UNIVERSIDAD NACIONAL DE COLOMBIA
SEDE BOGOTÁ
FACULTAD DE INGENIERÍA

Proyecto de Grado: Diseño de un sistema de realidad virtual escalable para simular la utilización de un torno hibrido industrial.

Design of a scalable virtual reality system to simulate the use of an industrial hybrid lathe.

JOSE ADOLFO ROBAYO MURILLO

Universidad Nacional de Colombia

Facultad de ingeniería, Departamento de ingeniería mecánica y mecatrónica

Bogotá, Colombia

2023

Diseño de un sistema de realidad virtual escalable para simular la utilización de un torno hibrido industrial.

Design of a scalable virtual reality system to simulate the use of an industrial hybrid lathe.

Jose Adolfo Robayo Murillo

Trabajo final presentado como requisito parcial para optar al título de:

Magister en Ingeniería - Ingeniería Mecánica

Director:

Luis Miguel Méndez

Universidad Nacional de Colombia

Facultad de ingeniería del Departamento de ingeniería mecánica y mecatrónica

Bogotá D.C. 2023

Dedicatoria:

A mi familia Robayo Murillo, padre, madre y hermana
Por su apoyo incondicional. Especialmente a mi perro Dylan, siempre estuvo detrás de mi silla en el desarrollo de este proyecto, pero partió al cielo el primer mes del 2023.

Declaración de obra original

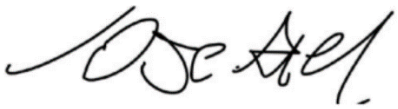
Yo declaro lo siguiente:

He leído el Acuerdo 035 de 2003 del Consejo Académico de la Universidad Nacional. «Reglamento sobre propiedad intelectual» y la Normatividad Nacional relacionada al respeto de los derechos de autor. Esta disertación representa mi trabajo original, excepto donde he reconocido las ideas, las palabras, o materiales de otros autores.

Cuando se han presentado ideas o palabras de otros autores en esta disertación, he realizado su respectivo reconocimiento aplicando correctamente los esquemas de citas y referencias bibliográficas en el estilo requerido.

He obtenido el permiso del autor o editor para incluir cualquier material con derechos de autor (por ejemplo, tablas, figuras, instrumentos de encuesta o grandes porciones de texto).

Por último, he sometido esta disertación a la herramienta de integridad académica, definida por la universidad.



_____

Nombre

Fecha 01/02/2023

Agradecimientos:

Especialmente al director de mi trabajo de grado, ingeniero Luis Miguel Méndez, por su enfoque, colaboración y confianza y permitirme tomar las riendas de uno de sus proyectos y su apoyo constante para desarrollarlo.

Resumen:

Para el proyecto de profundización se implementó un sistema de realidad virtual de un torno hibrido industrial, apoyado con sistemas de retroalimentación con el usuario; buscando generar un aprendizaje significativo de esta herramienta mecánica-industrial y a su vez disminuir los accidentes que se puedan presentar durante el uso de un torno real. Este sistema se desarrolló mediante *Unity 3D* (motor de video juegos), modelos en tres dimensiones configurados para este proceso, lenguaje de programación en *scripts* y animaciones para el usuario. La función principal es la simulación de operaciones que el usuario tiene que realizar al utilizar esta herramienta en la realidad, desglosando el proceso de utilización y los posibles accidentes que se pueden generar al errar en los pasos y obviar las normas de seguridad industrial. Todo esto da como resultado que los ingenieros y estudiantes que se sometan a la simulación adquirieran experiencia en el uso de la herramienta y estrictas medidas de prevención estando un paso adelante en comparación con quienes no hacen uso de la simulación.

Palabras clave: realidad virtual, simulación, herramientas industriales, seguridad.

Abstract:

For the deepening project the virtual reality system of an industrial hybric lathe was implemented, supported by user feedback systems; seeking to generate a significant learning of this mechanical-industrial tool and in turn reduce the accidents that could occur when using a real lathe. This system was developed using Unity 3D (video game engine), three-dimensional models configured for this process, script programming language, and user animations. The main function is the simulation of operations that the user would face when using this tool, breaking down the process of use and the possible accidents that could be generated by making mistakes in the steps and ignoring industrial safety regulations. As a result, the engineers and students who underwent the simulation gained experience in the use of the tool and strict prevention measures that were significantly relevant compared to those who did not use the simulation.

Keywords: Virtual Reality, Simulation, Industrial Tools, Security.

Contenido

Lista de ilustraciones

Lista de Tablas:

Lista de figuras

Abreviaturas

Abreviatura	Término
2D	Dos Dimensiones
3D	Tres Dimensiones.
CNC	Control Numérico Computarizado
C#	Código de Programación Orientada a Objetos
PC	Computador Personal (Ordenador)
UX	User Experience (Experiencia de Usuario)
UI	User Interface (Interfaz de Usuario)
CAD	Diseño Asistido Por Computadora
RPM	Revoluciones por minuto

CAPÍTULO 1

1 INTRODUCCIÓN

La realidad virtual es utilizada en variados entornos de entrenamiento que pueden tener un porcentaje de riesgo para el usuario en la práctica real, desde simuladores de aviación; conducción; aplicaciones en la medicina; terapias psicológicas, etc. Además, el campo de los videojuegos es de los grandes desarrolladores de realidad virtual, debido al interés de los creadores en generar un alto *marketing* dentro de los usuarios a través de las sensaciones experimentadas en los entornos virtuales de juego.

En la ingeniería, campo que compete este estudio, la experiencia en el uso de herramientas mecánicas es generalmente adquirida de forma empírica y en el sitio real, exponiendo a los usuarios a riesgos de accidentes que comprometen su integridad física. Por lo anterior, es evidente que hace falta más desarrollo en la implementación de realidad virtual para el uso de herramientas mecánicas de alta complejidad y, así mismo, aumentar la calidad de aprendizaje en los usuarios inexpertos.

Los tornos convencionales son herramientas que no han ingresado directamente a la simulación por realidad virtual. Por ejemplo, los *tornos CNC* incluyen simuladores para el aprendizaje en códigos programados sin riesgo a la integridad del usuario, pero aun así están expuestos al daño del sistema por un uso incorrecto. En el caso del torno hibrido que está compuesto por el sistema manual y el sistema de código CNC es necesario el desarrollo de un entorno virtual que permita a los usuarios aprender sobre los conceptos de torneado, mostrando los posibles accidentes que conlleva usar de manera errónea los sistemas manuales del torno y enfatizando en las recomendaciones de seguridad, como el uso de los implementos convencionales (el overol y las gafas) necesarios para este tipo de proceso en la ingeniería mecánica. [14]

1.1 Objetivo general

Diseñar e implementar un sistema de realidad virtual que permita al usuario interactuar y realizar las operaciones de mecanizado de un torno hibrido industrial, priorizando la seguridad del usuario, para ser utilizado como método pedagógico reduciendo las fallas en operación y actos inseguros cuando se manipule un torno real.

1.2 Objetivos Específicos

- Construir el entorno virtual de un torno hibrido industrial con los diferentes componentes físicos utilizados en los diversos procesos de mecanizado en un torno.

- Generar la inmersión visual del manejo de un torno hibrido en realidad virtual, que sirva para aprender el manejo adecuado y las medidas de seguridad para tener en cuenta.

- Establecer una comunicación bidireccional de variables con un sistema de interacción humano-máquina entre el usuario y el entorno virtual del torno hibrido industrial, llevando a experimentar la inmersión en los procesos realizados en este dispositivo y las posibles situaciones durante su uso.
- Aportar herramientas de aprendizaje en seguridad industrial y manejo adecuado dentro del ámbito de las máquinas y herramientas industriales que involucran algunos riesgos.

1.3 Glosario

Assets: elemento accesible gratuito o pago de elementos a incluir dentro de la interfaz *de Unity 3D*, donde se pueden encontrar modelados 3D de personajes, casas, vehículos, texturas, sonidos y *scripts* de animaciones automáticas para incluir en proyectos personales. [13]

***Colliders*: s**e incluyen en los objetos para propósitos de colisiones físicas, normalmente es invisible y el modelador debe ajustar el *Collider* de acuerdo con la forma del objeto; se puede ajustar de forma cuadrada alrededor del objeto o usarlo de malla *mesh* que se adapta a las formas irregulares del objeto, pero implica un uso más intenso para el procesador de la computadora. Se pueden adaptar a la forma de un elemento modelado en 3D o 2D, dándole la condición de poderse colisionar con otros elementos que contengan otro *Collider*, aportando este parámetro evita que dos o más elementos se traslapen entre sí. *Mesh Collider*, herramienta. [13]

Escala Unity: *Unity* no muestra unidades de medidas determinadas, pero de acuerdo con su configuración la unidad es igual a 1 metro (1 = 1 m). [13]

***Prefabs*:** conjunto de elementos de *Unity,* como elementos 3D, 2D o *Canvas* de información o códigos asignados a los objetos, generalmente conocidos como *Gameplays*, esto con el fin de reutilizarlo y organizar la información. [13]

GamePlay: elemento visualizado en las ventanas de creación de *Unity*, (cámaras, personajes, capas). [13]

GameObject: puede ser cualquier elemento que utiliza el usuario una o varias veces dentro de la creación de una aplicación o video juego, puede contener figuras 3D, *Canvas*, un generador de partículas, textos interactivos, entre otros. [13]

Canvas: es un *GameObject* dentro de *Unity* que permite almacenar elementos de UI (*User Interface*), para crear menús para el usuario. [13]

***Prefabs*:** conjunto de elementos en *Unity* que permite almacenar *GameObjects* con características correspondientes, tales como modelos 3D con propiedades de comportamiento físico o *canvas* de interfaz del usuario. [13]

Realidad aumentada: sucede cuando el usuario explora un espacio real o interactúa con un dispositivo real acompañado de un periférico que le brinda información relevante para la situación.[25]

Realidad virtual: entorno virtual que permite al usuario experimentar una experiencia controlada mediante un PC, y opcionalmente con periféricos de realidad virtual, para lograr una inmersión sensitiva, utilizada para aprender un proceso determinado y diseñado. Existen diferentes *softwares* que permiten diseñar estos entornos actualmente, entre ellos *Unity*, un *software* motor de videojuegos que es utilizado en este proyecto. [11].

Retopologia: calcar polígonos de modelos 3D de gran extensión y definición para que solo tengan polígonos de 3 o 4 vértices máximo, así aligerar el rendimiento de los procesos con estos archivos o elementos y así facilitar el trabajo de desarrollo con ellos.[23]

***Rigidbodies*:** componente principal que permite el comportamiento físico para un objeto, respondiendo inmediatamente a la gravedad del entorno de juego, mediante el código de interno de programación se rige mediante la función *Transform,* para ajustar la rotación y posición de los objetos; el objeto podrá recibir fuerza y torque. [13]

Scenes: En español traduce escena, en este caso refiriéndose a las escenas creadas dentro de los proyectos de Unity 3D, tipo de archivo que puede contener más archivos para el desarrollo de una aplicación, como modelos 3D, sonidos y animaciones. [13]

Scripts: son archivos de código que se asignan a los *GameObjects* para ser controlados de una manera específica, escritos en lenguaje de programación *UnityScripts* y C#.[13]

Software Unity: motor de videojuegos compatible con diferentes plataformas de desarrollo de software, utilizado para el entretenimiento y para el campo de la enseñanza. [12]

Textura: archivo de imagen que se añade a los objetos 3D para animarlos y dar color para la visualización del usuario. [13]

Triggers: espacio geométrico dentro del entorno virtual que cuenta con la función detección, pero no genera colisión. Funciona para utilizar la programación de alertas cuando hay interacción entre dos o más elementos dentro del entorno. [13]

Torno industrial: herramienta mecánica utilizada para el tornear de piezas cilíndricas, utilizado normalmente para las necesidades del sector industrial.[5]

CAPÍTULO 2

2 Marco teórico

2.1 Teoría y tipos de torno

El torno es una máquina que hace girar la pieza que se va a procesar mientras que otras herramientas le realizan cortes para obtener una pieza de medidas y forma específica. En la industria del torneado se presentan variedad de equipos que permiten realizar procesos con diferentes características. A continuación, se describe los tipos de torno convencional, hibrido y CNC (torno por control numérico computarizado) actuales en la industria. El uso de tornos industriales se realiza para obtener piezas especificas necesarias para un sistema, generalmente ejes o piezas circulares con los requerimientos geométricos solicitados.[5]

Los tornos industriales fueron inicialmente utilizados para mecanizar madera varios siglos atrás, pero en el siglo XVI los tornos fueron adaptados a energía hidráulica. Después en la revolución industrial, se realizó el cambio de madera hacia los metales, de esta manera se incorporaron las aleaciones metálicas, tales como la de aluminio, la de titanio, magnesio, las aleaciones de titanio, cobre y magnesio; utilizadas para diferentes aplicaciones y diseños específicos requeridos en un sistema mecánico. [9]

2.1.1 Torno mecánico convencional

Es una máquina industrial con herramientas que permiten hacer procesos de mecanizado específicos, la cual puede generar piezas con geometría de revolución de acuerdo con medidas específicas requeridas, por medio de un husillo que realiza un corte en la pieza produciendo virutas por el torneado.[1]

2.1.2 Torno CNC

Respecto a sus siglas CNC hace mención a “control numérico computarizado”; es un dispositivo de mecanizado que se compone de un sistema mecánico, uno electrónico y uno de software, el cual procesa las órdenes de ejecución contenidas en un software que previamente ha configurado un programador conocedor de la tecnología de mecanizado en tornos, siguiendo los ejes cartesianos X, Y y Z según sus grados de libertad; actuando para generar un mecanizado paramétrico requerido por un diseño técnico de base [9]. En el caso de este tipo de torno, la velocidad de giro de cabezal porta piezas, el avance de los carros longitudinal y transversal, las cotas de ejecución de la pieza están programadas y, por tanto, exentas de fallos imputables al operario de la máquina. Las rutinas más básicas dentro del torno industrial son el código G00 que significa “posicionamiento inicial” (sin maquinar); G01 que significa “Interpolación lineal (Maquinando); G74 que significa “Taladrado Intermitente con salida para virutas”; G28 “volver al home” [14]

2.1.3 Torno hibrido industrial.

Un torno hibrido industrial es un sistema mecánico que se compone de las características de un torno mecánico convencional y también posee propiedades de instrucciones automatizadas de un centro de control numérico (CNC), el cual permite generar un código de programación de proceso para obtener un diseño especifico de pieza.[14]

La definición de la palabra híbrido surge de la mezcla de dos sistemas donde se puede referir a dos tecnologías en una, donde el los movimientos que hace el operario, como un movimiento lineal, un contacto de mecanizado, un desbastado, son identificados por códigos específicos y almacenados en un archivo para reutilizar un comportamiento.

2.2 Partes del torno: [4]

Las partes más comunes que componen un torno hibrido son el plato-husillo, torre porta herramientas, carro transversal, carro auxiliar, carro principal y bancada.[5]

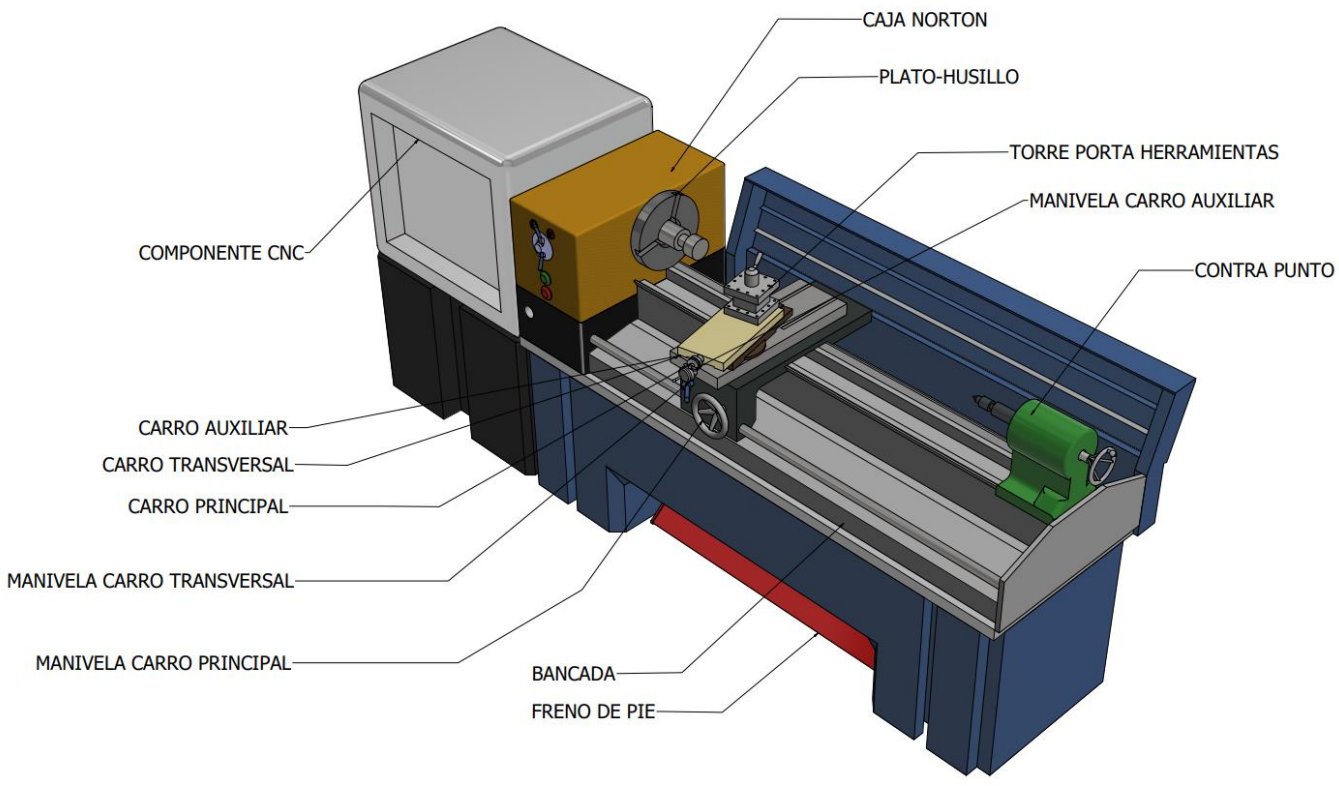


Ilustración 1: partes generales de torno industrial

2.2.1 Caja Norton:

Es el espacio donde se ubican sistemas de transmisión enfocadas al husillo y plato porta herramientas que permite al usuario activar diversas velocidades de avance de acuerdo con la marca y el tipo de aplicación del torno, accionadas mediante palancas con placas indicadoras del cambio.[9]

2.2.2 Plato husillo:

El plato de sujeción permite acomodar la materia prima a tornear mediante mordazas de sujeción, las cuales se ajustan manualmente por medio de una llave operada por el usuario. [9]

2.2.3 Torreta porta herramientas:

Ensamble de piezas que permite ajustar los buriles o demás herramientas de corte mediante la sujeción por medio de tornillos, y de acuerdo con esta configuración y al ángulo de la torreta, se pueden variar entre los procesos de torneado cilíndrico, cónico, ranurado, avellanado entre otros. [26]

2.2.4 Contra punto:

Herramienta que se ubica linealmente al plato-husillo, se puede desplazar a lo largo de la bancada, se aplica generalmente para fijar la pieza a tornear y evitar vibraciones por el corte de los buriles. [9]

2.2.5 Freno de pie:

Permite desencadenar los mecanismos del torno, promoviendo una parada inminente. Solo algunas marcas cuentan con este pedal. [9]

2.2.6 Bancada:

Es el apoyo principal del torno donde se ubican las herramientas principales como el contra punto, el carro longitudinal; generalmente sobre esta bancada se encuentran guías prismáticas o colas de milano, de acuerdo con la marca, las cuales brindan el apoyo y deslizamiento de los demás sistemas. [9]

2.2.7 Carro principal:

También es mencionado como carro longitudinal y este transporta el carro transversal y la torreta porta herramientas. [9]

2.2.8 Carro transversal:

Este carro se ubica sobre el carro principal y se desplaza transversalmente a este o respecto a la bancada, cumple con dar profundidad a los cortes y se manipula manualmente por una manivela propia. [9]

2.2.9 Carro auxiliar:

Este se ubica sobre el carro transversal, su eje de libertad es girar bajo su eje horizontal, utilizado para posicionar la torreta porta herramienta y así los ángulos de las herramientas de corte; aunque la torreta también es ajustable. [9].

