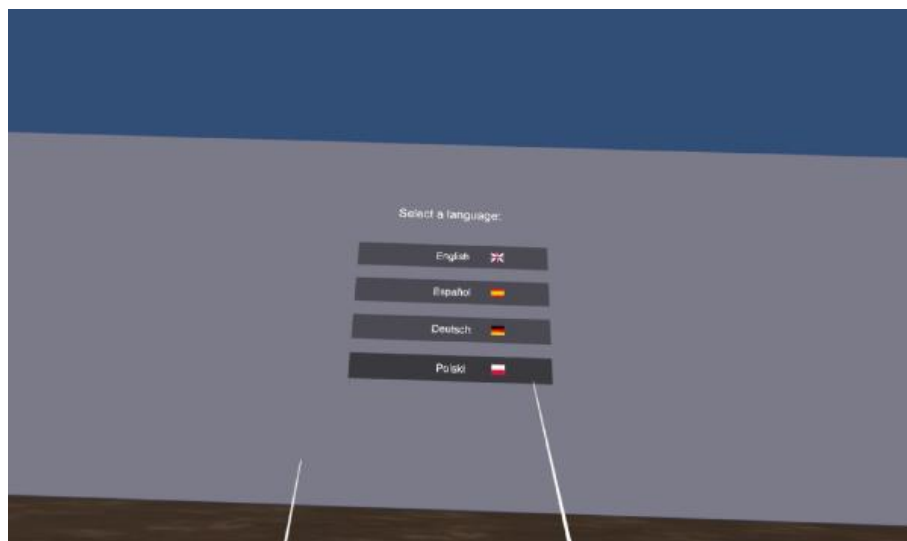
Este es el proceso de dar forma al guión desarrollado en apartados anteriores, es decir, idear el camino y la toma de decisiones que conllevará cada escena o pantalla.

La experiencia de la herramienta tiene dos partes diferenciadas. La estructura que se presentará se muestra en el diagrama de flujo de la Figura 9.

5.1.1. Menú principal

La primera pantalla permite al usuario adaptarse al entorno virtual en el que se verá involucrado durante el uso de la herramienta. Aunque los movimientos son limitados con el fin de centrar la atención del usuario en el objetivo real de la escena, que es seleccionar una de las situaciones propuestas para el uso correcto de los nano productos. También es necesario seleccionar el idioma en el que el usuario desea que se ejecute la aplicación.

La siguiente pantalla dependerá de la selección realizada en esta pantalla.



a)



b)

Figura 12: Menú principal (a) Selección de idioma (b) Selección de escenario

Escena: Una vez que el usuario haya elegido el escenario específico que desea probar, el usuario será llevado a esa escena con una pantalla de bienvenida flotante donde se detallarán los detalles de la escena y el requisito de la tarea. Si el usuario falla la tarea, puede rehacerla 3 veces seguidas; Un fracaso adicional resultará en redirigirlos para obtener un mejor conocimiento teórico antes de volver a intentarlo. Si pasan, pueden ir al siguiente escenario y así sucesivamente.

