

SEP

TNM

INSTITUTO TECNOLÓGICO DE CULIACÁN



Reconocimiento de gestos de la mano aplicado a una interfaz para
ambientes de aprendizaje

TESIS

PRESENTADA ANTE EL DEPARTAMENTO ACADÉMICO DE ESTUDIOS DE POSGRADO
DEL INSTITUTO TECNOLÓGICO DE CULIACÁN EN CUMPLIMIENTO PARCIAL DE LOS
REQUISITOS PARA OBTENER EL GRADO DE

MAESTRO EN CIENCIAS DE LA COMPUTACIÓN

POR:

Brandon Antonio Cárdenas Sainz
INGENIERO EN MECATRONICA

DIRECTOR DE TESIS:

Dr. Ramón Zatarain Cabada

CULIACÁN, SINALOA

Agosto, 2019



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



Instituto Tecnológico de Culiacán

"2019, Año del Caudillo del Sur, Emiliano Zapata"

Culiacán, Sin., 5 de Agosto del 2019

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN
OFICIO: DEPI: 330/VIII/2019

ASUNTO: Autorización Impresión

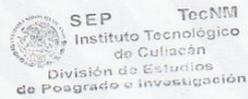
ING. BRANDON ANTONIO CÁRDENAS SAINZ
ESTUDIANTE DE LA MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN
PRESENTE.

Por medio de la presente y en virtud de que ha completado los requisitos para el examen de grado de la **Maestría en Ciencias de la Computación**, se concede autorización para la impresión de la tesis titulada: **"RECONOCIMIENTO DE GESTOS DE LA MANO APLICADO A UNA INTERFAZ PARA AMBIENTES DE APRENDIZAJE"** bajo la dirección del(a) **Dr. Ramón Zatarain Cabada**

Sin otro particular reciba un cordial saludo.

ATENTAMENTE
Excelencia en Educación Tecnológica®

M.C. MARÍA ARACELY MARTÍNEZ AMAYA
JEFE(A) DE LA DIVISIÓN DE ESTUDIOS DE
POSGRADO E INVESTIGACIÓN



C.c.p. archivo

MAMA/lucy *



Juan de Dios Bátiz 310 Pte. Col. Guadalupe, Culiacán, Sinaloa, C.P. 80220

Tel. 01 (667) 713-3804 y 713-1796

www.tecnm.mx | itculiacan.edu.mx



La Norma Mexicana NMX-R-021-SCFI, "Igualdad Laboral y No Discriminación" es el número de aplicación 0041, 072 y fecha de inicio 2017 (04-12) y pertenece de la serie de normas NMX-R-021-SCFI.



“RECONOCIMIENTO DE GESTOS DE LA MANO APLICADO A UNA INTERFAZ PARA AMBIENTES DE APRENDIZAJE”

Tesis presentada por:

ING. BRANDON ANTONIO CÁRDENAS SAINZ

Aprobada en contenido y estilo por:

Dr. Ramón Zatarain Cabada
Director de Tesis

Dra. María Lucía Barrón Estrada
Secretario

M.C. Rosalío Zatarain Cabada
Vocal -1

M.C. Gloria Ekaterine Peralta Peñuñuri
Vocal -2

M.C. María Aracely Martínez Amaya
Jefe(a) de la División de Estudios de
Posgrado e Investigación



Dedicatoria

Esta tesis es dedicada a mi madre Carmen Sainz de Anda, a mi Padre Antonio Cárdenas Ibarra y mi hermana Kenia Alejandra Cárdenas Sainz por todo su apoyo moral, motivación y empuje para superarme y mejorar día a día.

Agradecimientos

Agradezco a mis profesores Dra. Lucía Barrón, Dr. Ramón Zatarain Cabada, Dr. Héctor Rodríguez Rangel y al Dr. Ricardo Rafael Quintero Meza por compartir su experiencia y conocimiento. En especial a mi asesor de tesis el Dr. Ramón Zatarain por su guía, tiempo y dedicación en la preparación y desarrollo de este proyecto de investigación. Y a mis sinodales por darse la oportunidad, el tiempo y la dedicación para leer este trabajo. Agradecimientos a la MC. Gloria Ekaterine Peralta Peñúñuri por su apoyo y atención al realizar sus actividades como coordinadora.

Agradezco al Consejo Nacional de Ciencia y Tecnología CONACyT por el apoyo económico para llevar a cabo este proyecto de investigación, así como al Tecnológico de Culiacán por brindarme las oportunidades de curso y titulación de maestría.

Agradecimientos a la Dra. Juana Julieta Noguez Monroy y a su grupo de investigación de CyberLearning en el departamento de tecnologías de información y computación del Tecnológico de Monterrey Campus CDMX por aceptarme mi estancia de maestría y la valiosa experiencia y calidez en atención.