2.3 Entorno en laboratorios o talleres de mecanizado:

Los centros de mecanizado pueden tener procesos peligrosos debido a las partes giratorias, proyección de partículas y fragmentos (virutas, fragmentos de pieza y/o herramienta, etc.), atrapamiento de extremidades como la mano o el brazo por los componentes del sistema, alta tensión, ruido o vibraciones y aire comprimido. Se deben seguir una serie de precauciones básicas de seguridad cuando se utilice este tipo de máquina, reduciendo el riesgo de daño personal y mecánico. [24]

En los espacios donde se ubican las máquinas de procesos industriales como torneado y fresado es necesario incluir un área remarcada para trabajo, el almacenamiento de piezas y el sector de soldadura [7]. Debe hacerse la lubricación necesaria de las piezas durante el proceso, debe contar con salidas y señalización de evacuación, botiquín para accidentes, alarmas y excelente iluminación constante en el horario de trabajo con las máquinas. Por parte del usuario, es importante que mantenga el orden de los procedimientos en el montaje, procesado y desmontaje de cada pieza mecanizada, por último, el orden en el taller ayudara a prevenir accidentes y deslizamientos del usuario en el suelo. [24]

2.4 Herramientas de corte

Se conoce como herramientas de corte a todas aquellas piezas que funcionan a través de arranque de viruta de un material en específico, esto quiere decir que las herramientas de corte son todas aquellas herramientas que permitan arrancar, cortar o dividir algo a través del filo de la herramienta. En este caso, para efectos de la simulación, se utilizará el buril dado que es la herramienta de corte más implementada en la industria. [27]

2.4.1 Buril y aplicaciones

El buril es la pieza final de torneado en diferentes procesos, conllevan ángulos de afilado que permiten adecuar el tipo de torneado que se va a realizar, de acuerdo con las propiedades de la materia prima a mecanizar (tornear) se debe escoger el material del buril. Los buriles son herramientas de corte para torno utilizadas principalmente para cortar, marcar, ranurar o desbastar. En la siguiente imagen se muestra las partes de un buril, en este caso es un buril tipo HSS, hecho para metales blandos.[22]

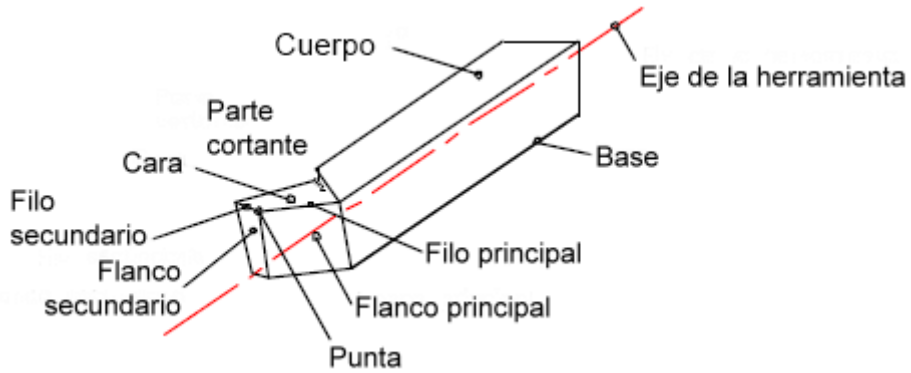


Ilustración 2 Partes de un buril [22]

El buril está compuesto básicamente por un cuerpo, base y un cabezal donde se encuentra la parte cortante de metal templado o acero rápido, según su función. Los buriles placa soldada están conformados por un cuerpo prismático de acero y la parte cortante es un componente incrustado de HSS o carbono de tungsteno en forma de pastilla, como se ve en la ilustración a continuación. [22]

Ilustración 3 Tipos de Buril [28]

2.4.2 Tipos de buril

Esta herramienta se clasifica de acuerdo con el proceso que se vaya a realizar, entre los cuales se encuentran

Tipo buril /proceso	**Características**	**Tipos**
Desbaste	Retira grandes cantidades de material para posteriormente pulir con otros tipos de puntas.	**Buril de desbaste recto**: este buril tiene un eje recto. **Buril de desbaste acodado**: en este buril el eje se curva hacia la derecha o la izquierda, hacia la parte cortante del mismo.
Cilindrar y refrentar	Hace formas cilíndricas y refrentado (es el proceso mediante el cual se pule la pieza hasta que quede perpendicular al eje de giro)	**Los buriles de cilindro:** se bajan para generar un ángulo adecuado que permita realizar las formas cilíndricas en el material. **Los buriles de refrentado**: este buril se baja desde el extremo para que haga un ángulo de 90° respecto al eje simétrico.
Roscado	Realiza una rosca en el elemento en el que se está trabajando	**Internos**: se usan para realizar las roscas en la parte internas en el material. **Externos**: se usan para realizar las roscas en la parte externas en el material.

Tabla 1 Procesos de Buril

2.4.3 Materiales de buril

En la industria es importante que los buriles presenten alta dureza, gran ductilidad y resistencia al desgaste y altas temperaturas, con el fin de que se reduzca la posibilidad de dañar la herramienta. Algunos de los materiales

utilizados en la fabricación de estos buriles son el cromo, el vanadio, el tungsteno, cobalto, entre otros; de acuerdo con la dureza de cada uno se selecciona el buril para tornear la pieza.[3]

Ilustración 4 Catalogo de ferretería JRC [20]

Material buril	**Características**	**Usos**
Acero rápido	Son herramientas de acero aleado con elementos como el tungsteno, cromo vanadio, entre otros. Este acero adquiere resistencia a altas temperaturas (600°C) y al desgaste.	En metales blandos o trabajos de baja producción, porque la única manera de afilarlas de nuevo es en amoladoras o esmeriladoras que son de uso común en un taller.
Metal duro	Se fabrican a partir de polvo de carburo y una porción de cobalto que se utiliza como aglomerante, esto le da una resistencia hasta 850 °C. Los componentes principales de un metal duro son el volframio y el molibdeno, además del cobalto y el carbono	Maquinar con hierro colado, metales no ferrosos y algunos materiales abrasivos no metálicos.
Cermet	Se hacen con materiales cerámicos y metal, el aglomerante que usan es níquel-cobalto	Gran resistencia a altas temperaturas y para trabajar materiales que producen viruta dúctil.
Cerámica	Mantienen la dureza en altas temperaturas y no reacciona químicamente con los materiales de la pieza, sin embargo, son puntas bastante	En producción en serie.

Material buril	Características	Usos
	frágiles. Los materiales cerámicos más comunes se basan en alúmina (óxido de aluminio), nitruro de silicio y carburo de silicio.	
Diamante policristalino (PCD)	Este diamante es sintético y casi alcanza la dureza del diamante natural, es resistente al desgaste y bajas temperaturas térmicas.	Es frágil por lo tanto las temperaturas no deben exceder los 600°C y no se debe usar en materiales ferrosos.

Tabla 2: materiales de buril [22,26,27]

2.4.4 Angulo de buril

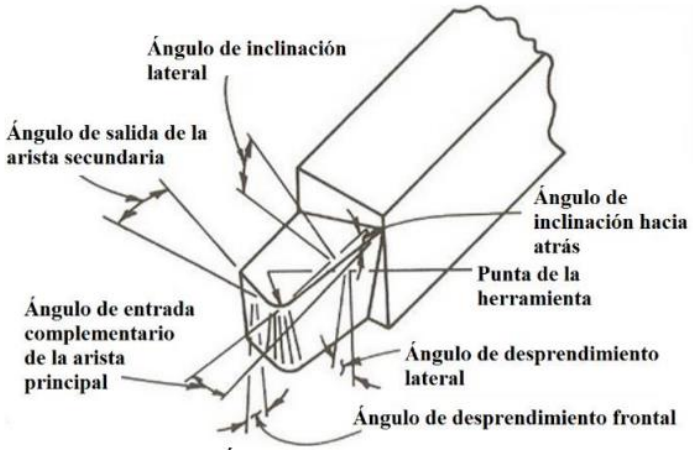


Ilustración 5 Ángulos de buril [22]

Los buriles en el torneado tienen 6 ángulos que son ángulo de inclinación lateral, ángulo de desprendimiento lateral, ángulo de desprendimiento frontal, ángulo de inclinación hacia atrás, ángulo de entrada complementario de la arista principal y ángulo de salida de la arista secundaria. En la siguiente tabla, se detallan algunos de los parámetros de la geometría de corte de los filos de una plaquita de torneado, y en la ilustración se observan algunos ejemplos de estas configuraciones.

Tipo de ángulo	Características
Ángulo de posición	La forma en que se posiciona el filo de corte respecto de la pieza. Este ángulo se normaliza en la codificación ISO de porta plaquitas. Este ángulo afecta a la longitud de filo implicado en el corte. Los valores más usuales son de 45º a 95º, tomando valores menores para desbastes pequeños y grandes para torneados en general.
Ángulo de corte recto	Es el formado por la punta de placa, entre el filo de corte principal y el filo de corte secundario. Para ángulos de corte más agudos la punta tiende a ser más frágil.

Tipo de ángulo	Características
Ángulo de posición secundario	Es el ángulo formado entre el filo de corte y la dirección de avance de la herramienta (f), de tal manera que con los ángulos de posición y corte recto suma 180°. Este ángulo limita la inclinación de perfiles en los que el diámetro de la pieza disminuye.
Ángulo de inclinación	Este ángulo viene dado por el alojamiento de la porta plaquitas, en el que se inserta la plaquita. Es el ángulo formado entre el eje perpendicular al avance de la herramienta en la punta de plaquita y la cara superior de la misma.
Ángulo de corte	Es el ángulo formado entre el plano de la cara superior de la plaquita y su lado lateral en la punta de corte. Según el ángulo de corte las placas se clasifican en neutras (de 90º), positivas (menores de 90º) o placas extrapositivas (si son de 73º o inferiores). Cuanto menor es este ángulo, más frágil es el filo, sin embargo, lo hace más cortante.
Ángulo de incidencia	Es el ángulo entre la vertical en la punta de la herramienta y la faceta lateral de la plaquita. En placas con ángulo de corte de 90º (neutras) es necesario un ángulo de inclinación (I) negativo
Ángulo de desprendimiento	El ángulo de desprendimiento está formado por un eje paralelo al avance de la herramienta colocado en la punta de la placa, con la superficie superior de la placa.

2.5 Virutas de material

El proceso de arranque de la viruta se realiza mediante la penetración de una herramienta de corte en el material, realizando un movimiento relativo entre la pieza que se desea tornear y la herramienta, dando lugar a un desperdicio llamado viruta. [30]

Teniendo en cuenta las velocidades y el material se clasifica el tipo de viruta en:

- Viruta continua
- Viruta segmentada
- Viruta discontinua

La viruta continua suele aparecer cuando el material torneado es dúctil, como pueden ser los hierros forjados, aceros suaves y aluminios. Para conseguir este tipo de viruta la formación se produce en la zona primaria de deformación,

arrancando el material por capas que deslizan por la zona secundaria de deformación, es decir, por la cara pegada a la herramienta. Las condiciones que se deben dar para que este tipo de viruta se genere son básicamente que se aplique la fuerza suficiente en la interfaz herramienta-material de tal manera que se consiga esta separación del material.[30]

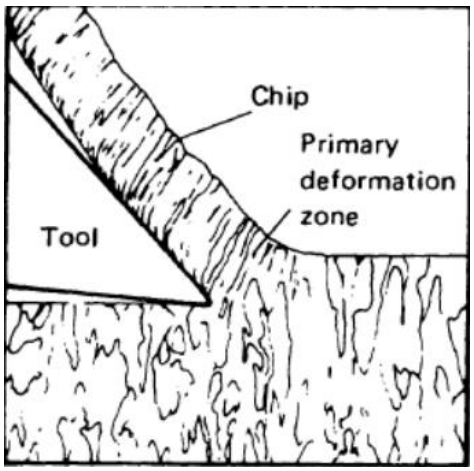


Ilustración *6 Viruta Continua [30]*

En la viruta discontinua el material se somete a fuertes tensiones, lo que provoca la ruptura del material en la zona primaria de deformación. Este tipo de viruta se da en materiales frágiles o en materiales dúctiles, como fundiciones del hierro o latón, cuando las condiciones de corte son a baja velocidad de corte y un avance considerable. [8] En la siguiente imagen se contempla una representación gráfica del tipo de viruta discontinua tipo “Chip” [30]

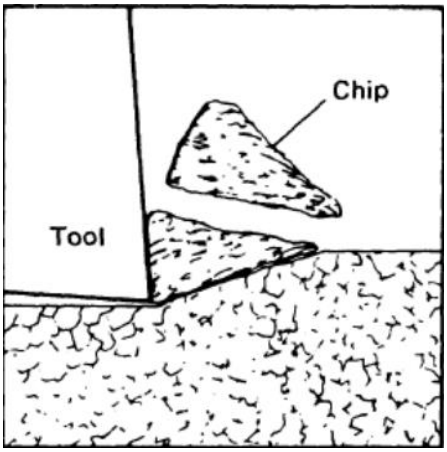


Ilustración *7 Viruta Discontinua [30]*

Las virutas segmentadas se consideran semicontinuas y se producen en materiales con baja conductividad térmica, con una resistencia que disminuye considerablemente con la temperatura, tal como sucede con el titanio. La gran peculiaridad de estas virutas reside en que presentan zonas de alta y baja deformación provocada por el fenómeno de cizalladura.[8] A continuación se observa una imagen de un ejemplo de simulación del desprendimiento de viruta, la cual puede cambiar con respecto a la velocidad de corte y al tipo del material de la pieza mecanizada y el buril,[30]

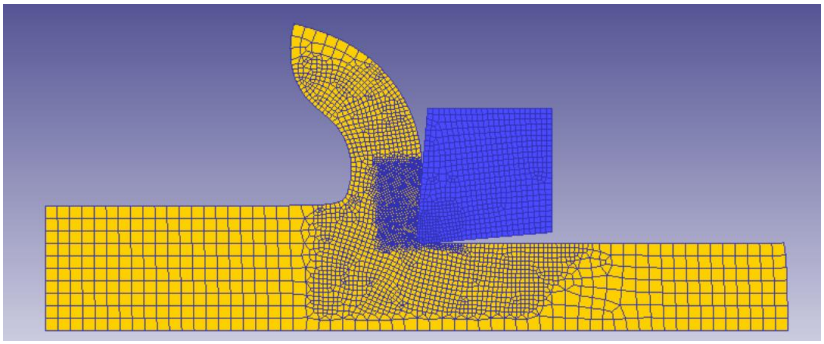


Ilustración 8 *Simulación de Virutas [30]*

Según los materiales a tornear proporcionan una morfología de virutas en función del avance y la profundidad del buril, de acuerdo con la velocidad de avance, el diámetro de la pieza, el tipo de buril puede provocar que las virutas sean más continuas o discontinuas ya que su morfología será distinta. En la siguiente Ilustración se puede observar las diferentes morfologías en relación con la profundidad y el avance de corte en el torno:

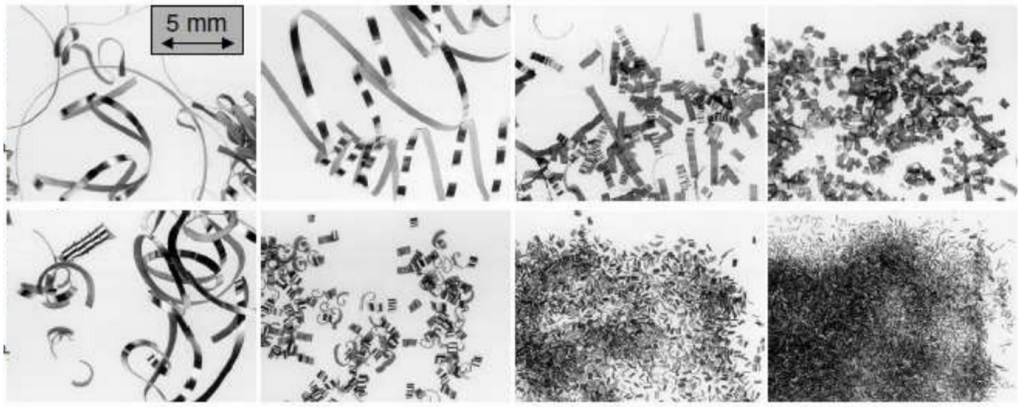


Ilustración 9 *Morfología de la viruta en función de la profundidad y del avance del corte [30]*

2.6 Materia Prima para tornear

La materia prima para tornear debe tener una figura cilíndrica para poder ser instalado en el plato-husillo, de esta manera se podrá tornear a través de las revoluciones por minuto del torno industrial de acuerdo con las características físico-eléctricas.[19] Los tipos de materiales a tornear depende de las aplicaciones a utilizar y del material de la herramienta de corte, en la siguiente tabla se detallan:

Material	**Características**	**Uso**
Metálicos	Los metales y sus aleaciones como también sustancias inorgánicas que están constituidas por uno o más elementos metálicos; por ejemplo: hierro cobre, aluminio, níquel y titanio.	
Metálicos Ferrosos	El más usado es el hierro, dado que las técnicas de extracción y los procesos de obtención del metal son relativamente económicos. Minerales	El **hierro** presenta algunos inconvenientes, se corroe con facilidad, tiene un punto de fusión elevado y es de difícil tornear, resulta frágil y quebradizo.

Material	Características	Uso
	que contienen mucho hierro: la magnetita, la hematites, la limonita y la siderita. El **acero** es una aleación del hierro con una pequeña cantidad de carbono. De este modo se obtienen materiales de elevada dureza y tenacidad y con una mayor resistencia a la tracción. Los aceros pueden contener otros elementos químicos, a fin de mejorar propiedades específicas; se obtienen así los **aceros aleados** que son: Silicio, Manganeso, Cromo, Níquel y Wolframio.	Se emplea en componentes eléctricos y electrónicos. Dentro de los elementos de máquinas que se construyen con materiales metálicos como el **acero** o las fundiciones, como ejes, poleas, carcasas, tornillos, piñones, bujes, rodamientos, entre otros.
Metálicos no ferrosos	Se utilizan otros muchos materiales metálicos **no procedentes del hierro** en la fabricación de elementos de máquinas. **Cobre**: se obtiene a partir de los minerales cuprita, calcopirita y malaquita. Presenta una alta conductividad eléctrica y térmica, así como una notable maleabilidad y ductilidad **Latón**: es una aleación de cobre y zinc. Presenta una alta resistencia a la corrosión y soporta el agua y el vapor de agua mejor que el cobre. **Bronce**: Es una aleación de cobre y estaño. Este metal presenta una elevada ductilidad y una buena resistencia al desgaste y a la corrosión. **Aluminio**: se obtiene de la bauxita, un mineral muy escaso. Presenta una alta resistencia a la corrosión. Es muy blando, de baja densidad y gran maleabilidad y ductilidad. Presenta una alta conductividad eléctrica y térmica.	Algunos de los elementos de máquinas que se fabrican con estos materiales son: piñones y bujes de bronce, piñones de aluminio, entre otros.
Poliméricos	De origen tanto natural como sintético, formados por moléculas de gran tamaño, conocidas como macromoléculas. Polímeros de origen natural son, por ejemplo, la celulosa, el caucho natural y las proteínas. Los poliésteres, poliamidas, son grupos de polímeros sintéticos	Elementos estructurales de las maquinas como, por ejemplo, empaques y sellos de caucho.

Material	Características	Uso
	con una composición química similar dentro de cada grupo.	

Tabla 3 Algunos tipos de materiales a tornear.[27]

El tipo de material a tornear varia en la clase de tornos a utilizar y la aplicación que requiera. En la siguiente tabla se puede observar los materiales más usados en la industria, su aplicación y propiedades:

Material	Propiedades	Aplicación
Madera	La madera es aislante térmico y eléctrico. Es un material renovable, biodegradable y reciclable. Es dúctil, maleable y tenaz.	Materia prima de origen vegetal más explotada por el hombre. Se pueden ver en juguetes o piezas hechas de madera.
Aleaciones de cobre (latón, bronce)	En estado natural del cobre tiene dureza 3 en la escala de Mohs (50 en la escala de Vickers).	Alta capacidad de conducción térmica para la construcción de intercambiadores de calor, construcción de tuberías sin soldadura, para el transporte de fluidos, para suministros de agua.
Aleaciones de Aluminio	La aleación de aluminio permite: - Aumentar la resistencia mecánica. -Aumentar la resistencia a la corrosión. -Aumentar la resistencia al desgaste. - Aumentar la dureza. - Disminuir la ductilidad.	El aluminio tiene unas excelentes características de conductividad térmica, lo cual es una importante ventaja, dado que permite que el calor generado en el torneado se disipe con rapidez. Sus aplicaciones son de tipo universal, como por ejemplo la fabricación de vajilla metálica, electrodomésticos, artillería, entre otros.
Aleaciones de magnesio	La aleación de magnesio permite: - Aumentar la resistencia a la tracción. - Aumentar la dureza. - Inhibir el crecimiento de grano. - Aumentar la cantidad de hierro que se puede disolver. - Disminuir la ductilidad.	Son innumerables las aplicaciones actuales de aleaciones de magnesio, incluyendo: palos de golf, raquetas, bicicletas, patines, motociclismo y elementos de protección para deportistas. Es el principal material para latas de bebidas.

Material	**Propiedades**	**Aplicación**
	Las aleaciones de magnesio se caracterizan por tener alta fluidez, bajo calor específico y baja tendencia a adherirse a las paredes del molde de acero.	
Polímeros	Los polímeros son materiales ligeros y baratos, relativamente blandos, elásticos y resistentes a la corrosión, pero no suelen trabajar bien a temperaturas medias y elevadas.	Se pueden usar para reforzar, aislar, espesar.
Acero Toolox 44	-Dureza: 45 HRC -Fácil de tornear -Posee una gran tenacidad a los impactos y sus tensiones residuales son muy bajas, a fin de proporcionar una buena estabilidad dimensional.	Adecuado para el uso en matrices de extrusión y componentes de ingeniería que requieren gran resistencia. Es apropiado para estampas y herramientas de moldeado de plástico y goma, y para colada por presión, así como para herramientas de doblar y conformar chapa.
Acero al carbono	El aumento del contenido de carbono en el acero eleva su resistencia a la tracción, incrementa el índice de fragilidad en frío y hace que disminuya la tenacidad y la ductilidad.	Es utilizado para fabricar carrocería de autos, máquinas, tuberías, cascos de buques, piezas de maquinaria, clavos, cerraduras, alfileres, motores, ferrocarriles, entre otros muchos usos.
Cerámicos	Normalmente están compuestos por materiales en estado puro, dopados con elementos que les dan las características deseadas.	Se usa para implantes ortopédicos, aislante eléctrico, motores de combustión.
Bronce	-Su fragilidad es mayor que la del cobre y tiene menos puntos de fusión. -Es hasta un 10% más pesado que el acero. -Es muy resistente a la corrosión.	Se utiliza sobre todo en recipientes de presión, tanques, para conductos hidráulicos que soportan presión y en general para la construcción marina.

Material	Propiedades	Aplicación
	-Resulta un excelente conductor de electricidad y calor. -No se generan chispas al golpearlo. -Es muy versátil a nivel físico, químico y mecánico.	

Tabla 4 materia prima a tornear [27, 33]

2.7 Velocidad de corte

La velocidad de corte es la velocidad relativa entre la pieza y la herramienta. Más específicamente es la velocidad relativa entre la arista de corte de la herramienta y la superficie a tornear de la pieza. La velocidad de torneado para el husillo se selecciona de acuerdo con el tipo de material que se va a tornear, el diámetro de la pieza, el tipo de viruta, el tipo de torneado a realizar, la temperatura de las piezas al tener contacto en revolución entre ellas y la seguridad para el operario.[8] Esta velocidad de corte depende directamente del diámetro de la pieza a tornear en relación con las revoluciones rpm con las que está funcionando al sistema.