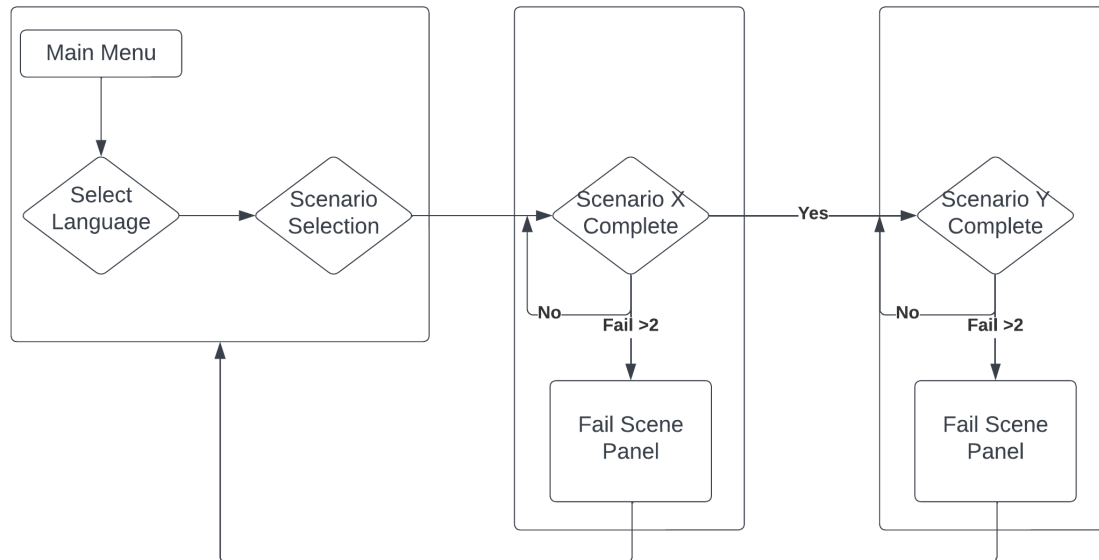


Figura 13: Diagrama de flujo de la herramienta.

5.2. Modelado de datos

En esta tarea, el objetivo es generar un esquema general en el que se almacena y recupera la información. Para conseguirlo, se ha utilizado el módulo `PlayerPrefs`, que es una clase proporcionada por Unity que permite almacenar datos como una clave con el nombre del proyecto.

Estas variables en una aplicación de Oculus se almacenan como un archivo XML dentro del directorio `/data/data/pkg-name/shared_prefs/pkg-name.v2.playerprefs.xml`, donde `pkg-name` será el nombre de la aplicación.

Además, Unity proporciona un acceso muy fácil a este conjunto de datos gracias a las funciones proporcionadas por la clase `PlayerPrefs`.

La única variable que será común y que tendremos que almacenar, debido a la estructura de esta aplicación, será la que haga referencia al lenguaje. Para ello, utilizaremos las siguientes funciones:

- **SetString (nombre, valor):** Esta función nos permitirá escribir un valor de cadena sobre la variable `name`. En este caso, se introducirá el código ISO de referencia del idioma seleccionado en el menú.
- **GetString (nombre):** Esta función devuelve el valor del nombre de la variable. Se utilizará en cada una de las escenas para saber qué idioma se selecciona.

5.3. Integración del SDK

El desarrollo de cualquier tipo de aplicación de Realidad Virtual requiere tres componentes. Un dispositivo compatible que se puede conectar al ordenador, un motor gráfico de videojuegos y un kit de desarrollo de software (SDK). Los dos primeros requisitos fueron explicados al principio del capítulo y serán Oculus y Unity, respectivamente.

Un SDK es cualquier conjunto de herramientas que permiten a un desarrollador de software en el proceso de creación. En el caso de la Realidad Virtual, y en particular los paquetes adaptables a Unity, existen varios, incluido uno específico para Oculus. Pero se ha elegido el XR Interaction Toolkit. Un paquete desarrollado por Unity que permite adaptar el proyecto o aplicación a cualquier tipo de dispositivo. En otras palabras, incluso si la herramienta está desarrollada con Oculus Quest, se puede instalar en cualquier dispositivo compatible con Unity. Por lo tanto, obtienes un alcance más amplio.

Para crear un proyecto que introduzca el SDK y permita la interacción en Unity, se deben seguir los siguientes pasos:

1. Al abrir el motor, genere un proyecto con la plantilla *Universal Render Pipeline*. Es importante hacerlo con esta plantilla ya que proporciona gráficos optimizados, y este es un requisito de la Realidad Virtual si queremos tener una experiencia correcta.
2. Instale el SDK, es decir, el paquete XR Interaction Toolkit. Esto se puede hacer desde la ventana Administrador de paquetes, a la que se accede desde la opción de menú Ventana.
3. Tendremos que indicar el tipo de dispositivo sobre el que se va a construir. Para hacer esto, active la opción compatible con VR en Configuración del proyecto / Reproductor.

De esta manera, el proyecto Unity está listo para el desarrollo de Realidad Virtual. Sin embargo, para iniciar la interacción del dispositivo con el entorno gráfico, se deben seguir los pasos descritos en la siguiente sección.

5.4. Gestión de eventos

La gestión de eventos podría considerarse la tarea más extensa dentro del desarrollo de la funcionalidad. Consiste en utilizar los componentes proporcionados por Unity y generar pequeños scripts para lograr la interacción usuario-máquina esperada.

Esta sección podría ser demasiado larga, ya que cada escena tiene muchos detalles específicos a nivel de interacción. Pero aquí solo se detallarán los puntos comunes e importantes para lograr el objetivo.