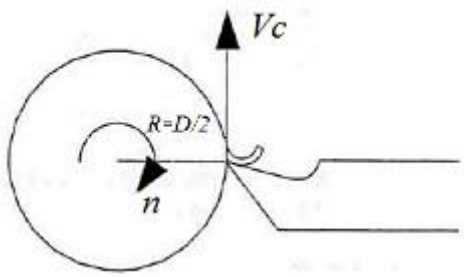


Ilustración 10 Velocidad de corte en torneado [21]

Este dato se puede encontrar en los manuales de usuario o catálogos de las herramientas del fabricante. A continuación, se observa las velocidades de corte de unos materiales:

Material a Tornear	Herramienta de acero rápido	Herramienta de carburo
Acero (resistente)	15 - 18	60 - 70
Acero dulce	30 - 38	110 – 140
Fundición (media)	18 - 24	70 – 85
Bronce	24 - 45	
Latón	45 - 60	
Aluminio	75 - 400	150 – 1000
Titanio	30	60 – 70

Tabla 5 Velocidad de corte para algunos materiales (m/min)[17]

La velocidad de corte depende en gran parte de las máquinas de corte empleadas, de la rigidez de las herramientas de torneado y del tipo de líquidos refrigerantes y lubricantes utilizados. Los paneles de configuración de velocidad en tornos industriales se caracterizan por sus velocidades altas y bajas, esto dependiendo del tipo de tornado que se requiera. En la siguiente ilustración se muestra la fotografía de un torno industrial en uso, allí se puede apreciar el sistema de palancas empleadas para esta configuración; la palanca de la parte superior es para seleccionar la cantidad de revoluciones por minuto y la segunda palanca en la parte inferior es para seleccionar velocidades altas o bajas. La cantidad de opciones de velocidad dependerá del fabricante y la aplicación del torno industrial.[8]

Ilustración 11 Caja Norton torno, Universidad Nacional de Colombia

Además, se observa en el panel de mando de un torno la opción de funcionamiento manual o automático. El funcionamiento manual se refiere a que el usuario controla los movimientos de los ejes y además debe tener conocimiento de las dimensiones de la pieza a realizar. Por otro lado, el funcionamiento automático se refiere al uso de un *software* para la generación de un programa de la pieza que se desea y, posteriormente, el uso de un torno CNC para la implementación del programa y la fabricación de la pieza. En este caso, los ejes se mueven con autonomía de acuerdo con las medidas y forma del programa realizado. [8]

2.8 Avance de Corte

Cuando el buril pasa a una mayor velocidad de corte torneando la pieza para una distancia especifica de un corte de viruta el espesor de la viruta será mucho más pequeño, aunque su longitud generalmente es mucho mayor. Aunque estas características de viruta también dependen de la posición y filos del buril y significativamente del tipo de material que se este torneando. Esta velocidad dependerá también rigurosamente del movimiento manual por el usuario con las manivelas correspondientes, o el avance automático determinado si se cuenta con el en el torno a usar.[30]

2.9 Profundidad de Corte

La profundidad del corte se rige por los movimientos de las perillas del carro transversal y el carro longitudinal, generalmente de acuerdo con el tipo de material se debe tener precaución para realizar el tipo de avance para evitar

el rompimiento del buril o de la misma pieza, lo más importante es tener verificar esta profundidad respecto a una vuelta o a cierto número de vueltas dependiendo de la aplicación.

2.10 Procesos de torneado (Mecanizado)

Para los procesos de torneado se debe asegurar el correcto agarre de la pieza. A continuación, se describen los procesos más comunes como lo son cilindrado, cilindrado cónico, punteado y roscado. [9]

2.10.1 Cilindrado

El proceso de cilindrado consiste en disminuir el diámetro inicial de la pieza cilíndrica o aumentar diámetros interiores. En la siguiente ilustración se muestra el proceso de cilindrado. Cada paso que se realiza debe hacerse en un rango cercano a 0,5 mm aproximadamente, ya que realizarlo a un tamaño mayor a 1.5mm podría ocasionar la ruptura del material o de la herramienta de corte (buril). El paso es la distancia que la herramienta de corte se acerca al centro de la pieza y desbasta. [9]

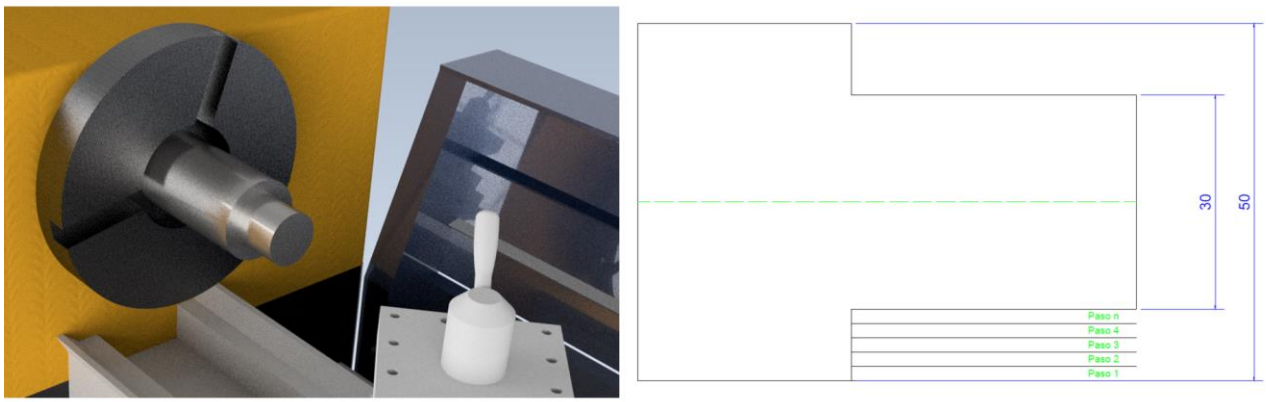


Ilustración 12 Proceso de Cilindrado [9]

2.10.2 Cilindrado cónico

El cilindrado cónico, también llamado torneado cónico, se realiza con un avance de torneado constante configurado con un ángulo especifico, donde el movimiento no será paralelo al eje longitudinal del torno, es decir debe ser diagonal o un movimiento inclinado, se debe realizar progresivamente para evitar romper el buril o el recalentamiento de las herramientas. Como se muestra en la siguiente ilustración, la cuchilla de la herramienta de corte no se encuentra paralela al eje de giro de la pieza, tiene un ángulo de corte especifico de acuerdo con el requerimiento diferente a 0 o a 90 grados, de esta manera se aseguran las piezas con formas cónicas.

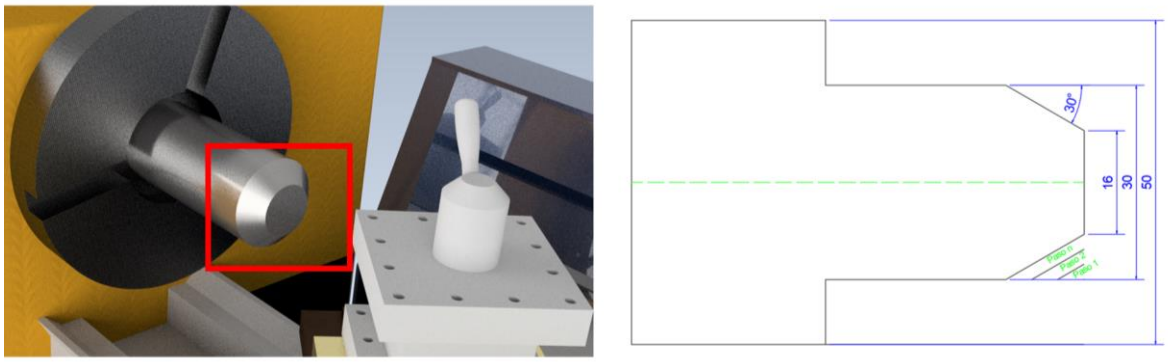


Ilustración 13 proceso de cilindrado cónico en torno [9]

2.10.3 Agujero de centros (centro punteado) y taladrado

El centro punteado es el proceso hacer un orificio en el punto más central de la cara de la pieza, inicia con la pieza a procesar en rotación, se aproxima una broca con el contra punto, hasta un tercio de la zona cónica. Con este proceso cada pieza estará preparada para iniciar la operación de taladrado, la cual se realiza por medio de una broca específica para este fin. Como se puede observar en la siguiente ilustración, el punteado se realiza en el centro de la pieza cilíndrica, de esta manera asegura el centro en el proceso de taladrado.

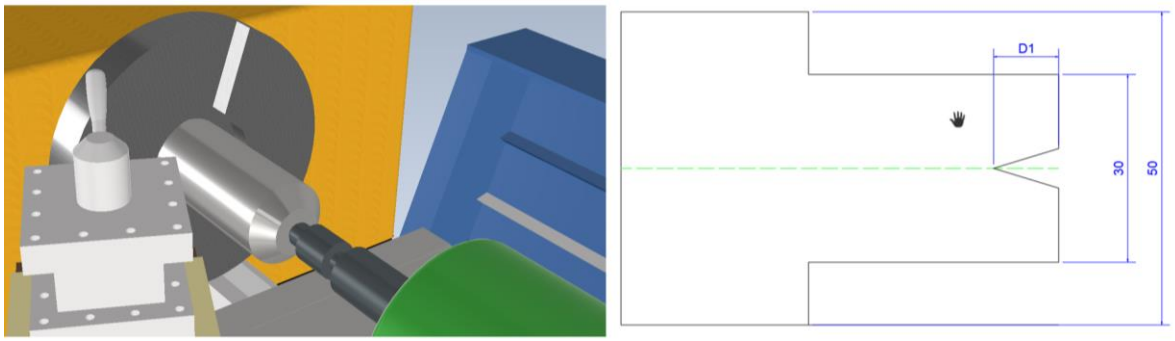


Ilustración 14 Proceso de contra punto (punteado)

2.10.4 Refrentado

Se utiliza para tornear la parte frontal o planos perpendiculares de la pieza a procesar, la herramienta de corte se desplaza de forma transversal y su objetivo comúnmente es aplanar la cara perpendicular de la pieza a las imperfecciones del corte que haya sufrido.

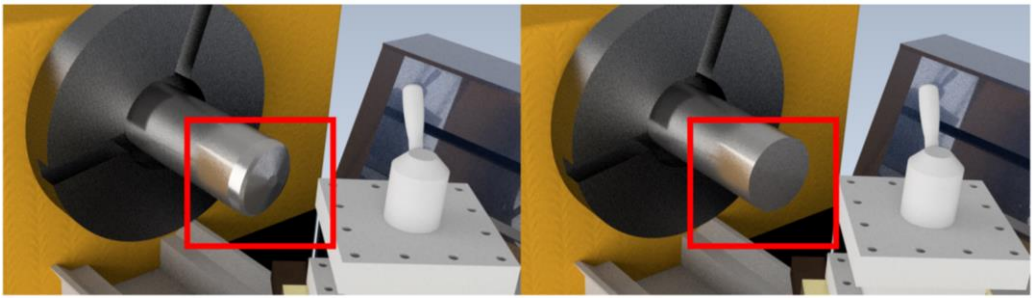


Ilustración 15 Proceso de Refrentado

2.10.5 Ranurado

Este tipo de torneado consiste en realizar ranuras perpendiculares en la pieza de acuerdo con la anchura y profundidad requerida, la pieza de corte que generalmente es un buril con una configuración de corte perpendicular para este fin, hace una penetración a la pieza en revolución de forma perpendicular al eje de la pieza, y el buril debe de tener la forma requerida para este fin, para realizar una ranura de mayor distancia que el buril este debe siempre penetrar en el material en dirección hacia el eje de la pieza.

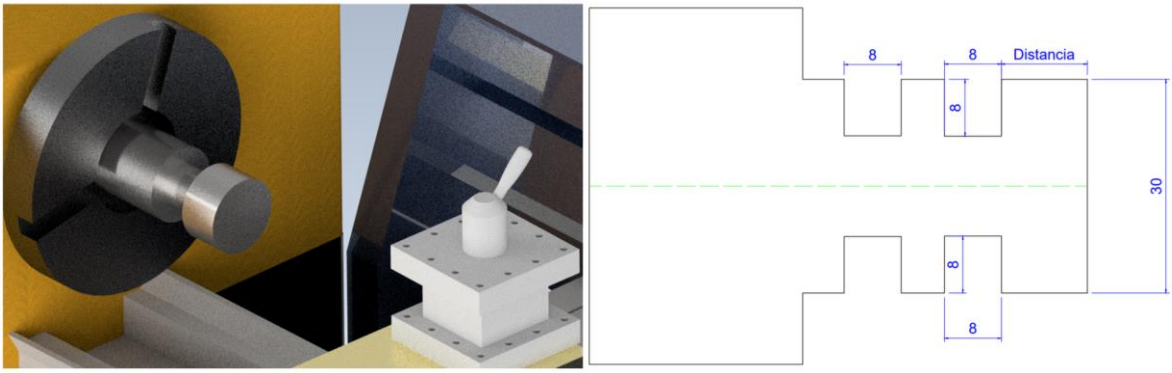


Ilustración 16. Ranurado

2.11 Seguridad Industrial

Los procesos industriales realizados por un operador requieren un protocolo de seguridad que el usuario de determinada maquinaria debe conocer para evitar accidentes que comprometan su integridad. Los accidentes se generan por condiciones inseguras o prácticas incorrectas del usuario. Diferentes aspectos como la falta de señalización, falta de mantenimiento preventivo, malos procedimientos en las operaciones o falta de comunicación entre trabajadores son algunos de los factores que ocasionan estos accidentes. [2]

Diferentes circunstancias se pueden presentar a los trabajadores y operarios en los espacios laborales, siempre estarán expuestos a peligros como temperaturas extremas, cambios bruscos de presión, riesgo químico o riesgo a elementos cortantes; para prevenir accidentes el usuario debe conocer previamente este tipo de situaciones y haber tenido una capacitación previa al respecto, para esto la realidad virtual es capaz de realizar una inmersión que les permite conocer la situación de manera controlada sin el riesgo latente.[6]

Los accidentes más comunes en un torno industrial son:

- Saltos de material al rostro (Corte por viruta, quemado por viruta o golpe por herramienta arrojada.)
- Cortes, aplastamiento y lesiones de extremidades del usuario por componentes de la máquina.
- Lesión por atoramiento de las manos o brazos del usuario.
- Accidentes eléctricos por corto o mal uso.

Teniendo en cuenta estos accidentes es vital el uso de materiales básicos para protección en el torno:

- Atención completa del usuario para el procesamiento a realizar con la herramienta.
- Evitar las distracciones del entorno, verificar que el espacio de ubicación del torno sea propicio para las operaciones del plan de trabajo.
- Gafas de protección.
- Calzado antideslizante con punta metálica.
- No utilizar guantes ni ropa de mangas largas
- Tener el cabello largo recogido tanto en hombres como en mujeres
- No tener accesorios de joyería entre ellos, manillas o pulseras, anillos, pendientes.
- Evitar que herramientas de ajuste no implicadas en las revoluciones de la caja Norton estén sobre el mismo al momento de funcionar.

2.11.1 Señalización de seguridad

Se deben seguir una serie de precauciones básicas de seguridad cuando se utilice este tipo de máquina, para reducir el riesgo de daño personal y mecánico. En la siguiente ilustración se observa la etiqueta de advertencia general del torno:

Ilustración 17 Señales de peligro en taller industrial [28]

Ilustración *18 Señalización en taller Industrial A [28]*

2.11.2 Seguridad en el simulador:

Se muestra al usuario las señalizaciones que debe tener el taller donde se encuentra el torno industrial a trabajar, también debe verificar la indumentaria adecuada para poder utilizar el torno industrial como el overol o vestimenta segura para el torno, las gafas de seguridad que deben ser transparentes, sin ningún tipo de filtro de sol, que principalmente proteja los ojos de los operadores del torno.[16]

El cabello de las mujeres u hombre de cabello largo debe estar recogido para evitar que se enrede con los instrumentos del sistema mecánico, con el fin de evitar accidentes y lesiones para el usuario.[16]

CAPÍTULO 3

3 Desarrollo en Unity

3.1 Antecedentes de realidad virtual de torno

De acuerdo a las estructuras de investigación de la Universidad Nacional de Colombia, se realizan con anterioridad dos proyectos de grado, los cuales implementan una conexión entre un sistema prototipo de volantes hápticos (físicos, tangible) y una conexión por medio de código ROS, al desarrollador de videojuegos *Unity 3D*, permitiendo la interoperabilidad del usuario con un entorno virtual y un entorno real, logrando que el usuario interactúe con los movimientos del carro longitudinal y transversal de un torno industrial.[10]

La compañía *Beitxusudios* generó un simulador de realidad virtual que permite el uso de un torno CNC con una gamificación que aporta al usuario la información sobre el correcto uso de la herramienta mecánica y el monte de los materiales a tornear.[18]

3.2 Metodología

La plataforma de realidad virtual para el aprendizaje del manejo de un torno hibrido industrial se desarrolla por medio de la investigación de procesos industriales de torneado, factores de seguridad al utilizar las herramientas industriales y desarrollo de la plataforma interactiva por medio del motor de video juegos *Unity 3D.*[12]

Al usuario se muestra los procesos de torneado, protocolos industriales y de seguridad implicados dentro de un taller de metalmecánica realizados en un torno hibrido industrial a través de la plataforma de realidad virtual del presente proyecto; mostrando 3 de los procesos más conocidos de modo que se les permite comprender el proceso de cilindrado, ranurado, roscado y torneado cónico.

El desarrollo se realiza mediante el motor de video juegos *Unity 3D*, programado mediante Código C# donde se deben modelar (diseñados) en *software CAD* los elementos industriales físicos para ser exportados al *software Unity* y ahí asignarles la programación orientada a objetos que permita ser interactiva con el usuario, mostrando las diferentes componentes del torno industrial, las medidas de seguridad y los procesos de torneado. [12]

3.2.1 Diseño de prototipo: el diseño del prototipo de realidad virtual

La forma en la que se realiza esta realidad virtual se inicia por la construcción de archivos *CAD* paramétricos de los elementos que componen un torno industrial del torno industrial, en este caso la caja Norton, los carros transversal y longitudinal, la torreta porta herramientas, el buril, entre otros, al igual que la planeación del espacio o taller de metalmecánica mediante *AutoCAD.* [26]

Después de ser migrados a *Unity*, donde a algunos de los archivos se le asignan funciones mediante las herramientas del motor de video juegos y códigos C#, se remodelan elementos del torno con las herramientas 3D de *Unity* para facilitar la compatibilidad con los *scripts* de programación y comportamiento de estos. Posteriormente se exporta un archivo ejecutable que permite al usuario navegar por la realidad virtual desde cualquier ordenador. Es posible utilizar periféricos RV como los visores que se ajustan alrededor de la cabeza del usuario o, en este caso, perillas que simulen el movimiento de las perillas del torno. [12]

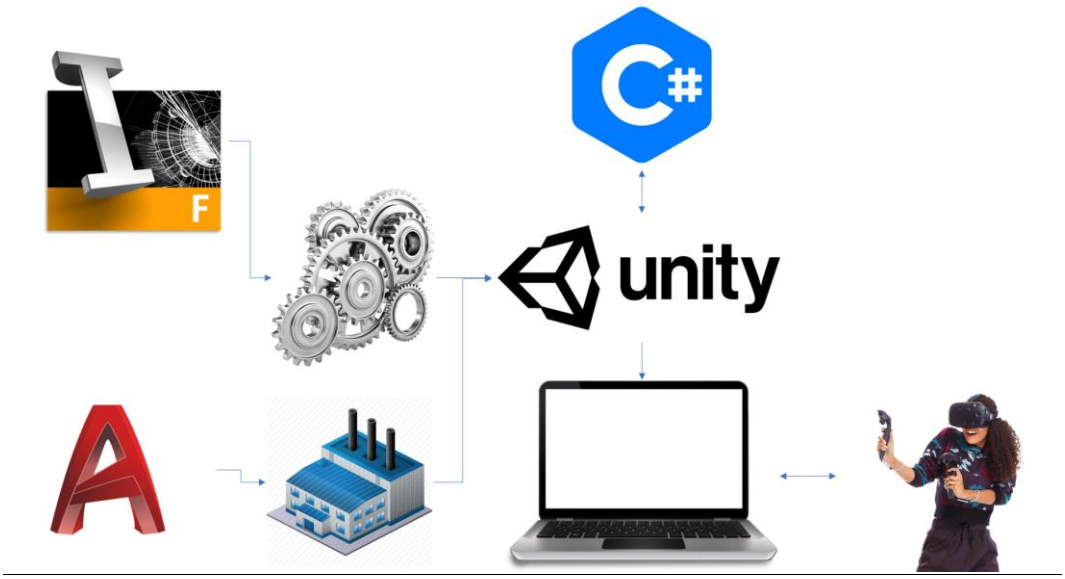


Ilustración 19 Estructura Metodológica General de Realidad Virtual.

La metodología para preparar la realidad virtual también se basa en la investigación de las partes del torno y los procesos de mecanizado o torneado en este caso; para este caso se realizaron los procesos de cilindrado, torneado cónico, taladrado y ranurado, donde el usuario puede simular estos procesos mediante una pieza estándar por medio de los archivos 3D modelados; aunque para llegar al prototipo final se realizaron varias pruebas de usuario verificando que se logre lo esperado en los objetivos de investigación. Para todo ello se recopilan los procesos de construcción, las experiencias del usuario y los diseños de interfaz en el presente documento.

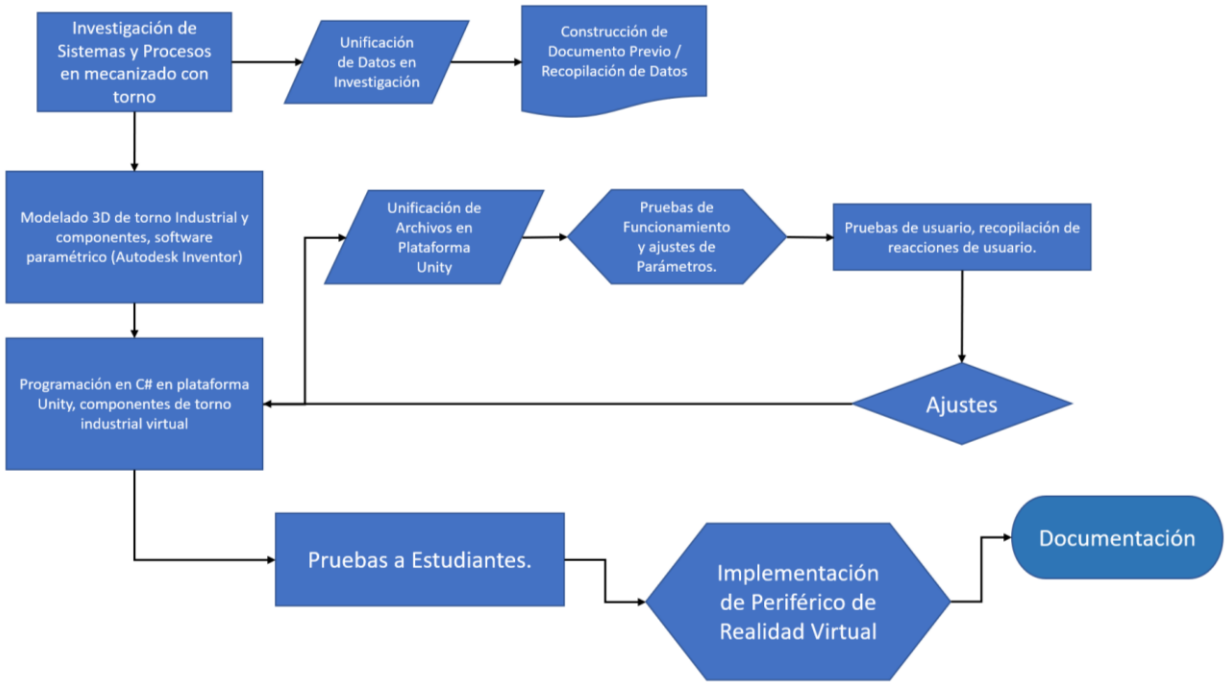


Ilustración 20 Metodología de Construcción

3.2.2 Creación del Archivo Unity.

El *software* de desarrollo de videojuegos empleado para implementar este proyecto en *Unity 3D*, es multiplataforma y se puede implementar en video juegos de tres y dos dimensiones al igual que aplicaciones de realidad virtual y realidad aumentada. Se descarga de manera gratuita o paga y se puede instalar en diferentes actualizaciones de acuerdo con la época, este proyecto se realizó con la versión 2020.42fl. La instalación automáticamente se realiza junto con el ***Unity Hub*** que es un gestor de versiones e instalación de complementos y el ***Visual Studio*** el cual permite escribir los códigos de programación orientada a objetos, en este caso en lenguaje *C sharp* (C#)***;*** algunos complementos son requeridos de acuerdo con actualizaciones automáticas del sistema, plataformas especificas o uso de periféricos de realidad virtual o aumentada. Estos *softwares* tienen soporte en su página web, como tutoriales y ejemplos, normalmente se genera una sesión mediante una inscripción con correo electrónico. Terminado el registro se puede iniciar abriendo el archivo nuevo de proyecto, así como se ve en la siguiente ilustración. [13]

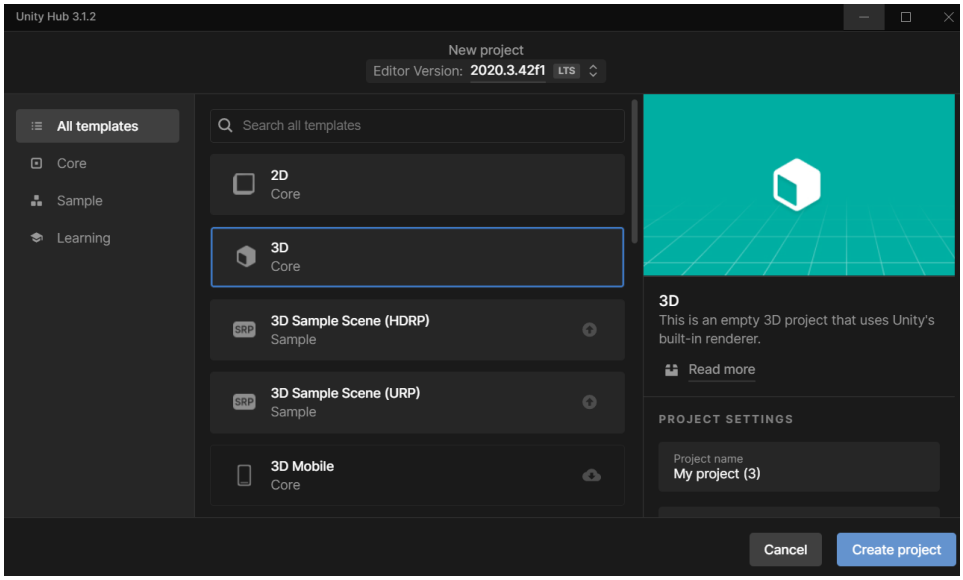


Ilustración 21 Ventana inicial de Unity

3.2.3 *Main* de estructuración de archivos empleados.

La estructura de archivos empleada para la simulación del torno industrial consta de diferentes archivos, ubicados en la jerarquía de *Unity*, entre los utilizados para generar este proyecto de simulación son los archivos 3D importados de otros software y generados en la misma interfaz, sonidos, *Canvas* compuestos por *Sprites* y archivos de imagen, paquetes (Packages), utilizados por el usuario y prediseñados para configuraciones específicas como el comportamiento de cámaras, ambientación y plantillas de código, en la siguiente imagen encontramos el detalle de esta jerarquía. [13]

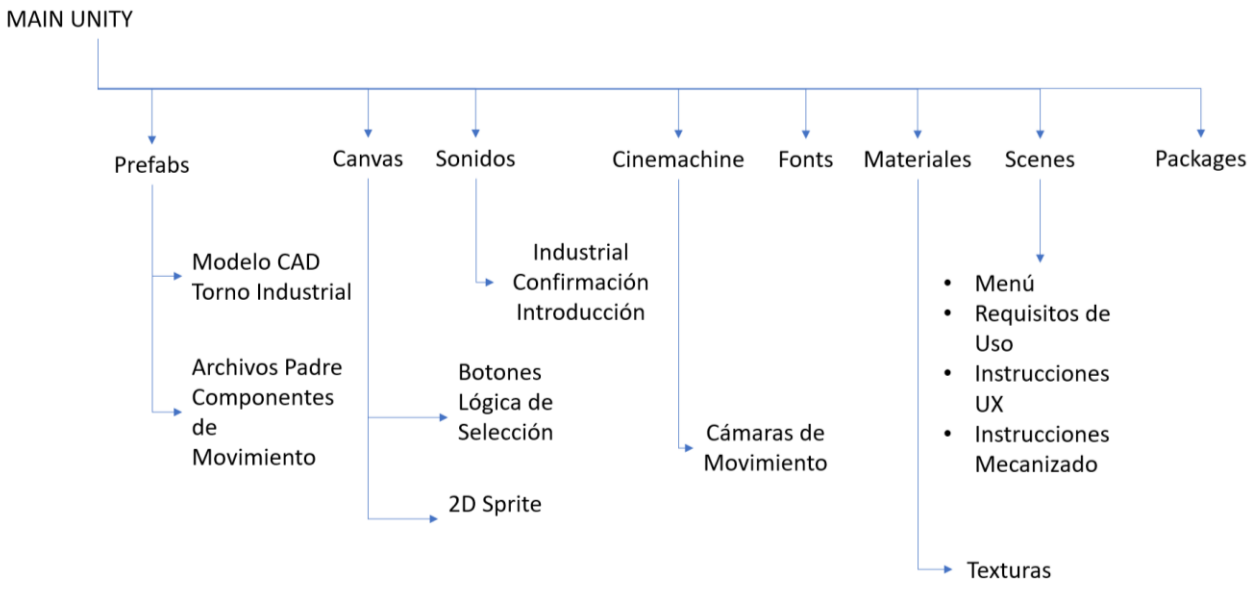


Ilustración 22 Estructura Main de Unity

3.2.4 Modelación de entorno virtual

Ensamble de piezas

La interoperabilidad entre el *software Autodesk Inventor* y *Unity* no soporta ciertos parámetros, debido a que en *Unity* no son reconocidos las uniones y posiciones entre piezas de los ensambles y subensambles que se construyen dentro del *software Autodesk*; por consiguiente, en *Unity* se deben crear nuevamente las respectivas relaciones entre elementos, en este caso, piezas o herramientas mecánicas del torno industrial. Piezas como el carro transversal y carro auxiliar se remodelan para ajustar con los movimientos programados por código. [11]

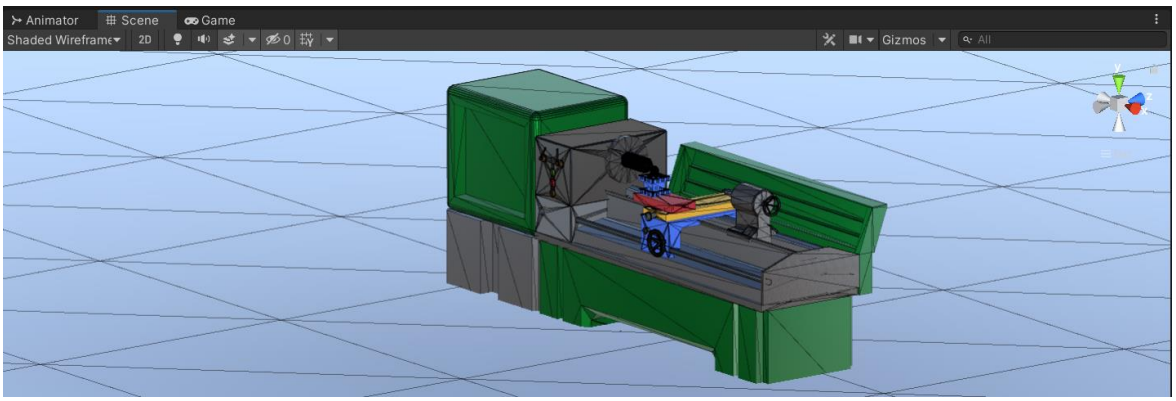


Ilustración 23. Ensamble de Torno, Importado desde Autodesk Inventor.

De acuerdo con la metodología propuesta se realiza el modelamiento 3D del torno híbrido con los componentes de ambiente del taller industrial donde se ubicará el torno híbrido, espacio adecuado para el correcto uso de la herramienta, el almacenamiento de las herramientas de montaje, gabinetes de la materia prima y lugar de la indumentaria necesaria para los procesos.[15] En la siguiente ilustración se muestra un ensamble en *Unity* donde la pieza “Carro Principal” es padre de las piezas “Carro Transversal”, “Carro Auxiliar”, “Carro Giro” y “Torreta Porta Herramientas”, ya que estos últimos son arrastrados al archivo padre, de esta forma quedan anclados y los movimientos dependerán de los pivotes correspondientes.

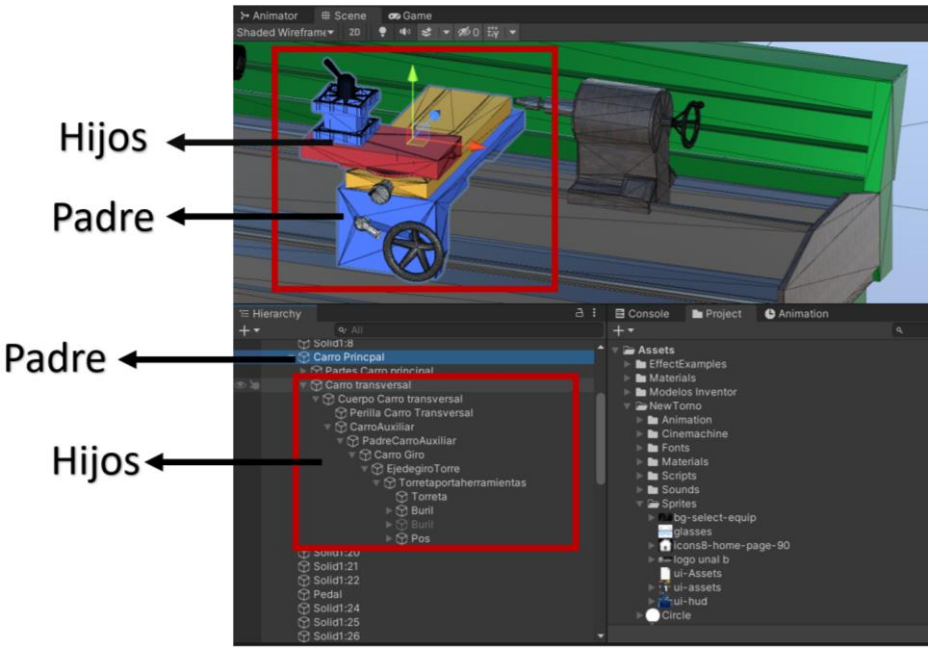


Ilustración 24 Construcción de GamePlays (Prefabs)

Estos elementos son almacenados en la carpeta *Prefabs* del proyecto de torno, almacenados en *Gameobjects* en este caso como "Carro Transversal", "Carro Principal", "Carro Longitudinal", "Contrapunto" y entre otros elementos almacenados para la construcción de la escena de *Unity* (*scene*); se almacenan de esta forma para la eventualidad de reutilizarse durante la construcción y organización de las piezas del *software CAD*; a los anteriormente mencionados se les asigna un código *script* que permite tener un movimiento específico, accionados por teclas y barras de movimiento programadas para este fin. [10]

3.3 Programación de objetos:

El algoritmo principal del programa consiste en la simulación de los procesos de torneado, en este caso es posible emplear el refrentado, cilindrado, cilindrado cónico y taladrado. Para este propósito el principal objetivo es lograr la simulación de desbaste de un objeto 3D a medida que hace contacto con el buril mientras la pieza está en revolución a una velocidad indicada por el sistema.

La interacción de los objetos 3D se realiza mediante elementos de colisión caracterizados en *Unity* como *Rigidbodies* y bloques de colisión denominadas *Colliders,* en la siguiente ilustración se puede apreciar la caja de propiedades de estos componentes donde se pueden modificar características de acuerdo con el comportamiento deseado del sistema. Se configura la masa del objeto en el factor *Mass*, comportamientos ante el viento con las variables *Drag y Angular Drag* y la configuración de las físicas por medio del uso de la gravedad o la cinemática de los movimientos.

Se observa también cuando un *script* esta incorporado en determinada pieza, en la ilustración a continuación se muestra el "*BurilController*" asignado a la pieza 3D modelada para representar el buril; además, se activan

generadores de partículas para chispas y virutas, y los parámetros para que el buril haga contacto con la pieza, detectando los factores y límites para continuar mecanizando o causar su ruptura. Este script es detallado en anexos.

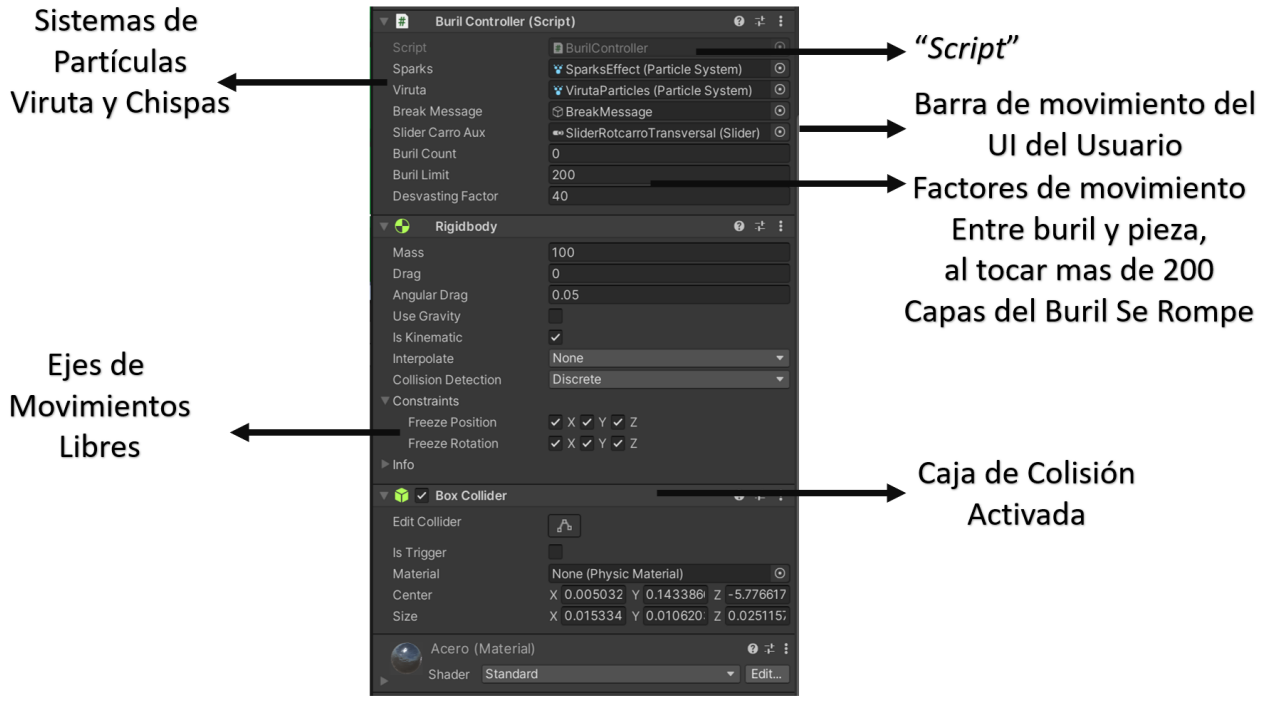


Ilustración 25 Caja de Propiedades de Box Collider y RigidBody

Se utilizan los métodos *OntriggerStay* y *OnTriggerExit* para detectar los elementos 3D entre sí, para ello deben poseer cajas de colisión y cuerpos rígidos, y así ocasionar eventos o acciones según la lógica programada, estos códigos y sentencias se detallan en los Anexos.

3.3.1 Arquitectura del programa (Códigos C#)

El código se organiza en clases, que a su vez tienen métodos y variables. Las clases cumplen una función específica de esta manera, el detalle de estos códigos se puede observar en los anexos de este documento.

- *SceneAdmin:* controla el cambio entre escenas permitiendo a los botones navegar por la aplicación
- *TornoControlManager:* controla el movimiento de los carros del torno vinculando las barras del UI con la posición en la escena 3D.
- *BurilDragController:* controla la posición del buril en la torreta según el movimiento realizado en los botones del UI
- *ContrapuntoController:* controla el movimiento del contrapunto vinculando las barras del UI con la posición en la escena 3D.
- *BurilController:* controla el movimiento vinculando las barras del UI con la posición en la escena 3D.

- *MaterialSetter:* permite cambiar el material de desbaste y del buril según las 3 posibilidades configuradas en el UI.
- *TornoSpeedUiControl:* controla la velocidad de giro del cabezal del torno en rpm, según la elección del usuario en el *UI*.
- *ParticlesSound:* controla la emisión y sonidos de las partículas para los efectos visuales de la viruta.
- *SeguridadIndustrialManager:* controla el minijuego para seleccionar los elementos de seguridad adecuados a la hora de trabajar en el torno.

3.3.2 Modificación del objeto 3D para simular el desbaste del material

Para simular el desbaste del material se modifica su escala cuando el objeto 3D interactúa con buril asignado, de esta forma el objeto se hace más pequeño. El objeto 3D a su vez está compuesto de cientos de anillos que modifican su escala localmente según sean tocados por el buril. Este componente se ubica en el plato junto al husillo, donde convencionalmente se ubicaría; en este caso aparece automáticamente con la opción de seleccionar el material a tornear de la pieza cilíndrica, en la siguiente ilustración se puede apreciar esta ubicación.

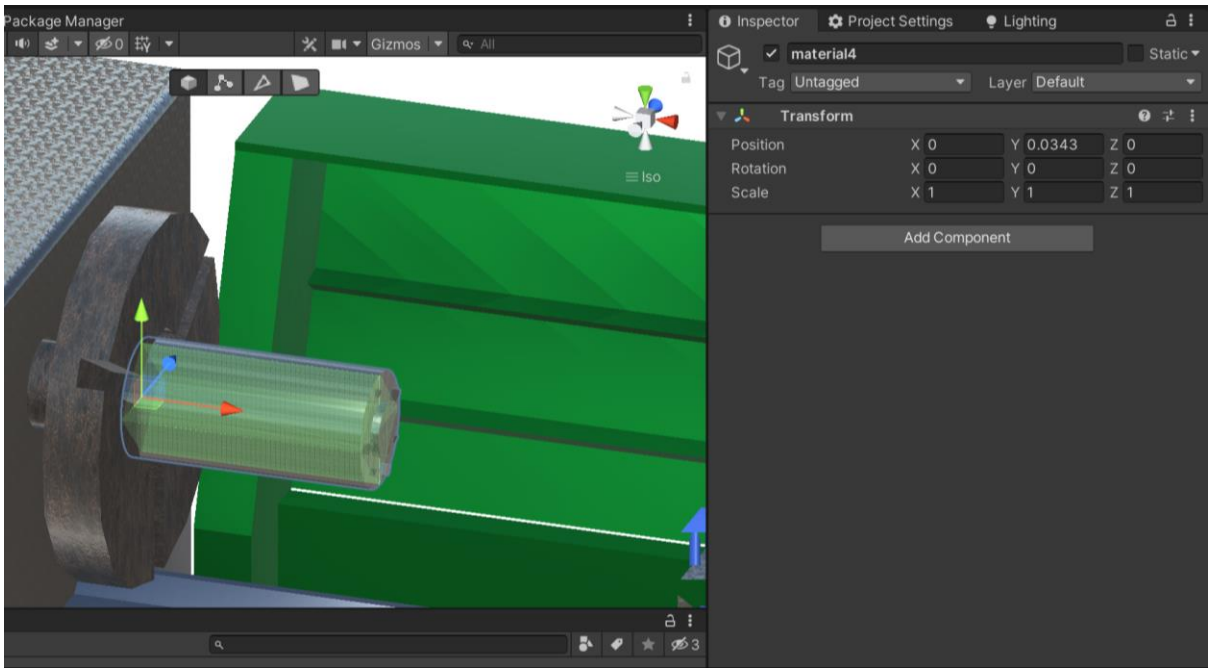


Ilustración 26 Materia prima en el plato de torneado.