5.4.1. Prepara la escena para la interacción de realidad virtual

Una vez que el proyecto de Unity se ha preparado y los escenarios se han configurado o preparado gráficamente, el siguiente paso será preparar la escena para que el dispositivo de interacción se pueda vincular a una cámara, y podamos usarlo para pruebas durante el desarrollo. El proceso explicado en los siguientes pasos se repetirá para cada una de las escenas que se utilizarán en el proyecto:

1. Lo primero es eliminar la cámara predeterminada.
2. Genere un objeto vacío con un componente XR Rig, que será el contenedor de toda la interacción hombre-máquina. Lo llamamos VR-Rig.
3. Generar un objeto hijo, también vacío, que será la posición de referencia, donde el usuario iniciará la interacción. Esto se llamará Desplazamiento de cámara.
4. Genere una cámara como elemento secundario de Desplazamiento de cámara, con un componente Controlador de pose rastreado. Este es el objeto que actuará como nuestros ojos y el componente añadido que nos permitirá girar.
5. Después de esto, rellene las variables del componente XR Rig (objeto padre, VR-Rig) con la posición de referencia y la cámara creada. Y colocar el suelo como valor del atributo Tracking Origin Mode, esto hará que la cámara se adapte a nuestra altura automáticamente. De esta manera, conseguimos que los ojos del sistema sean completamente funcionales.
6. Para lograr la movilidad y la visibilidad de los controladores, se generan dos nuevos hijos dentro del objeto Camera Offset y se agrega el atributo XR Controllor.
7. Agregue el modelo que queremos usar como controladores en el atributo Model Prefab.
8. Finalmente, a estos dos objetos, añadimos el componente XR Direct Interactor, que dará a los controladores interactividad con el resto de objetos de la escena.

5.4.2. Entrada del controlador

Este es el proceso mediante el cual se obtienen los valores generados al presionar cada uno de los botones de las perillas. Esto se utilizará para una gran cantidad de tareas, como elegir objetos, seleccionar opciones, etc.

Antes de generar algo con estos valores, es importante saber qué tipo de datos maneja cada entrada. Esto se puede ver en la ventana del *depurador de interacción XR*, a la que puede acceder desde Ventana / Análisis.

Con esta información, el siguiente paso es generar un script para obtener estos valores. La clase que se ha generado con esto se denomina *HandController* y se utiliza como componente de cada uno de los modelos 3D que se agregaron en el atributo Model Prefab de los controles. Los aspectos más destacados del código son:

```
public class HandController : MonoBehaviour
{
    //Input
    private InputDevice targetDevice;

    //Controller model options
    public bool showController = false;
    public List<GameObject> controllers;
    private GameObject spawnedController;

    //Device characteristics
    public InputDeviceCharacteristics controllerChar;

    //Hand model
    public GameObject handModel;
    private GameObject spawnedHandModel;

    private Animator handAnimator;

    //Get if is Right or left
    public string isRight;

    //Input variables
    public float trigVal;
    public bool primaryButton;
    public float gripValue;
```

Figura 14: Clase HandController

1. Lo primero que ves en la imagen de arriba es la declaración de variables. Los que son públicos corresponderán a atributos alterables del componente.
2. Al generar un nuevo script, siempre se crean automáticamente dos funciones: *Start()*, que se ejecuta una vez al inicio de la instanciación, y *Update()*, que se ejecuta iterativamente durante la vida útil del componente.
3. Lo primero que sucede dentro de este componente es la función *TryInitialize()*, en esta, se obtiene el dispositivo VR según sus características. Para cada controlador, la característica será si es derecha o izquierda. Esto se logra llamando a la función *GetDevicesWithCharacteristics()* dentro del módulo *InputDevices*.

```
//Try get input device and attach the selected model to it
void TryInitialize()
{
    //Get VR devices by characteristics
    List<InputDevice> devices = new List<InputDevice>();
    InputDevices.GetDevicesWithCharacteristics(controllerChar, devices);

    if (devices.Count > 0)
    {
        //Get target device and get a controller model
        targetDevice = devices[0];
        GameObject prefab = controllers.Find(con => con.name == targetDevice.name);

        if (prefab)
        {
            spawnedController = Instantiate(prefab, transform);
        }
        else
        {
            Debug.LogError("Did not find corresponding controller model");
            spawnedController = Instantiate(controllers[0], transform);
        }

        //Instantiate Hand Model and get Animator component
        spawnedHandModel = Instantiate(handModel, transform);
        handAnimator = spawnedHandModel.GetComponent<Animator>();
    }
}
```

Figura 15: Función TryInitialize()

4. A continuación, se ejecuta la función `Update()`, que llama a las funciones `UpdateEvents()` y `UpdateAnimations()`.
5. `UpdateEvents()` se encarga de obtener cada uno de los valores generados por el controlador a través de la función `TryGetFeatureValue()` y almacenarlos en las variables públicas que se generaron en primera instancia.
6. `UpdateAnimations()` es una función creada para activar las animaciones que cada uno de los botones supondría en caso de tener un modelo de mano amañado. En cualquier caso, no es una función importante en la tarea de obtener los valores.

```
//Update hand events
void UpdateEvents()
{
    //Trigger event
    if(targetDevice.TryGetFeatureValue(CommonUsages.trigger, out float triggerVal))
    {
        trigVal = triggerVal;
    }

    //PrimaryButton event
    if (targetDevice.TryGetFeatureValue(CommonUsages.primaryButton, out bool primary))
    {
        primaryButton = primary;
    }

    //Grip event
    if (targetDevice.TryGetFeatureValue(CommonUsages.grip, out float gripVal))
    {
        gripValue = gripVal;
    }

    //Joystick event
    if(targetDevice.TryGetFeatureValue(CommonUsages.primary2DAxis, out Vector2 axis))
    {
        axisVal = axis;
    }
}
```

Figura 16: Función `UpdateEvents()`.

5.4.3. Administrador de cuestionarios

Para hacer el desarrollo más fácil y rápido, se generó una clase llamada *QuizManager* para estandarizar el proceso de creación de paneles. Este elemento consta de los siguientes pasos:

1. Las variables se generan para todas las preguntas, la pregunta actual, la respuesta correcta y los elementos de la interfaz de usuario.
2. Se genera la función `InitQuiz()`, que se encarga de inicializar cada una de las variables que se necesitarán. A continuación, llamará a las funciones `SetNextQuestion()` para inicializar el cuestionario.

3. *SetNextQuestion()* y *SetAnswerOptions()*: Estas funciones actualizan las variables y los elementos del panel como textos y botones, estableciendo el valor de la siguiente pregunta en cola, eliminando la anterior si había una.
4. La función que se activa cuando se presiona un botón de respuesta es *SelectButton(Button but)*. En el caso de una pregunta con una sola respuesta, esto llamará a la función *CheckResponse()*.
5. *Checkresponse ()*: Es la función con la lógica que se encarga de comprobar si la respuesta seleccionada es la correcta. Si es así, pasará al siguiente panel, si no lo está, agregará una falla a la escena y dará lugar al panel de falla o al panel de falla de la escena.



Figura 17: Interfaz de cuestionario

5.4.4. Translator

Este proceso actúa sobre cada una de las escenas en segundo plano para proporcionar a cada elemento de la interfaz de usuario de la herramienta los textos traducidos al idioma seleccionado.

En primer lugar, se ha llevado a cabo una traducción de los textos a cada uno de los idiomas de los socios participantes. Estos documentos tendrán un formato en el que cada palabra o frase tendrá una clave, lo que nos permitirá acceder a esta frase en cualquiera de los idiomas.

Para ello, se han creado dos clases:

- *MainTranslate*: Esta clase supervisa la creación del vínculo entre los guiones de traducción y el entorno de trabajo, en este caso, el código. Cada vez que se realiza una búsqueda de esta clave generada en el script anterior, esta clase devolverá la palabra o frase en el idioma correcto.
- *SceneTranslate*: Esta clase supervisa indicar a *MainTranslate* al comienzo de cada nueva situación, cuál es el idioma de búsqueda seleccionado en el menú.

Con estos procesos implementados, todo lo que debe hacer es cambiar cada una de las líneas de texto que pretendía ingresar a la siguiente línea de código:

```
text = MainTranslate.Fields[textKey];
```

Figura 18: Línea de traducción

5.4.5. Interacción con rayos

Esta tarea permite al usuario interactuar con objetos o paneles a cierta distancia. Consiste en generar un haz que sale de la mano y permite al usuario realizar acciones sobre determinados objetos. Para este proyecto, será de gran importancia responder a las preguntas que se han planteado.