3.3.3 Efecto de viruta

Cuando el buril y el material a tornear tienen contacto en esta simulación, estos emiten una acción de desprendimiento de virutas, lo cual se realiza mediante el sistema de partículas del motor de videojuegos, emitiendo mallas 3D o imágenes desde un punto en el espacio y con una periodicidad en el tiempo si hay contacto. Estas mallas también poseen propiedades físicas como gravedad, velocidad y rotación, permitiendo mostrar al usuario el efecto visual de esta acción.[13]

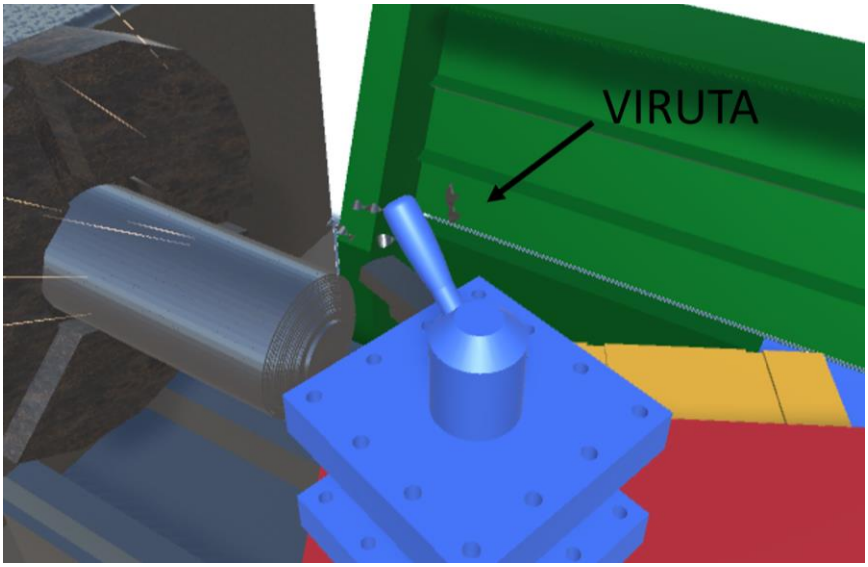


Ilustración 27 Efecto de viruta

3.3.4 Movimientos mecánicos de las partes del torno

Los objetos 3D dentro del motor de videojuegos poseen un componente básico llamado *transform.* Este componente determina la posición, rotación y escala de un objeto 3D en la *scene*, los movimientos que mayormente interactúan en la *scene* se crean mediante las mismas herramientas de *Unity*, a pesar de que la mayoría de las piezas son de origen de la modelación paramétrica 3D del *software Autodesk Inventor*, se reemplazan por modelos nativos hechos en *Unity* los que implican mayor relación con la programación C#.

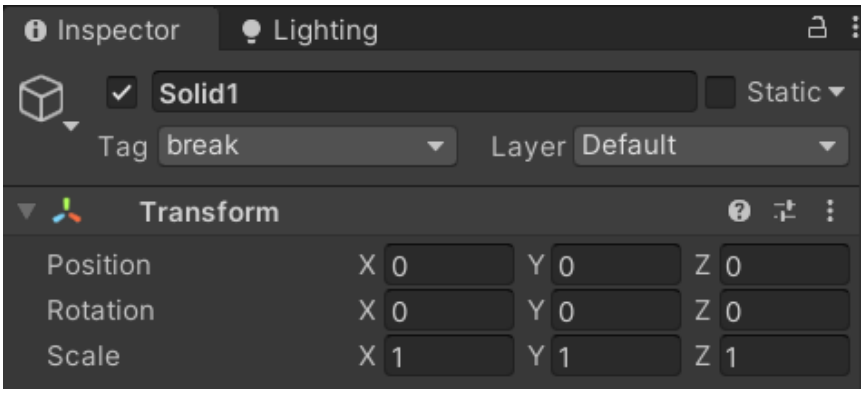


Ilustración 28 Variables de Posicion, rotacion y escala.[13]

Por medio del código es posible manipular estas propiedades para mover, rotar o escalar los objetos en la escena y así generar movimientos específicos.

Esta técnica fue utilizada para simular el movimiento de los carros alterando su posición y rotación, cambiando sus ángulos mediante “cuaterniones”. Estos valores se guardan en vectores de 3 elementos llamados “vector3” en *Unity*, son variables que al guardar 3 valores se utilizan para almacenar coordenadas en x, y, z.

3.3.5 Velocidades de Torneado

Se programan velocidades aleatorias, donde se clasifican en dos secciones altas y bajas, entre las altas se asignan 460, 755, 1200, 2000 rpm; en las velocidades bajas se asignan 70, 100, 115, 300 revoluciones por minuto (rpm); tanto altas como bajas son seleccionadas de un promedio de visto de diferentes catálogos[#]; el usuario puede cambiar la configuración de las velocidades solamente dando un clic en la palanca de selección entre altas y bajas y la palanca de selección de revoluciones por minuto disponible. Estas se inician mediante el pulsador verde de inicio en la interfaz del usuario (UI), y se detienen con el pulsador rojo de "stop" o el paro de emergencia, en la siguiente ilustración se observa esta distribución. El detalle de la codificación para este funcionamiento se observa en el Anexo *"TornoSpeedUiControl"* del código C#.

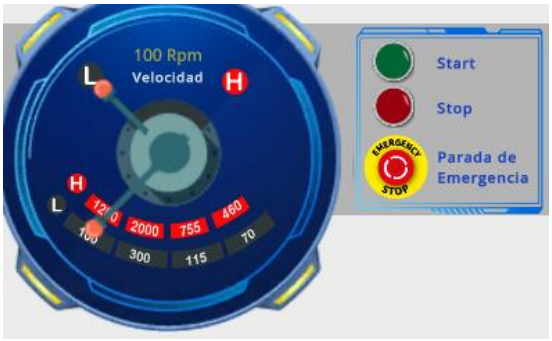


Ilustración 29 Seleccionador de Caja de Velocidades, inicio y paros de las revoluciones por min.

3.3.6 Materiales y texturas

Por medio del componente *MeshRenderer* es posible manipular las texturas de un material para simular distintos materiales de desbaste, como el aluminio, el hierro o el acero.

3.3.7 (*Chunks*) Técnica descartada del desarrollo.

Los *Chunks* traducido como "pedazos" en español. El término *chunk* se utiliza en el desarrollo de videojuegos para hacer referencia a la división en fragmentos de un objeto 3D, los cuales se desprenden para simular la destrucción de dicho objeto.

Esta técnica se investigó e implementó para simular la destrucción del material cuando el buril lo tocaba. Sin embargo, para que la simulación sea realista, los pedazos deben ser muy pequeños, lo que afecta enormemente el rendimiento de la aplicación bajando los cuadros por segundo (*FPS*) a un rango de 5 a 10, cuando se recomienda 30 o más para una experiencia fluida, en la siguiente ilustración se muestra la colisión entre los dos elementos con físicas respectivamente por este método de *Chunks*, aunque no fue el definitivo dentro de la simulación, por la razón anteriormente mencionada.[13]

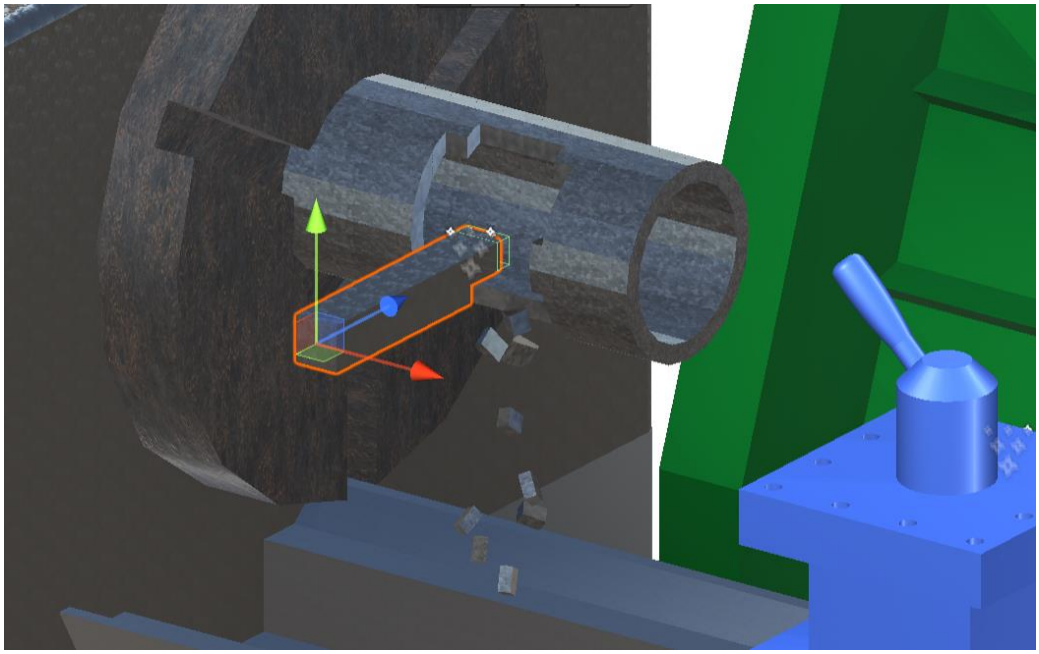


Ilustración 30 Desvanecimiento por "Chunk"

3.3.1 Efecto de torneado por escalamiento.

Para el obtener el efecto de torneado en realidad la pieza cilíndrica no es un sólido completo, sino que está conformado por cientos de anillos juntos los cuales forman la pieza a tornear. Estos anillos tienen la programación para que cuando el buril los toque estos empiecen a reducirse simulando el torneado real del sistema de torneado, con las condiciones de que si el buril supera los 200 anillos contactados mediante los *collider*, el buril se romperá, de lo contrario si es menos de esta cantidad, los anillos reducirán progresivamente su dimensión en el eje y, este evento se programa de manera que pueda efectuarse por el eje “x” y “y” así como se ve en la siguiente ilustración, este método permite simular los 5 tipos de torneado propuestos para este proyecto.

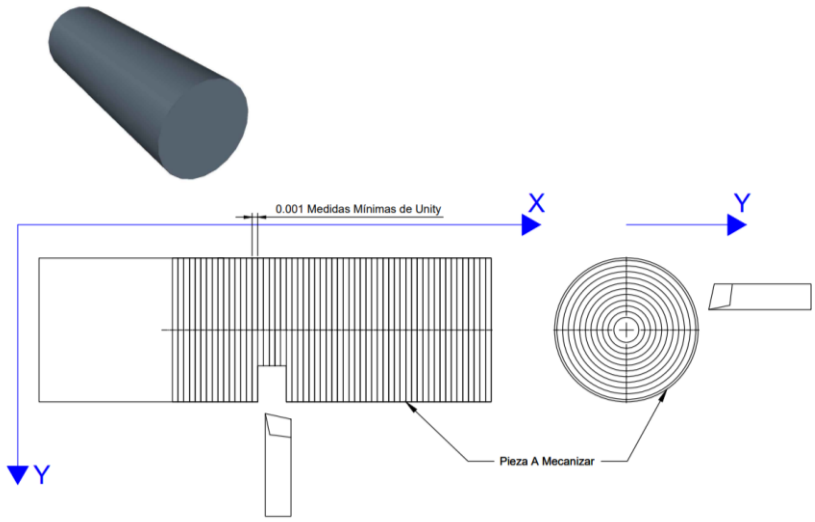


Ilustración 31 Geometría de anillos que componen el cilindro de pieza a tornear, desbastándose desde la parte exterior hacia el eje central.

3.4 Interfaz del usuario (desarrollo)

Para la interfaz de usuario el primer proceso que se realiza es la arquitectura de la información, en donde se planean las pantallas que tendrá la aplicación y cómo se conectan entre ellas. De esta forma se establece el flujo del usuario, que consiste en la forma en que el usuario interactuara por las distintas opciones buscando siempre que no tenga ningún bloqueo y pueda entrar y salir de todas las pantallas.

La arquitectura de la información se representa por medio de un diagrama de flujo, que servirá posteriormente para generar el diseño gráfico de la aplicación.

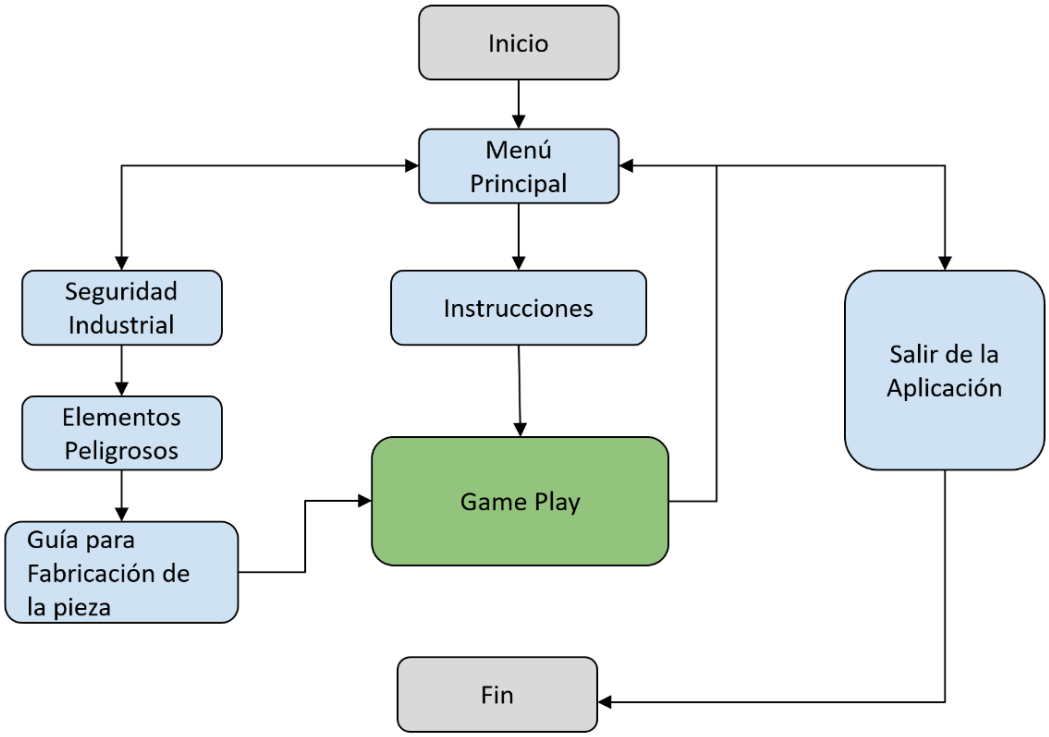


Ilustración 32 Diagrama de proceso de interfaz

3.4.1 Menú principal

El menú principal de la simulación de realidad virtual del torno hibrido industrial se crea mediante un *canvas* generado en *Unity* con enlaces para la interacción del usuario que permite navegar por las escenas e interactuar con el torno hibrido industrial.

La UI (interfaz del usuario) se construye con una temática industrial, donde el usuario puede interactuar entre las 5 escenas del simulador, encontrando botones y barra de manejo, En cuanto al color se utilizan tonos azules y grises, alusivos a los sitios industriales físicos, y el amarillo y naranja utilizados para contrastar y relacionar con la señalización de las normas de seguridad en los talleres industriales.

Para la configuración de los botones se utiliza la herramienta de *Unity* llamada *Layer*, la cual nos permite animar los botones al momento de pasar el puntero por ellos, al oprimirlo o soltarlos, los cuales se componen de unas

condiciones estandarizadas para aplicar en cada botón, permitiendo estar en diferentes estados de visualización, dependiente del puntero que controla el usuario, y así permitir pasar por diferentes estados como “Normal”, “Highlighted”, “Pressed” “Selected” y “Disable”, funciones que se aplican a los pulsadores de cada escena (scene), comportamiento que también depende de los parámetros asignados, en la siguiente ilustración se observa el esquema de configuración estandarizado en *Unity*.

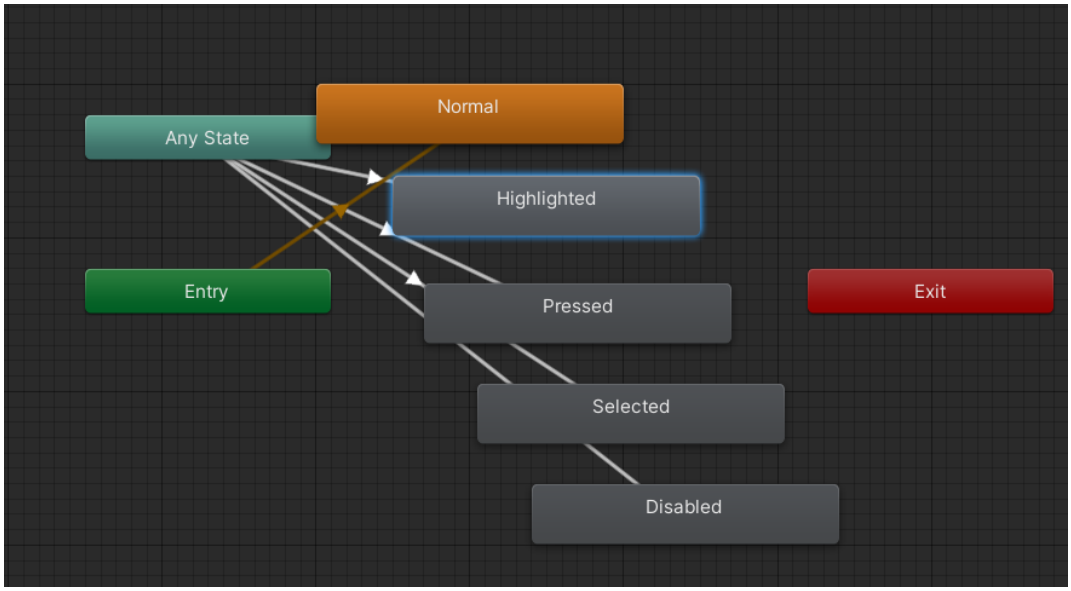


Ilustración 33 Base Layer de Animación [13]

A continuación, se presentan las 6 escenas del proceso de experiencia del usuario que se obtiene al iniciar la aplicación, la cual contiene diferentes botones para navegar por la simulación, y el usuario pueda manipular las escenas obteniendo la información de la manera más sencilla posible, la ilustración 31 muestra el inicio de la aplicación, el intercambio de *scenes* se realiza mediante código C# detallado en el anexo *“MenuDelJuego”* donde también se asigna el botón de “salir” de la aplicación.

Ilustración 34 Menú principal del usuario (inicio de aplicación)

3.4.2 Instrucciones de Usuario.

Aunque la interfaz del usuario es muy intuitiva se creó una sección donde se explica el procedimiento de esta plataforma en la sección de "INSTRUCCIONES" donde el usuario puede ingresar dando un "clic" y leyendo sobre los aspectos a tener en cuanta para realizar los torneados virtuales que es uno de los objetivos principales de este proyecto, en la siguiente ilustración se observa este mensaje.

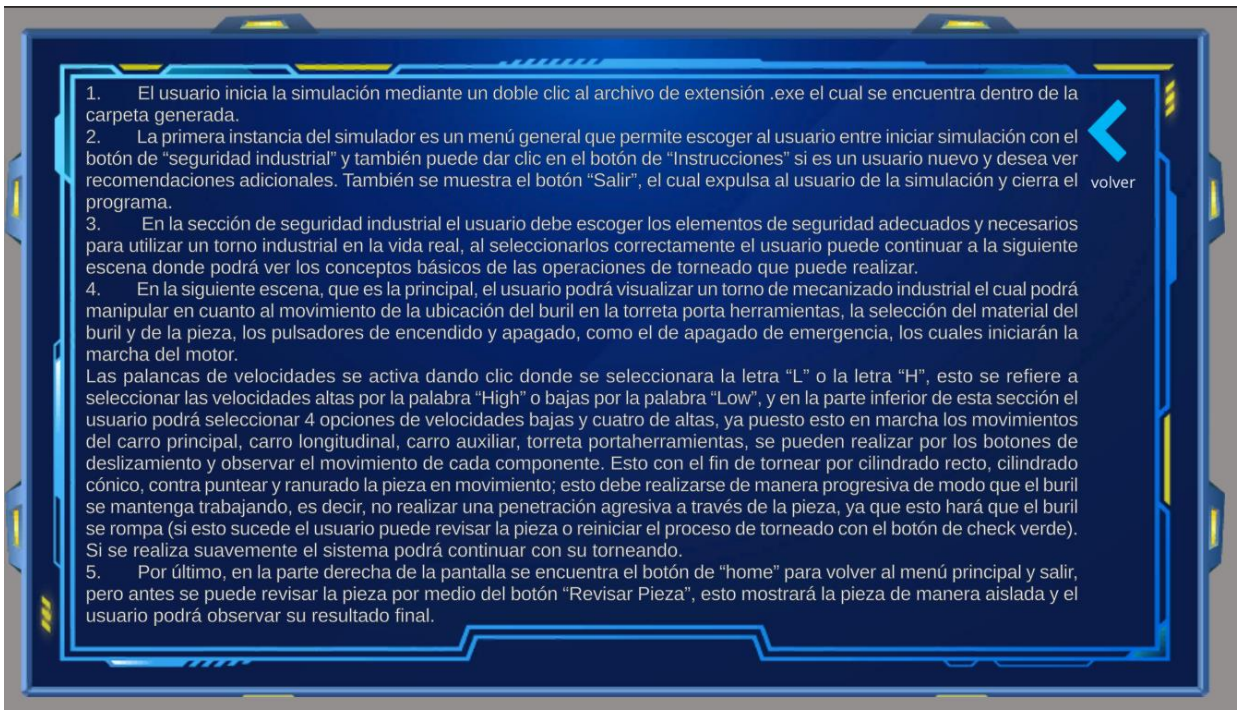


Ilustración 35: Instrucciones de uso

3.4.3 Menú de selección de herramientas de seguridad

Después de oprimir el botón de “Seguridad industrial” el usuario se dirige a la sección donde se seleccionan los elementos personales a usar mientras se utiliza el torno industrial, así como se visualiza en la siguiente ilustración; el usuario debe elegir los elementos correctos de seguridad y así poder continuar a la siguiente *scene* de la simulación. En este caso para la lógica de selección para el usuario se utiliza el código C# detallado en el Anexo “SeguridadIndustrialManager”, sobre las mismas “scenes”, se aplica el código *“ButtonSound”* que permite emitir un sonido al dar clic en los botones.

Ilustración 36 Selección de elementos de seguridad por el usuario.

3.4.4 Normas de seguridad adicionales

Posteriormente cuando el usuario selecciona la indumentaria correcta, la siguiente escena muestra unas sugerencias adicionales para la seguridad e integridad del usuario que utiliza un torno industrial real.