Para generar este tipo de interacción, serán necesarios los siguientes pasos:

1. Cree un objeto Ray Interactor que aparezca en el menú GameObject/XR para cada una de las manos.
2. A continuación, seleccione los objetos sobre los que se va a actuar. Para ello, cambie el parámetro Raycast Mask del componente XR Ray Interactor. Este atributo nos dará la opción de elegir sobre qué Capa queremos que actúe el rayo para que los objetos sobre los que queremos que actúe el rayo tengan que tener esa Capa.
3. En este caso, hemos seleccionado la capa de interfaz de usuario de los paneles proporcionados por Unity.
4. Finalmente, para que el rayo no aparezca todo el tiempo, tendremos que cambiar el atributo Invalid Color Gradient del componente XR Interactor Line Visual a un valor transparente. De modo que solo cuando la mano apunte a un panel aparecerá el rayo.
5. Estos dos objetos se introducirán como hijos de Camera Offset, un objeto creado en la preparación de la escena para que se incluya en el objeto referente al dispositivo VR.

Con la implementación de esta sección, obtenemos la funcionalidad que nos permite generar cada uno de los diferentes paneles descritos e interactuar con ellos.

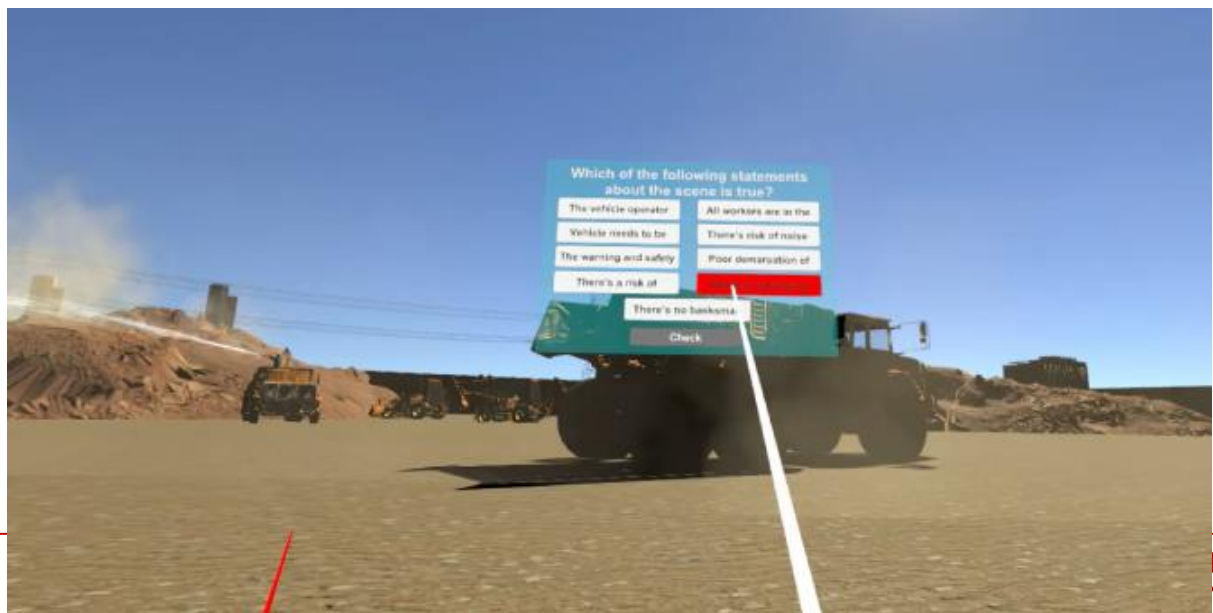


Figura 19: Imagen de interacción de rayos

6. Conclusión

El objetivo principal de este proyecto es la creación de un entorno de formación inmersivo e interactivo muy innovador basado en la tecnología de Realidad Virtual (VR) para impartir a los trabajadores de la construcción habilidades esenciales y educación para interactuar con la maquinaria y los materiales que se logran a través del desarrollo de la herramienta SafeCrobot presentada en este informe. Este informe presenta la metodología utilizada para el desarrollo de la herramienta SafeCrobot, que es una herramienta virtual inmersiva de capacitación en seguridad. La conciencia de los riesgos y las necesidades de seguridad de la automatización se está convirtiendo gradualmente en una habilidad esencial en los lugares de trabajo actuales y futuros como resultado de las complejas interacciones impuestas por las máquinas y materiales asociados en el mismo entorno de trabajo. Esta herramienta permitirá a las diferentes partes interesadas del sector de la construcción (asociaciones profesionales, sindicatos, administración) lograr la conciencia y la mejora de habilidades deseadas sobre el impacto de la robótica y la automatización en la salud, la seguridad y el medio ambiente en la industria de la construcción en toda Europa.