Ilustración 37 Sugerencias adicionales

3.4.5 Simulación definición de las operaciones de torneado

Posterior a la confirmación de los elementos de seguridad, la siguiente escena muestra al usuario la definición de los procesos de torneado que se pueden utilizar en la simulación; entre los cuales se encuentran refrentado, cilindrado, cónico, taladrado y ranurado, donde se muestra una breve explicación conceptual de estos procesos.

Ilustración 38 Definiciones de torneado a utilizar

3.4.6 Ventana principal de simulación

Después de ver las definiciones de los tipos de torneado que se pueden realizar dentro de la simulación de realidad virtual y al dar clic en el botón "Continuar" el usuario visualizara un torno industrial y puede iniciar un proceso de torneado ya listo, al costado derecho puede seleccionar el material que desea tornear y puede también configurar la posición del buril mediante los círculos verdes que se observan a este costado, dando clic en estos la configuración se realizara intuitivamente, la sonido de ambientación de las primeras *scenes* se detiene al cambiar a la *scene* de torneado gracias al código "*GameplaySoundBehaviour".* Los códigos de CNC a identificar en lenguaje CNC a través de los movimientos del operario son movimientos básicos que el torno industrial realiza en este caso el código G00 que significa "posicionamiento inicial" (sin maquinar), lo toma el sistema como el inicio de la operaciones, también el G01 que significa "Interpolación lineal (Maquinando), el sistema lo identifica cuando se realiza un mecanizado lineal; G74 que significa "Taladrado Intermitente con salida para virutas", y es identificado cuando la herramienta contrapunto hace contacto con la pieza y G28 que se implementó, cuando el buril se rompe y se desea volver al home. [14]

Ilustración 39 Posición inicial para empezar a tornear

Desarrollo de hud (head up display)

El *head up display* es la parte de la interfaz gráfica que le ofrece información al usuario sobre lo que está pasando en la aplicación y le permite tomar decisiones. Para la aplicación se ofrece información de la posición de los carros, su ángulo de rotación, la velocidad de rotación del cabezal en rpm, el material de trabajo a devastar y del buril, finalmente por medio de luz en los botones *start / stop* se indica si el torno está en funcionamiento o detenido, los movimientos de los componentes del torno como el carro principal, el carro longitudinal, la torreta, el carro auxiliar y el contra punto se pueden controlar por los *sprite* por la conexión realizada mediante el código del anexo "*TornoControlManager"*. El material del buril es cambiado por el código del anexo *"MaterialSetter"* el cual permite cambiar la textura de esta herramienta, pero no tiene un efecto físico o de comportamiento en la *scene.*

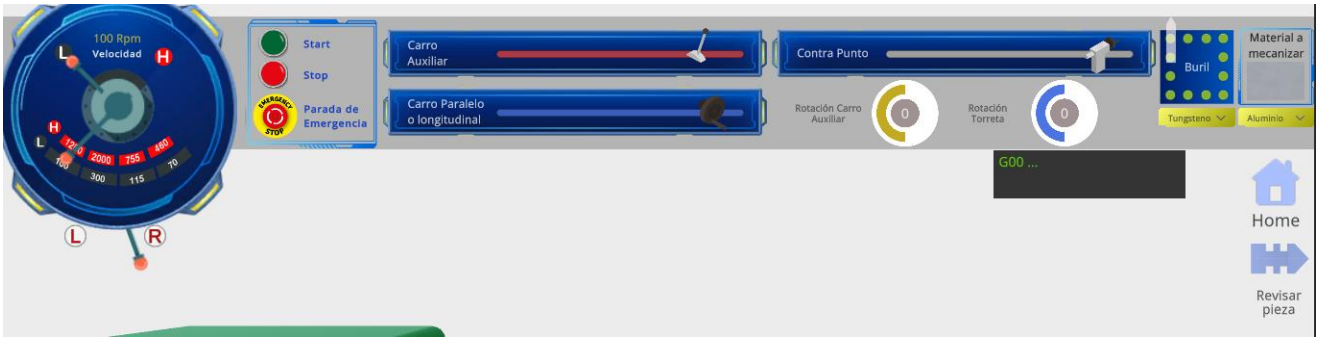


Ilustración 40 Elementos de la interfaz de movimientos para el torno

Herramientas *sprite sheet* (atlas).

Los elementos gráficos utilizados en el HUD fueron agrupados usando la técnica del *Sprite Sheet*, qué consiste en generar una imagen tipo atlas cuyo tamaño sea potencia de 2 (64x64 512x512 1024x1024 etc..) que contenga todos los elementos necesarios para componer la interfaz, posteriormente estos elementos son recortados por medio de coordenadas cartesianas dentro del motor de videojuegos. Este proceso se realiza para optimizar el rendimiento gráfico de las aplicaciones ya que es mejor para la tarjeta gráfica cargar una sola imagen grande que varias pequeñas. En la siguiente ilustración se visualizan las imágenes utilizadas para el desarrollo de la herramienta.[13]

De igual forma, la selección de las ilustraciones para la aplicación se realiza buscando contrastar la rigidez propia del sector industrial, buscando que los dibujos sean figurativos y representan claramente los conceptos asociados por medio del lenguaje visual.[13]

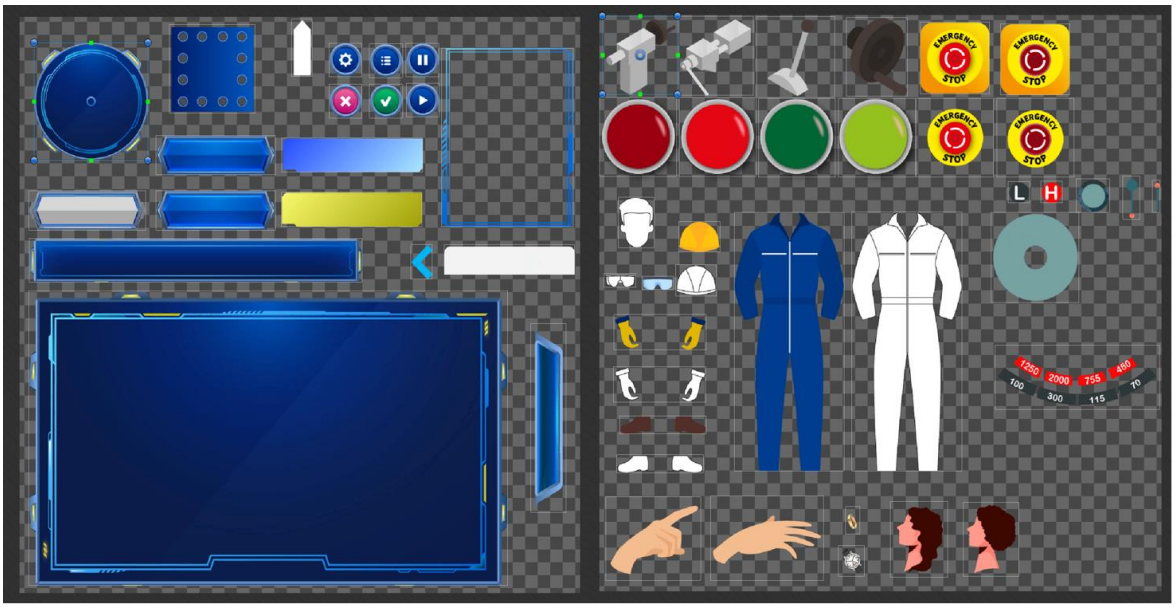


Ilustración 41 Imágenes PNG de uso como Sprite

3.4.7 La técnica utilizada fue la ilustración vectorial y el diseño flat

Al mover los carros del torno como el “Carro paralelo” o el “Contra punto” dando clic en los elementos de cada barra y sosteniendo el mismo para arrastrarlos a la derecha o a la izquierda, el usuario podrá visualizar como los carros se mueven simultáneamente y de esta forma buscar la manera de iniciar y realizar el torneado, la intención es realizar al principio el refrentado de la pieza y, posterior a ello, realizar mecanizados como tornado cilindrado, cilindrado cónico y ranurado, se debe realizar progresivamente para evitar que el buril se rompa, la programación se desarrolló de modo de que la simulación del mismo fuera lo más cercano a la realidad posible para el nivel de programación enfocado a este proyecto, en este caso se realizaron scripts destinados para el buril, permitiendo cambiar su posición, cambiar el material y textura de su modelo, y el que le asigna las propiedades de colisión con la pieza a modificar, este código se detalla en el Apéndice. [13]

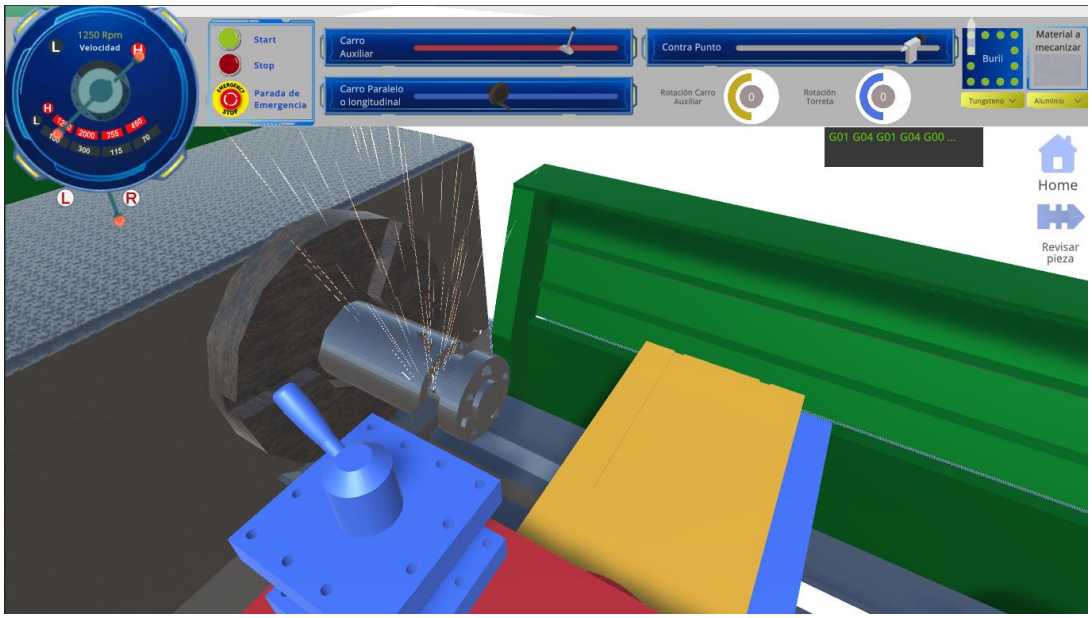


Ilustración 42 Proceso de tornado

El buril al realizar un avance de una manera incorrecta o demasiado rápido se romperá en esta simulación, indicando al usuario que los movimientos de avance con los carros deben ser progresivos como son en la realidad. La señal de “Se ha roto el buril”, saldrá junto con un *Check* blanco de fondo verde que se puede oprimir para reiniciar la operación, así como se ve en la siguiente ilustración.

Para realizar este efecto de rompimiento de buril se aplica una sentencia *if* el cual funciona con valores de contacto del “inspector” de *Unity* entre el buril y la pieza, teniendo en cuenta el rango de movimiento y contacto entre los dos, permitiendo que el usuario realice los movimientos similares de la realizad para tornear evitando romper el buril, también permite seleccionar el tipo de material de buril y activar el sistema de partículas de unity para generar las virutas y chisca al contacto entre piezas, este código se puede detallar en el anexo de los códigos de programación del anexo “*BurilController*”. El *script* de anexo *“BurilPointSetter”* permite posicionar el buril en las diferentes posiciones de la torreta porta herramientas, representada por *sprite 2D*, dentro del *canvas*. El tipo de

buril el cual cambia la textura del modelado 3D de este se realiza mediante el código del anexo *"MaterialSetter"*, el movimiento del contrapunto lineal por el eje longitudinal se manipula por medio del código del anexo *"ContrapuntoController"*. [13]

El entorno virtual posee animaciones de sonido programadas para funcionar dentro de las escenas, se programa una canción de ambientación dentro del menú principal y la selección de elementos de seguridad; aunque dentro de la *scene* de simulación de las operaciones de torneado se asignó un sonido para cada suceso que ocurre, entre ellas la colisión entre el buril y la pieza a tornear, el rompimiento del buril y el funcionamiento de las velocidades de la caja Norton, estas asignaciones se realizan mediante el código que se muestra en los anexos "*SoundManager"*, sin embargo la animación de sonido de las virutas se aplica mediante el código especificado para las partículas nombrado como "*ParticlesSound"*. [13]

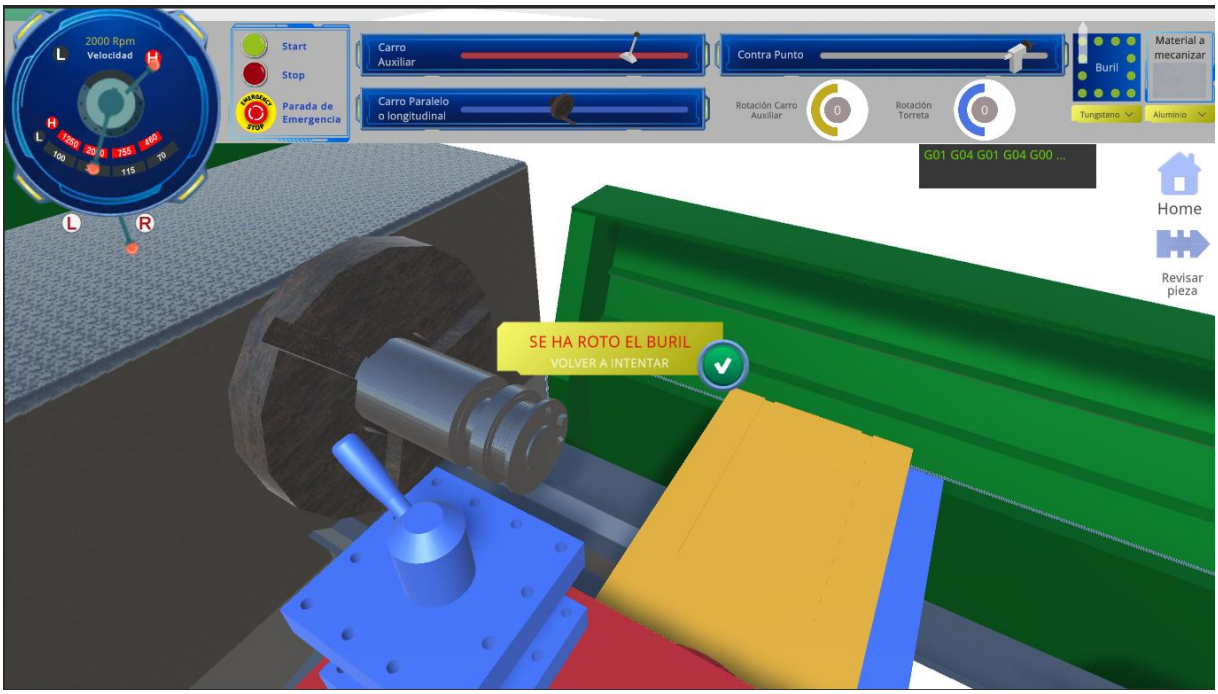


Ilustración 43 El buril roto.

3.5 Resultado Final

Al terminar de tornear la pieza utilizando las diferentes operaciones de torneado en la escena, se acciona el botón de "stop" o de "Parada de Emergencia" cuando se tenga el resultado esperado, allí el usuario puede oprimir el botón de "Revisar Pieza", y de esta forma aislar la visualización de la pieza realizada.

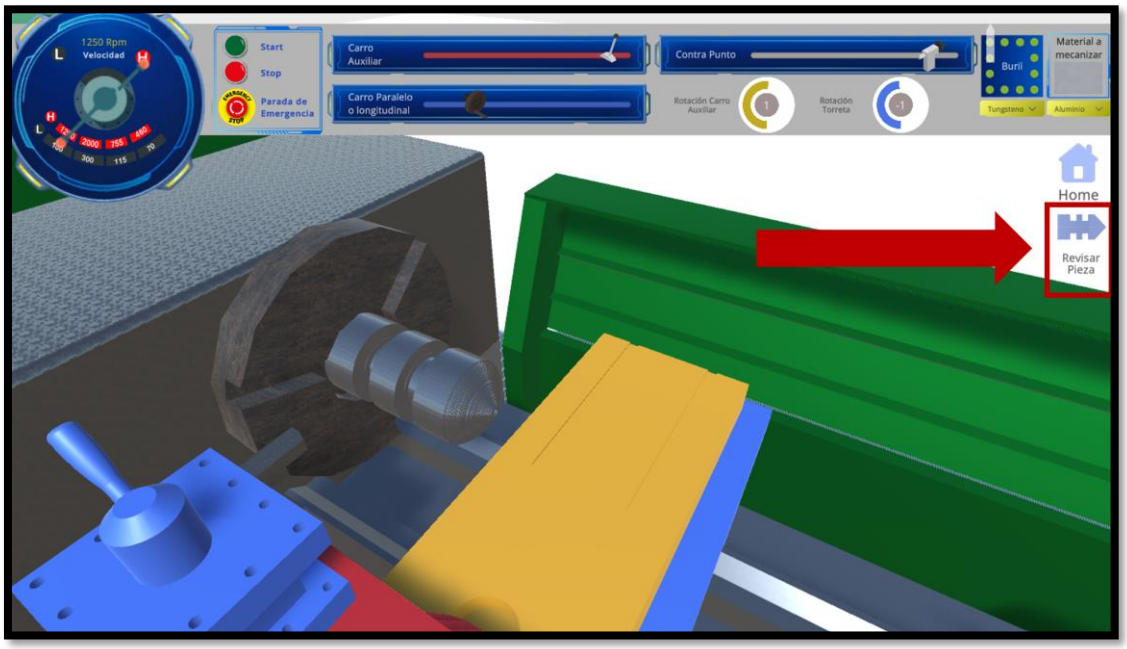


Ilustración 44. Resultado Final, Botón de Observación Final

Al dar clic en el botón de revisión, la pieza es centrada y aislada en la pantalla, allí el usuario puede orbitar la pieza para detallar el resultado final o parcial en fondo blanco, esta modificación en la scene se realiza mediante el código del anexo *"IsolateController"*.

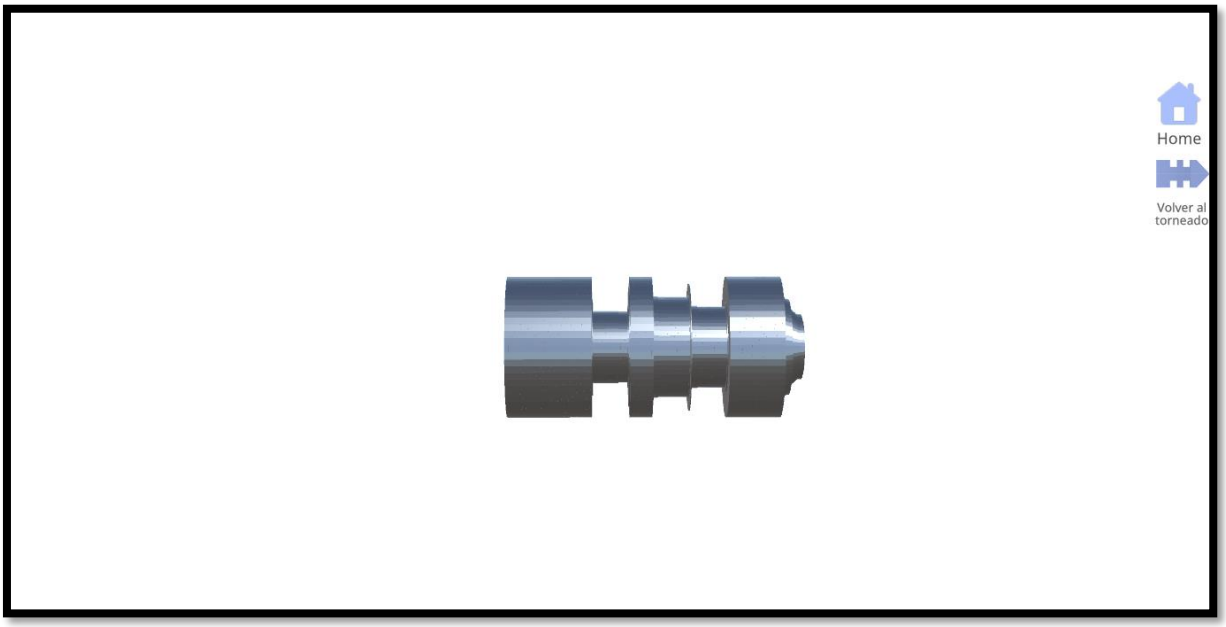


Ilustración 45 aislamiento de pieza para revisión final.

CAPÍTULO 4

4 Ejecutable y simulación

La generación del archivo ejecutable se realiza para poder compartir el simulador sin la necesidad de tener el *software Unity* instalado, este se genera mediante el menú *Build and Settings*, donde se deben enlistar las *scenes*, en este caso se generaron 5 *scenes*, enumeradas automáticamente como 0,1,2,3 y 4; creadas para la simulación del torno y allí se selecciona el tipo de dispositivo por el cual se va a ejecutar la simulación, en el caso de este proyecto se genera el archivo ejecutable para PC con sistema operativo Windows, posterior a ello, se da un *clic* en *Build*, y se guarda en la ruta deseada dentro de los archivos, en la siguiente se muestra los parámetros de este proceso. [13]

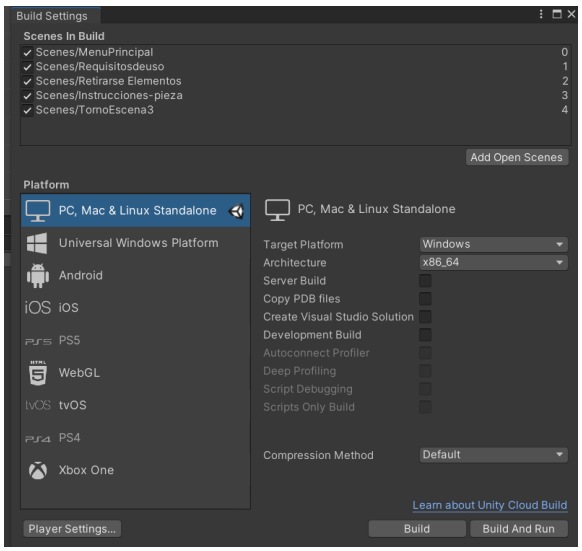


Ilustración 46 Parámetros para generar archivo ejecutable

4.1 Ejecución de la simulación virtual

El archivo ejecutable puede ser utilizado mediante un ordenador sin la necesidad de tener el *software Unity 3D*, este archivo es inicialmente generado para utilizarse en un sistema operativo *Windows* y basta con dar doble clic en el archivo ".exe". En este caso, como se observa en la ilustración, con el nombre de TornoT que es el nombre del proyecto dentro del motor de video juegos dentro de la carpeta que es generada pot *Unity*. Posteriormente al dar doble *clic* sobre el archivo principal del programa aparecerá el símbolo de *Unity* antes de iniciar la simulación planeada. [13]

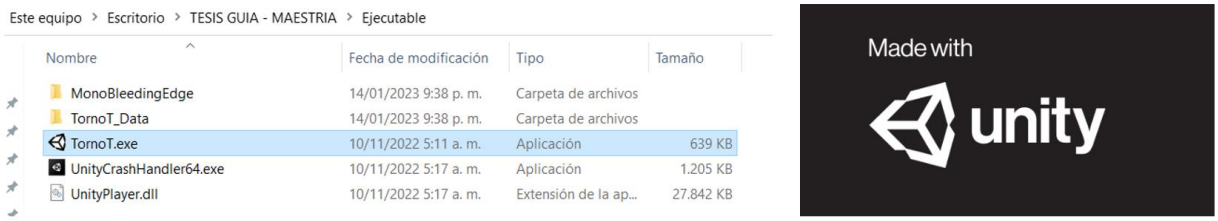


Ilustración 47 Ejecución de la simulación.

Es posible incluir dispositivos periféricos de realidad virtual adicionales como volantes hápticos, gafas óculos o de otra marca de visores de realidad virtual, al igual que controles remotos desde que se realice la configuración pertinente.

4.2 *Testing* del simulador:

Pruebas y experiencias de usuario

Después de realizadas las pruebas de usuario a 6 personas que manipularan la simulación del torno industrial, estas personas realizaron una encuesta de calificación para la simulación, entre los aspectos de innovación, aprendizaje, proyección, y animaciones. Con el objetivo de realizar un análisis sobre mejoras que se pueden implementar y que camino podría direccionar el continuo desarrollo del aplicativo.

Usuarios	Innovación	Aprendizaje	Proyección	Animación
Usuario 1 (Diseñador Industrial) Universidad Nacional	**3**	**4**	**5**	**4**
Ingeniero en Automatización (Unisalle) – Sector Agrario Industrial	**4**	**3**	**4**	**5**
Socióloga U. Externado	**4**	**5**	**5**	**4**
Ingeniera Mecatrónica Universidad Piloto de Colombia.	**4**	**5**	**4**	**2**
Ingeniera mecánica Universidad Distrital	**4**	**3**	**5**	**4**

Tabla 6 Calificación de Usuarios de acuerdo con la experiencia de Uso.

Las pruebas de experiencia de usuario se les realizaron a 5 personas, los cuales completaron la encuesta del Anexo, teniendo en cuenta los aspectos del proyecto los resultados los observamos en la siguiente ilustración donde fueron compilados.

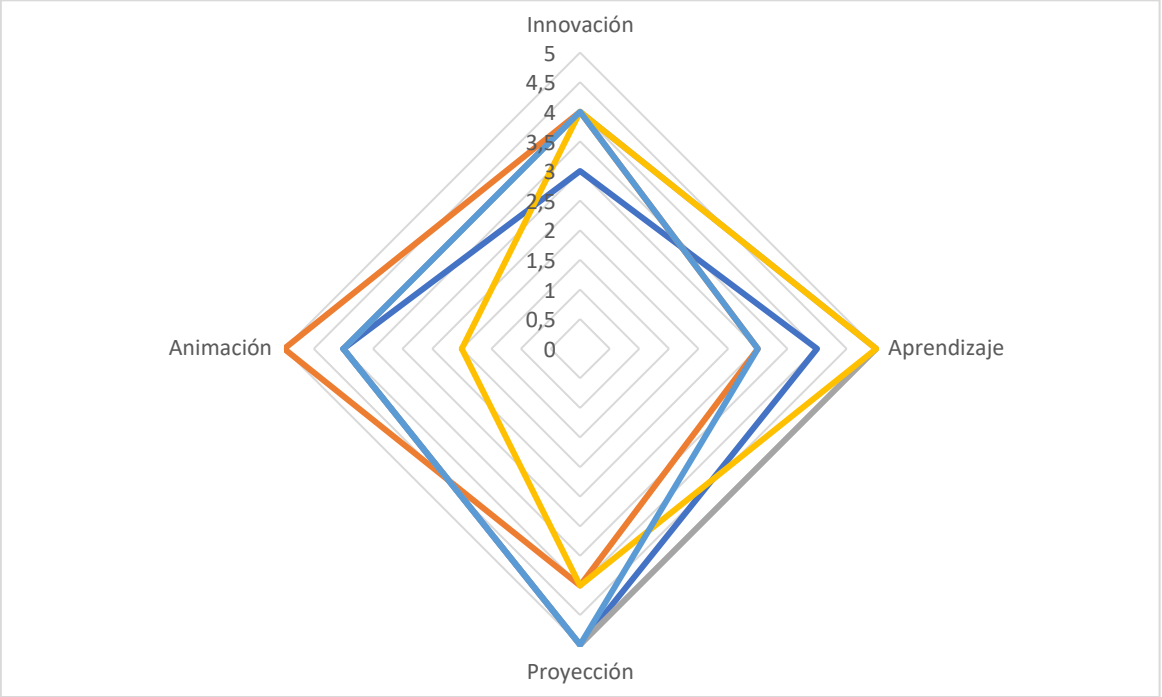


Ilustración 48 resultados encuesta a quienes probaron el simulador.

- **Guía de usuario:**

1. El usuario inicia la simulación mediante un doble clic al archivo de extensión .exe el cual se encuentra dentro de la carpeta generada.
2. La primera instancia del simulador es un menú general que permite escoger al usuario entre iniciar simulación con el botón de “seguridad industrial” y también puede dar clic en el botón de “Instrucciones” si es un usuario nuevo y desea ver recomendaciones adicionales. También se muestra el botón “Salir”, el cual expulsa al usuario de la simulación y cierra el programa.
3. En la sección de seguridad industrial el usuario debe escoger los elementos de seguridad adecuados y necesarios para utilizar un torno industrial en la vida real, al seleccionarlos correctamente el usuario puede continuar a la siguiente escena donde podrá ver los conceptos básicos de las operaciones de torneado que puede realizar.
4. En la siguiente escena, que es la principal, el usuario podrá visualizar un torno de mecanizado industrial el cual podrá manipular en cuanto al movimiento de la ubicación del buril en la torreta porta herramientas, la selección del material del buril y de la pieza, los pulsadores de encendido y apagado, como el de apagado de emergencia, los cuales iniciarán la marcha del motor.

   Las palancas de velocidades se activa dando clic donde se seleccionara la letra “L” o la letra “H”, esto se refiere a seleccionar las velocidades altas por la palabra “*High*” o bajas por la palabra “Low”, y en la parte inferior de esta sección el usuario podrá seleccionar 4 opciones de velocidades bajas y cuatro de altas, ya puesto esto en marcha los movimientos del carro principal, carro longitudinal, carro auxiliar, torreta portaherramientas, se pueden realizar por los botones de

deslizamiento y observar el movimiento de cada componente. Esto con el fin de tornear por cilindrado recto, cilindrado cónico, contra puntear y ranurado la pieza en movimiento; esto debe realizarse de manera progresiva de modo que el buril se mantenga trabajando, es decir, no realizar una penetración agresiva a través de la pieza, ya que esto hará que el buril se rompa (si esto sucede el usuario puede revisar la pieza o reiniciar el proceso de torneado con el botón de *check* verde). Si se realiza suavemente el sistema podrá continuar con su torneando.

5. Por último, en la parte derecha de la pantalla se encuentra el botón de "*home*" para volver al menú principal y salir, pero antes se puede revisar la pieza por medio del botón "Revisar Pieza", esto mostrará la pieza de manera aislada y el usuario podrá observar su resultado final.

CAPÍTULO 5

5 Recomendaciones y experiencias.

5.1 Recomendaciones

- Se recomienda leer las instrucciones de la plataforma antes de su uso, realizar las simulaciones de forma cómoda, sentado forma erguida frente al ordenador sin periféricos de realidad virtual, si se usan periféricos como visores o volantes hápticos seguir las recomendaciones de cada dispositivo, en algunos casos es posible ejecutar las simulaciones de pie.
- El uso de la realidad virtual apoya el aprendizaje del uso del torno industrial, mas no asegura la capacitación total de los usuarios para usar las herramientas mecánicas, así que siempre se debe contar con la presencia de un instructor y de su capacitación convencional para empezar las primeras sesiones de torneado en tornos reales.
- La inclusión de periféricos dentro de la realidad virtual debe ser acorde a las perillas de los tornos industriales existentes o que se encuentran en desarrollo y así brindar un acercamiento mayor a la inmersión de la realidad virtual.
- Para la continuación del desarrollo de este proyecto, o de cualquier tipo de proyecto que se inicie en *Unity*, se debe considerar que los elementos 3D importados desde otro *software* deben tener un proceso de retopología para priorizar el rendimiento de los procesadores al activar las aplicaciones en el dispositivo deseado.

5.2 Experiencia de construcción del sistema de realidad virtual.

- La información es lo que hace valiosa a una plataforma de aprendizaje, pero al mismo tiempo la manera de presentarla, en este caso *Unity* tiene la posibilidad de mostrar de múltiples maneras y esto también depende del nivel de programación que haya dentro de los desarrolladores. En este caso se intentaron desarrollar los procesos de torneado por medio de anillos que formaban el cilindro de materia prima, donde el buril por medio de la colisión corría estos anillos hacia la izquierda, simulando la acción de tornear progresivamente, pero de esta forma se impedía desarrollar el proceso de ranurado al igual que el proceso cónico de una forma correcta, posterior a ellos se utilizó la técnica de ***Chunk***, la cual permite desbastar el material de la materia prima de una forma libre, aunque este tipo de procesos abarca mayor demanda al rendimiento del procesador del *software* de desarrollo, por tal razón el método final utilizado para tornear es modelar la pieza por medio de múltiples piezas de filas de anillo y programar el efecto de rompimiento de buril si hace contacto con dos de estos anillos en un tiempo muy corto.

- Los parámetros del proyecto dependen masivamente de la programación usada en los *scripts* orientada a cada objeto. Estos parámetros orientados a los movimientos longitudinales y giros específicos en grados daban la pauta para llegar a los objetivos específicos de la programación, como el rompimiento del buril en cierto ángulo o a cierta profundidad o acercamiento entre los sólidos. Es allí donde siempre estará la dificultad del desarrollo de este tipo de proyectos, además de la correcta modelación geométrica de los objetos y la capacidad para ajustar la experiencia de usuario mediante colores y animaciones, para la creación de aplicaciones y videojuegos el desarrollador no debe asumir la parametrización longitudinal con la que se modelan ensambles en software CAD, ya que en los entornos 3D de videojuegos se prioriza la reducción de la cantidad de polígonos.

- Una de las dificultades en el desarrollo en *Unity 3D* es equilibrar las unidades de medida dentro del espacio tridimensional, aunque existan las herramientas para obtener movimiento lineales y rotacionales muy exactos de acuerdo con la escala presentada en los manuales (1 = 1m), la compatibilidad de piezas importadas con las desarrolladas dentro de la interfaz siempre tendrá que ser verificadas mientras estas son ubicadas en el espacio ya que la interfaz no posee una herramienta de medición como se presenta en los *software* de modelados 3D.

- Los motores de video juegos y la modelación 3D en ellos no priorizan la exactitud de medidas como lo hace un software CAD y como debería ser dentro de un simulador de torneado, la opción para evitar perder los detalles dimensionales dentro de *Unity 3D* es equilibrar el proceso de retopologia especialmente en los objetos que tienen el contacto directo con la simulación de colisión, para no sacrificar los efectos dimensionales, aunque esto también dependerá del dispositivo donde se desea ejecutar el simular, ya que en este receptor se puede solventar la carga de gráficos de la interfaz creada.

- El efecto de sistema de partículas de *Unity* permitió expresar el desprendimiento de virutas al momento del contacto entre las dos piezas sólidas y simular un proceso de torneado; aunque el acercamiento del efecto para el usuario es entendible, se requiere de mayor variabilidad de parámetros para poder expresar como se comportarían las virutas de acuerdo con el cambio de material y de velocidad, en este proyecto no se tuvieron en cuenta estas variables debido a la cantidad de datos que se deberían procesar para lograr un acercamiento con la realidad.

6 Conclusiones

- Se diseñó e implementó un sistema de realidad virtual que permite al usuario interactuar y realizar las operaciones de mecanizado de un torno convencional industrial, en un entorno virtual controlado para priorizar la seguridad del usuario y ser utilizado como método pedagógico reduciendo las fallas en operación y actos inseguros cuando se manipule un torno real. Lo más importante es que el usuario consigue entender los procesos de torneado más usados, realizando un diseño propio conceptual hasta finalizarlo, teniendo la posibilidad de experimentar el error de romper la herramienta de corte, pero en la seguridad de la simulación. Lo más complicado de este proyecto es el nivel de programación de código c#, por esta razón la metodología de torneado hibrido donde se combina el CNC y el torno convencional se desarrolló mediante una rutina que permite identificar los movimientos de código, en este caso la detección inicio, los movimientos lineales, el posicionamiento inicial y el taladrado intermitente.
- Se construyó mediante *software* paramétrico CAD, en este caso *Autodesk Inventor*, el ensamble del torno hibrido industrial con cada uno de sus componentes; al ser migrado este archivo al motor de video juegos *Unity 3D* se puede visualizar y es compatible con algunos de los códigos de programación C#; pero en este caso elementos como el carro transversal, longitudinal, auxiliar, el buril y la pieza a mecanizar, se remodelaron con las herramientas de la interfaz de *Unity 3D*, para obtener una mayor compatibilidad con los efectos asignados por medio de *scripts,* como reducir su escala al ser mecanizados y aplicar el efecto de generación de partículas, al igual que hacer el efecto de rompimiento de la herramienta por colisión, aunque esto sacrifica la exactitud de las medidas de estos modelos.
- Se generó una interfaz para el usuario que permite operar un torno industrial con los procesos de mecanizado más utilizados, en este caso fueron el torneado cilíndrico, torneado cónico, refrentado, ranurado y centro punteado; permitiendo utilizar y visualizar un proceso virtual de mecanizado con la posibilidad de experimentar el error de romper el buril si se excede el avance progresivo que es requerido para cada operación. Ya que el desarrollo del modelado 3D junto con los *scripts* del funcionamiento se pensaron específicamente para generar estos eventos, siempre informando y advirtiendo al usuario de los elementos de seguridad que se deben utilizar en un torno real para cuidar su integridad.
- Se estableció una comunicación entre el usuario y el simulador ya que las acciones del usuario repercuten en la interfaz y avance de la simulación, retroalimentando la selección de implementos de seguridad, donde este debe elegir los correctos para el uso adecuado; además, el proceso de mecanizado se debe realizar con la atención y concentración suficiente para no arruinar la herramienta de corte, si este rompimiento sucede el usuario visualizara este suceso y es ratificado por los mensajes de pantalla y el sonido, donde podrá

reiniciar el torneado de una pieza nueva para volver a cumplir con el objetivo de tornear la pieza utilizando uno o varios tipos de mecanizado. El archivo de desarrollo de *Unity 3D* con sus archivos respectivos de desarrollo queda abierto para la implementación de periféricos de realidad virtual como volantes hápticos o visores de realidad virtual para mejorar la inmersión del usuario en entorno de simulación.

- Se aportó por medio de la interfaz gráfica del menú del simulador los tipos de elementos de seguridad que son necesarios para reducir la posibilidad de tener accidentes cuando se utilice un torno industrial, realizando una interfaz interactiva de selección de elementos donde el sistema definirá cuales son los correctos, al igual que brinda sugerencias de extraer objetos de uso cotidiano que pueden aumentar la posibilidad de tener un accidente mientras se utiliza este sistema mecánico.

Referencias

[1] M. A. Fredes García y A. A. Vial Villalonga, Manual instructivo de operación y prácticas didáctica de torno CNC para el desarrollo docente, Universidad Técnica Federico Santa María , Concepción, 2018.

[2] J. J. Velásquez Bravo, N. J. López Lezama y F. R. Chávez Cerpas, *Propuesta de un manual de higíene y seguridad industrial en el taller de torno y fundición Franciso Bonilla*, Universidad Nacional De Ingeniería, Managua, 2013.

[3] J. E. Díaz Gutiérre, E. D Demoya Correal y O. A González Torres, *Diseño y construcción de prototipo para el aprendizaje del proceso técnico de afilado de buril*, Universidad Pedagógica Nacional De Colombia, Bogotá, 2016.

[4] J. A. Rivas Díaz, *Cuál es el efecto de la velocidad de corte, velocidad de avance y la profundidad de corte sobre la rugosidad y fuerza de corte para el mecanizado del bronce sae*, Escuela De Ingeniería Mecánica, Universidad Nacional de Trujillo, Perú, 2019.

[5] *Mejoramiento e inspección mediante una innovación de mantenimiento preventivo y correctivo para maquinaria industrial (torno convencional y fresadora cnc)*, Plastimaq de Toluca, Chiapas, 2014.

[6] N. Konstantinos, *Improving Chemical Plant Safety Training Using Virtual Reality*, University of NottinghamSchool of Chemical, Environmental, and Mining Engineering, U. K, 2001

[7] D. F. Bueno Guapacha y J. S. González Vargas, Guía de aprendizaje para manejo de torno CNC WABECO CC-D6000 E, Universidad Tecnológica de Pereira, Pereira, 2015.

[8] D. García, Simulación del proceso de mecanizado por torno a alta velocidad de la aleación Inconel 718, 2010.

[9] F. Morales, Trabajos básicos-explicación, procedimiento en los mecanizados con máquinas herramientas torno, 2017.

[10] García, Santiago Andrés. Módulo de comunicación entre un volante háptico de carro de avance y un entorno de realidad virtual para la simulación de un torno híbrido virtual. Universidad Nacional de Colombia 2021.

[11] J.A. Robayo Murillo, *Diseño e implementación de un sistema de realidad virtual para una planta pasteurizadora de leche*, Facultad de ingeniería, Universidad de la Salle, Bogotá, 2016.

[12] (2023, en. 30) "Unity", [Internet]. Disponible en https://www.Unity.com

[13] (2023, en. 30) "Unity Documentation", [Internet]. Disponible en https://docs.unity3d.com

[14] (2018, ago.23). "Máquinas herramientas híbridas: lo mejor de dos mundos", [Internet]. Disponible en https://espanol.worldindustrialreporter.com/maquinas-herramientas-hibridas-lo-mejor-de-dos-mundos/

[15] (2023, en. 30). "Mi pesa, grupo empresarial", [Internet]. Disponible en https://www.mipesa.es/que-es-un-taller-de-mecanizado/

[16] (2018, abr. 13). "Seguridad en un taller de mecanizado", [Internet]. Disponible en https://mecanicacuriel.com/2018/04/13/seguridad-en-un-taller-de-mecanizado/

[17] J. Crespo Cánovas, *Aleaciones de cobre: Desarrollos recientes y nuevas perspectivas*, Escuela Técnica Superior de Ingeniería Industrial, Universidad Politécnica de Cartagena, Cartagena, 2021

[18] (2023, en. 30)." BEITXU STUDIOS, Soluciones de comunicación a medida con las tecnologías más innovadoras", [Internet]. Disponible en http://www.beitxustudios.eus/

[19] (2023, en. 30). "Platos para torno", [Internet]. Disponible en https://tornosparametal.net/platos-torno/

[20] (2021, ag. 18). "Ferretería JRC, Buril: herramienta de corte para torno", [Internet]. Disponible en https://ferreteriajrc.com/blog/tipos-de-buril/

[21] M. Rodríguez Gabarro. (2023, en.). "Tutorial n° 38, Fundamentos de los procesos de mecanizado, [Internet]. Disponible en https://ingemecanica.com/tutorialsemanal/tutorialn38.html

[22] J. E. Caballero Prieto, *Evaluación de un buril de acero superrápido (hss) modificado con implantación de iones de nitrógeno y titanio*, Universidad Francisco de Paula Santander, San José de Cúcuta, 2017.

[23] G. Gómez Martínez, *Diseño y Modelado 3D de un Personaje para Animacion*, Facultat de Belles Arts de Sant Carles. Universitat Politécnica de Valencia., Valencia, 2021. [Internet]. Disponible en https://riunet.upv.es/bitstream/handle/10251/172683/Gomez%20-%20Diseno%20y%20modelado%203D%20de%20un%20personaje%20para%20animacion.pdf?sequence=1&isAllowed=y

[24] J. E. Márquez *Delgado, Optimización de la programación (scheduling) en Talleres de Mecanizado*, Escuela Técnica Superior De Ingenieros Agrónomos, Universidad Politécnica De Madrid, Madrid, 2012.

[25] P. Fraga-Lamas et al. (2018, dic.). "Review on IAR Systems for the Industry 4.0 Shipyard", IEEE Access. [Internet]. Vol.6. Disponible en https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8298525

[26] F. C Chaparro y R. Quintero (2007, nov.). “Propuesta de diseño, construcción y ensayo de portaherramienta funcional”, Escuela Tecnologica Instituto Técnico Central, pp 16-29.

[27] D. Guzmán Perea, *Diseño y fabricación de un sistema servoasistido de torreta con capacidad de 3 herramientas para la reconversión de un torno convencional a CNC*, Facultad de Ingeniería Tecnología Mecánica, Institución Universitaria Pascual Bravo, Medellín, 2013.

[28] D. F. Bueno Guapacha, y J. S González Vargas, *Guía de aprendizaje para manejo de torno CNC WABECO CC-D6000 E*, Universidad Tecnológica de Pereira, Pereira, 2015.

[29] L. C. Flórez García, H. A.González Rojas y et al., “Simulación en 3D del arranque de viruta en torno para el acero AISI 1045 bajo Johnson Cook usando ANSYS”. En *XIII Congresso Ibero-Americano de Engenharia Mecânica/Ingeniería Mecánica, Universidade Nova de Lisboa,* 2017, pp- 23-26.

[30] P. Blanco Ostos, *Tipo de viruta durante un proceso de mecanizado en función de la velocidad de corte*, Universidad de Sevilla, Sevilla, 2021.

Anexos

Encuesta:

Encuesta de Prueba de Usuario de Simulador de Torno Industrial

Califique de 1 a 5 las características de la plataforma de realidad virtual mencionadas en cada pregunta, siendo 5 la mejor calificación y 1 la calificación más baja.

Nombre: _____ Profesión: _____

1. ¿Qué tan innovadora es para usted la aplicación del torno industrial?
   1☐ 2☐ 3☐ 4☐ 5☐
2. ¿Qué tan viable al aprendizaje es la experiencia de esta plataforma?
   1☐ 2☐ 3☐ 4☐ 5☐
3. ¿Considera que el proyecto tiene proyección para continuar con su desarrollo?
   1☐ 2☐ 3☐ 4☐ 5☐
4. ¿Qué calificación le otorga usted a las animaciones y experiencia de este simulador?
   1☐ 2☐ 3☐ 4☐ 5☐

Anexo: Códigos C# Unity 3D.

Menú del Juego

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;
public class Menualjuego : MonoBehaviour
{
   void Start()
   {
```

```
    }
    void Update()
    {
    }
    public void EscenaSimulador()
    {
        SceneManager.LoadScene("TornoEscena");
    }
    public void MenuPrincipal()
    {
        SceneManager.LoadScene("MenuPrincipal");
    }
    public void Seguridad_Industrial()
    {
        SceneManager.LoadScene("Requisitosdeuso");
    }
    public void Salir()
    {
        Application.Quit();
    }
}
```

BurilChunkDestroy

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BurilChunkDestroy : MonoBehaviour
{
    [SerializeField]
    ParticleSystem sparks;

    public float   InfluenceRadius       = 1.0f;
    public float   Force                 = 1.0f;
    public bool    CheckStructureIntegrity = false;
    public Vector3 PosStart;
```

```
public Vector3 PosEnd;
public float   MoveDuration          = 2.0f;

FracturedObject[] m_aFracturedObjects = null;
float m_fStartTime = 0.0f;

void Start()
{
    m_aFracturedObjects = FindObjectsOfType(typeof(FracturedObject)) as FracturedObject[];
    m_fStartTime        = Time.time;
    foreach(FracturedObject fracturedObject in m_aFracturedObjects)
    {
    }
    print(m_aFracturedObjects.Length);
}
private void OnTriggerEnter(Collider other) {

    //Destroy(other.gameObject);
    //print("collision with"+other.name);

     var f = other.gameObject.GetComponent<FracturedObject>();
     var o = other.gameObject.GetComponent<FracturedChunk>();
     var p = other.gameObject.GetComponentInParent<FracturedObject>();

    if(o != null)
    {
        print("collision with"+other.name);
    }
}
private void OnCollisionEnter(Collision other) {

   var o = other.gameObject.GetComponent<FracturedChunk>();

    if(o != null)
    {
     print("collision with"+other.gameObject.name);
      o.DetachFromObject(false);
```

```
        }
    }
}
```

BurilController

```
using UnityEngine;
using UnityEditor;
using System.Collections;
using System.Collections.Generic;

[CustomEditor(typeof(CombinedMesh))]
public class CombinedMeshEditor : Editor
{
    SerializedProperty PropSaveMeshAsset;
    SerializedProperty PropKeepPosition;
    SerializedProperty PropPivotMode;
    SerializedProperty PropMeshObjects;
    SerializedProperty PropRootNode;

    [MenuItem("GameObject/Create Other/Ultimate Game Tools/Combined Mesh")]
    static void CreateFracturedObject()
    {
        GameObject combinedMeshObject = new GameObject();
        combinedMeshObject.name = "Combined Mesh";
        combinedMeshObject.transform.position = Vector3.zero;
        combinedMeshObject.AddComponent<CombinedMesh>();

        Selection.activeGameObject = combinedMeshObject;
    }
    void Progress(string strMessage, float fT)
    {
        CombinedMesh combinedMesh = serializedObject.targetObject as CombinedMesh;

        Repaint();

        if(EditorUtility.DisplayCancelableProgressBar("Combining", strMessage, fT))
```

```
        {
            combinedMesh.CancelCombining();
        }
    }
    void OnEnable()
    {
        PropSaveMeshAsset = serializedObject.FindProperty("SaveMeshAsset");
        PropKeepPosition  = serializedObject.FindProperty("KeepPosition");
        PropPivotMode     = serializedObject.FindProperty("PivotMode");
        PropMeshObjects   = serializedObject.FindProperty("MeshObjects");
        PropRootNode      = serializedObject.FindProperty("RootNode");
    }

    public override void OnInspectorGUI()
    {
        int nButtonWidth = 200;

        serializedObject.Update();

        CombinedMesh combinedMesh = serializedObject.targetObject as CombinedMesh;

        EditorGUILayout.Space();
        EditorGUILayout.Space();

        int nNumObjects = combinedMesh.MeshObjects != null ? combinedMesh.MeshObjects.Length :
0;
        EditorGUILayout.PropertyField(PropSaveMeshAsset, new GUIContent("Enable Prefab
Usage", "If activated, will save the mesh to an asset file on disc. Use this  if you want to use the
generated mesh in prefabs, otherwise prefabs won't reference the mesh correctly."));
        EditorGUILayout.PropertyField(PropKeepPosition,  new GUIContent("Keep Position", "If keep
position is activated, the gameobject will keep its current position. Otherwise it will reposition itself to
match the objects being combined."));
        EditorGUILayout.PropertyField(PropPivotMode,  new GUIContent("Place Pivot Mode",
"Where to place the pivot."));
        EditorGUILayout.PropertyField(PropMeshObjects,  new GUIContent("Source Mesh Objects
List (" + nNumObjects + " elements)", "The list of objects whose meshes to combine."), true);
```

```
        EditorGUILayout.PropertyField(PropRootNode,      new GUIContent("Root node", "Specify an object to set it and its whole hierarchy to the list of objects to combine."));

        EditorGUILayout.Space();
        EditorGUILayout.Space();

        EditorGUILayout.BeginHorizontal();
        GUILayout.Label(" ");

        serializedObject.ApplyModifiedProperties();

        if(GUILayout.Button(new GUIContent("Build List From Root Node", "Will build the Source Mesh Objects List using the given object and all the hierarchy below it."), GUILayout.Width(nButtonWidth)))
        {
          if(PropRootNode.objectReferenceValue)
          {
            List<MeshFilter> listMeshFilters = new List<MeshFilter>();
            BuildMeshFilterListRecursive(PropRootNode.objectReferenceValue as GameObject, listMeshFilters);
            combinedMesh.MeshObjects = listMeshFilters.ToArray();
          }
        }
        GUILayout.Label(" ");
        EditorGUILayout.EndHorizontal();

        EditorGUILayout.BeginHorizontal();
        GUILayout.Label(" ");
        if(GUILayout.Button(new GUIContent("Combine", "Starts the combine process."), GUILayout.Width(nButtonWidth)))
        {
          try
          {
            combinedMesh.Combine(Progress);
          }
          catch(System.Exception e)
          {
```

```
            Debug.LogError("Exception Type: " + e.GetType().ToString() + ". Message: " +
e.Message.ToString() + ". Stack Trace: " + e.StackTrace.ToString());
        }
        EditorUtility.ClearProgressBar();
    }
    GUILayout.Label(" ");
    EditorGUILayout.EndHorizontal();
  }
  void BuildMeshFilterListRecursive(GameObject node, List<MeshFilter> listMeshFilters)
  {
    MeshFilter meshFilter = node.GetComponent<MeshFilter>();

    if(meshFilter && node.GetComponent<Renderer>())
    {
      listMeshFilters.Add(meshFilter);
    }
    for(int nChild = 0; nChild < node.transform.childCount; nChild++)
    {
      BuildMeshFilterListRecursive(node.transform.GetChild(nChild).gameObject,
listMeshFilters);
    }
  }
}
```

BurilPointSetter

```
using UnityEngine;
using UnityEngine.EventSystems;
public class BurilPointSetter : MonoBehaviour,IPointerDownHandler
{
  public void OnPointerDown(PointerEventData eventData)
  {
    print("click on point");

FindObjectOfType<BurilDragController>().SetBurilPos(GetComponent<RectTransform>().anchoredPosition,
this.gameObject);
  }
```

```
}
```

ButtonSound

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class ButtonSound : MonoBehaviour
{
    // Start is called before the first frame update
    AudioSource audioSource;
    private void Awake() {

        if(GetComponent<Button>() is Button b)
        {
            b.onClick.AddListener(playSound);
        }
    }
    void playSound()
    {
        if(FindObjectOfType<SoundManager>() is SoundManager manager)
        {
            manager.PlayBtnSound();
        }
    }
}
```

ContrapuntoController

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ContrapuntoController : MonoBehaviour
{


    [SerializeField]
```

```
    ParticleSystem sparks;


    [SerializeField]
    float desvastingFactor;


    private void OnTriggerStay(Collider other) {

        print("TriggerIn");

        var fixedDesvastingFactor = desvastingFactor * Time.deltaTime;

        if(other.CompareTag("material"))
        {
            if(other.gameObject.name == "1" ||  other.gameObject.name == "2" || other.gameObject.name
== "3" || other.gameObject.name == "4")
            {
                var p = other.transform.parent.gameObject.transform;
                p.localScale = new Vector3(1,p.localScale.y-0.01F,1);
                sparks.Play();
            }

        }

        if(other.CompareTag("material4"))
        {

                var p = other.transform;
                p.localScale    =    new    Vector3(p.localScale.x,    p.localScale.y,    p.localScale.z    -
fixedDesvastingFactor);;
                sparks.Play();
        }
    }
    private void OnTriggerExit(Collider other) {
        sparks.Stop();
    }
```

```
}
```

GameplaySoundBehaviour

```
using UnityEngine;
public class GameplaySoundBehaviour : MonoBehaviour
{
    void Start()
    {
        if(FindObjectOfType<SoundManager>() is SoundManager manager)
        {
            manager.StopBgMusic();
        }
    }
}
```

IsolateController

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class IsolateController : MonoBehaviour
{
    [SerializeField]
    GameObject[] tornoIsolateObjects;
    [SerializeField]
    GameObject[] piezaIsoleteObjects;

    [SerializeField]
    Cinemachine.CinemachineFreeLook isolateCamera;

    [SerializeField]
    GameObject material, isolateParent, tornoParent;

    [SerializeField]
    TMPro.TMP_Text buttonIsolateText;
```

```
[SerializeField]
string textIsolate, textBack;

bool isIsolate = false;

public void SwitchIsolate()
{
    if(isIsolate)
    {
        isIsolate = false;
        activeObjects(tornoIsolateObjects);
        buttonIsolateText.text = textIsolate;
        isolateCamera.Priority = 9;
        material.transform.SetParent(tornoParent.transform);
    }
    else
    {
        isIsolate = true;
        desactivateObjects(tornoIsolateObjects);
        isolateCamera.Priority = 11;
        material.transform.SetParent(isolateParent.transform);
        buttonIsolateText.text = textBack;
    }
}
void activeObjects(GameObject[] objects)
{
    foreach (var item in objects)
    {
        item.SetActive(true);
    }
}
 void desactivateObjects(GameObject[] objects)
{
    foreach (var item in objects)
    {
        item.SetActive(false);
    }
```

```
    }
}
```

MaterialMesh

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class MaterialMesh : MonoBehaviour
{
    Vector3[] newVertices;
    Vector2[] newUV;
    int[] newTriangles;

   void Start()
    {
        Mesh mesh = new Mesh();
        GetComponent<MeshFilter>().mesh = mesh;
        mesh.vertices = newVertices;
        mesh.uv = newUV;
        mesh.triangles = newTriangles;
    }
      void Update()
    {
        Mesh mesh = GetComponent<MeshFilter>().mesh;
        Vector3[] vertices = mesh.vertices;
        Vector3[] normals = mesh.normals;

       for (var i = 0; i < vertices.Length; i++)
        {
            vertices[i] += normals[i] * Mathf.Sin(Time.time);
        }

       mesh.vertices = vertices;
    }
}
```

MaterialSetter

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class MaterialSetter : MonoBehaviour
{

    [SerializeField]
    Material[] materials;

     [SerializeField]
    Material[] BurilMats;

    [SerializeField]
    MeshRenderer BurilMat;

    [SerializeField]

    GameObject mat;
    MeshRenderer[] mecanizeMats;

    [SerializeField]
    RawImage rawImageMaterial;

    private void Start() {
        SetMaterial(0);
        SetBurilMaterial(0);
    }
    public void SetMaterial(int val)
    {
        rawImageMaterial.texture = materials[val].mainTexture;
        mecanizeMats = mat.GetComponentsInChildren<MeshRenderer>();
        foreach (var item in mecanizeMats)
        {
            item.material = materials[val];
```

```
        }
    }
    public void SetBurilMaterial(int val)
    {
        BurilMat.material = BurilMats[val];
    }
}
```

MenuSoundBehaviour

```
using UnityEngine;
public class MenuSoundBehaviour : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {
        if(FindObjectOfType<SoundManager>() is SoundManager manager)
        {
            if(!manager.bgMusic.isPlaying)
            {
                manager.PlayBgMusic();
            }
        }
    }
}
```

OrbitCameraController

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Cinemachine;
public class OrbitCameraController : MonoBehaviour
{
    CinemachineFreeLook freeLookCamera;
    // Start is called before the first frame update
    void Start()
    {
```

```
        freeLookCamera = GetComponent<CinemachineFreeLook>();


        freeLookCamera.m_XAxis.m_InputAxisName = "";


        freeLookCamera.m_YAxis.m_InputAxisName = "";
    }
    // Update is called once per frame
    void Update()
    {
        if(Input.GetMouseButton(0))
        {
            freeLookCamera.m_XAxis.m_InputAxisValue = Input.GetAxis("Mouse X");
            freeLookCamera.m_YAxis.m_InputAxisValue = Input.GetAxis("Mouse Y");
        }
        else
        {
            freeLookCamera.m_XAxis.m_InputAxisValue = 0;
            freeLookCamera.m_YAxis.m_InputAxisValue = 0;
        }
    }
}
```

ParticlesSound

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class ParticlesSound : MonoBehaviour
{
    AudioSource audioSource;
    ParticleSystem particleSystem;
    // Start is called before the first frame update
    void Start()
    {
        audioSource = GetComponent<AudioSource>();
        particleSystem = GetComponent<ParticleSystem>();
    }
```

```
    // Update is called once per frame
    void Update()
    {
        if(particleSystem.isEmitting)
        {
            if(!audioSource.isPlaying)
                audioSource.Play();
        }
    }
}
```

SeguridadIndustrialManager

```
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;
public class SeguridadIndustrialManager : MonoBehaviour
{
    [SerializeField]
    GameObject[] textosObjetosSeleccionados;

    [SerializeField]
    GameObject textFail, textGood;

    [SerializeField]
    Button continueBTN;

    [SerializeField]
    Toggle[] toogleObjetosSeleccionados;
    // Start is called before the first frame update

    private void Start() {
        EvaluateToogle();
    }
    public void EvaluateToogle()
    {
        for (int i = 0; i < toogleObjetosSeleccionados.Length; i++)
```

```
        {
            textosObjetosSeleccionados[i].SetActive(toogleObjetosSeleccionados[i].isOn);
        }
        if(!toogleObjetosSeleccionados[0].isOn && toogleObjetosSeleccionados[1].isOn && toogleObjetosSeleccionados[2].isOn && !toogleObjetosSeleccionados[3].isOn && toogleObjetosSeleccionados[4].isOn)
        {
            textFail.SetActive(false);
            textGood.SetActive(true);
            continueBTN.interactable = true;
        }
        else
        {
            textFail.SetActive(true);
            textGood.SetActive(false);
            continueBTN.interactable = false;
        }
    }
    public void GotoNewLevel(int level)
    {
        SceneManager.LoadScene(level);
    }
}
```

SoundManager

```
using UnityEngine;
public class SoundManager : MonoBehaviour
{
    // Start is called before the first frame update
    public AudioSource bgMusic, sfxSound;
    [SerializeField]
    AudioClip btnSfxSound;
    private void Awake() {
        if(FindObjectsOfType<SoundManager>() is SoundManager[] managers)
        {
            if(managers.Length>1)
```

```
            Destroy(gameObject);
            else
            DontDestroyOnLoad(gameObject);
        }
    }
    public void PlayBgMusic()
    {
        bgMusic.Play();
    }
    public void StopBgMusic()
    {
        bgMusic.Stop();
    }
    public void PlaySFXSound(AudioClip clip)
    {
        sfxSound.PlayOneShot(clip);
    }
     public void PlayBtnSound()
    {
        sfxSound.PlayOneShot(btnSfxSound);
    }
}
```

TornoControlManager

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using Cinemachine;

public class TornoControlManager : MonoBehaviour
{
    [SerializeField]
    Sprite spriteRedOn,spriteRedOff,spriteGreenOn,  spriteGreenOff;

    [SerializeField]
```

```
    Button buttonOn, buttonOff;

    [SerializeField]
    GameObject carroPrincipal, carroTransversal, CarroRot, Torre, contraPunto, mandril, carroAuxiliar, carroTransversalRot;

    [SerializeField]
    Slider sliderCarroPrincipal, sliderTrasnversal, SliderRotCarro, sliderRotTorre, sliderContraPunto, sliderCarroAux, sliderRotCarroTransversal;

     [SerializeField]
     CinemachineVirtualCamera cam2, cam3;

     [SerializeField]
     TMPro.TextMeshProUGUI labelRotAux, LabelRotTorre , LabelRotTrasnversal, labelRpm;

    public float ZoomDistancecarro, ZoomDistanceContraPunto;
    public float limitLeftCarro, limitRightCarro;
    public float limitLeftContraPunto, limitRightContraPunto;
    public float limitBottomCarro, limitTopCarro;
    public int limitLeftRotCarro, limitRightRotCarro;
    public int limitLeftRotTorre, limitRightRotTorre;

    bool on;
    public float mandrilSpeed;

    AudioSource audioSource;

    // Start is called before the first frame update
    void Start()
    {
        audioSource = GetComponent<AudioSource>();
        OFF();
    }
    // Update is called once per frame
    void Update()
    {
```

```
        carroPrincipal.transform.localPosition = new
Vector3(Mathf.Lerp(limitLeftCarro,limitRightCarro,sliderCarroPrincipal.value) ,0,0);
        contraPunto.transform.localPosition = new
Vector3(Mathf.Lerp(limitLeftContraPunto,limitRightContraPunto,sliderContraPunto.value) ,0,0);
        carroTransversal.transform.localPosition = new
Vector3(0,0,Mathf.Lerp(limitBottomCarro,limitTopCarro,sliderTrasnversal.value));
        //CarroRot.transform.localRotation = Quaternion.Euler(0,SliderRotCarro.value,0);
        Torre.transform.localRotation = Quaternion.Euler(0,sliderRotTorre.value,0);
        carroAuxiliar.transform.localPosition = new Vector3(sliderCarroAux.value,0,0);
        carroTransversalRot.transform.localRotation =
Quaternion.Euler(0,0,sliderRotCarroTransversal.value);
        labelRotAux.text = SliderRotCarro.value.ToString();
        LabelRotTorre.text = sliderRotTorre.value.ToString();
        LabelRotTrasnversal.text = sliderRotCarroTransversal.value.ToString();

        if(sliderCarroPrincipal.value<ZoomDistancecarro )
        cam2.Priority =11;
        else
        cam2.Priority =9;
        if( sliderContraPunto.value<ZoomDistanceContraPunto)
        cam3.Priority =11;
        else
        cam3.Priority =9;
        if(on)
        {
           mandril.transform.Rotate(Vector3.up*mandrilSpeed*Time.deltaTime, Space.Self);
        }
        var speed = mandrilSpeed/1000;
        audioSource.pitch = speed;
        labelRpm.text = mandrilSpeed.ToString() + " Rpm";
    }
    public void ON()
    {
        on = true;
        buttonOff.image.sprite = spriteRedOff;
        buttonOn.image.sprite = spriteGreenOn;
        audioSource.Play();
```

```
    }
    public void OFF()
    {
        on = false;
        buttonOff.image.sprite = spriteRedOn;
        buttonOn.image.sprite = spriteGreenOff;
        audioSource.Stop();
    }
    private void OnEnable()
    {
      if(on)
      {
        audioSource.Play();
      }
    }
}
```

UiMecanizadosSelector

```
using UnityEngine;
using UnityEngine.UI;

public class UiMecanizadosSelector : MonoBehaviour
{
    [SerializeField]
    GameObject[] UiMecanizadosWindows;

    [SerializeField]
    Button[] UiMecanizadosButtons;

    [SerializeField]
    Color[] colorsForUi;
    // Start is called before the first frame update

    static public int indexMecanizado = 0;

    void Start()
```

```
{
    SelectUi(0);
}
public void SelectUi(int index)
{
    foreach (var item in UiMecanizadosButtons)
    {
        item.image.color = colorsForUi[0];
    }

    UiMecanizadosButtons[index].image.color = colorsForUi[1];

    foreach (var item in UiMecanizadosWindows)
    {
        item.SetActive(false);
    }
    indexMecanizado = index;

    switch (index)
    {
        default:
            UiMecanizadosWindows[0].SetActive(true);
            UiMecanizadosWindows[4].SetActive(true);
            UiMecanizadosWindows[2].SetActive(true);
            UiMecanizadosWindows[3].SetActive(true);
        break;
        case 0:
            UiMecanizadosWindows[0].SetActive(true);
            UiMecanizadosWindows[4].SetActive(true);
            UiMecanizadosWindows[2].SetActive(true);
            UiMecanizadosWindows[3].SetActive(true);
        break;

        case 1:
            UiMecanizadosWindows[0].SetActive(true);
            UiMecanizadosWindows[6].SetActive(true);
            UiMecanizadosWindows[4].SetActive(true);
```

```
            UiMecanizadosWindows[5].SetActive(true);


        break;
        case 2:
            UiMecanizadosWindows[7].SetActive(true);
            UiMecanizadosWindows[2].SetActive(true);
            UiMecanizadosWindows[3].SetActive(true);
        break;
      }
   }
}
```