Agradezco la amistad y aprecio de mis compañeros de generación Héctor M. Cárdenas López, Jorge A. Romero Polo, Juan Ramon Valenzuela Barraza, Francisco Hernández González y Emmanuel Francisco Ramírez Hernández con quienes logré una buena dinámica de trabajo en equipo, quienes me compartieron sus experiencias y me ayudaron a aprender cosas nuevas.

Declaración de autenticidad

Por la presente declaro que, salvo cuando se haga referencia específica al trabajo de otras personas, el contenido de esta tesis es original y no se ha presentado total o parcialmente para su consideración para cualquier otro título o grado en esta o cualquier otra Universidad. Esta tesis es resultado de mi propio trabajo y no incluye nada que sea resultado de algún trabajo realizado en colaboración, salvo que se indique específicamente en el texto.

Brandon Antonio Cárdenas Sainz.

Culiacán, Sinaloa, México, 2019

Resumen

En este trabajo se presenta el prototipo de una aplicación web interactiva 3D, la cual combina dos interfaces en un ambiente de programación: la interfaz de programación visual mediante bloques de Google llamada Blockly, y la interfaz de interacción natural con implementación de reconocimiento de gestos de manos humanas.

El prototipo del ambiente de aprendizaje enfocado a la programación con bloques posee funciones de una herramienta de autor. Ofrece al usuario final la posibilidad de programar sus propios gestos en 3D, asociarlos a eventos e implementarlos dentro de un entorno gráfico, mientras se utiliza programación basada en bloques para la creación de escenarios interactivos con simulación de eventos físicos. El prototipo tiene la finalidad de desarrollar el pensamiento computacional del usuario y que éste se familiarice con diversos conceptos como son las interfaces humano máquina natural, la lógica algorítmica y el pensamiento espacial.

Se establece un trabajo enfocado al desarrollo de aspectos cognitivos que el sector educativo actual busca desarrollar: la mejora de habilidades y capacidades de lógica y abstracción a partir del desarrollo del pensamiento computacional del individuo, así como el desarrollo del pensamiento espacial para la construcción y la manipulación de las representaciones mentales de objetos en el espacio. Todo esto a partir de la implementación de interfaces naturales de usuario dentro de ambientes de aprendizaje, donde ambos conceptos están referidos a la percepción e interacción intuitiva o racional de un entorno, ya sea real o virtual, y de los objetos que hay en él.

Aplicando evaluaciones de diseño centrado en el humano, se demuestra que las interfaces interactivas e interfaces gráficas presentadas en este trabajo de tesis cumplen con los principios básicos de interacción natural y se tienen expectativas que indican que este enfoque impacta de forma positiva el proceso de aprendizaje, mejorando aspectos como son la motivación y disfrute del usuario.

Palabras clave

Interacción humano-máquina

Interfaces naturales de usuario

Diseño centrado en el humano

Entornos de aprendizaje interactivos

Pensamiento computacional

Pensamiento espacial

Reconocimiento de gestos

Interactividad kinestésica

Índice general

1. Introducción	1
1.1. Objetivo general.....	3
1.2. Objetivos específicos	3
1.3. Hipótesis.....	4
1.4. Justificación y contribuciones.....	4
1.5. Estructura de la tesis	5
2. Marco teórico	7
2.1. Pensamiento Computacional.....	7
2.2. Pensamiento espacial.	9
2.3. Interacción Humano-Computadora.....	10
2.4. Interfaz Natural de Usuario.....	13
2.5. Sensores.	14
2.5.1. Dispositivos sin visión.	15
2.5.2. Dispositivos con visión	16
2.5.3. Ventajas y desventajas entre las categorías de dispositivos.....	18
2.6. Tipo de gestos.	18
2.7. Modelos de representación.....	20
2.8. Controlador Leap Motion para interfaces naturales de usuario	21
2.8.1. Acceso a datos.....	21
2.8.2. Clase Hand.....	22
2.8.3. Clase Finger y Pointable.	23
2.8.4. Clase Bone.	24
2.8.5. Sistema de coordenadas.	25
2.9. Algoritmo \$P\$ Point-Cloud para el Reconocimiento de gestos.	26
2.10. Interfaz Gráfica de Usuario.....	28
2.10.1. Blockly	28
2.10.2. Babylon js	29
3. Estado del arte.....	31
3.1. Ambientes de aprendizaje orientados al desarrollo del pensamiento computacional	31
3.2. Aplicaciones interactivas orientadas al aprendizaje.....	40
3.2.1. Sobre realidad virtual.	41
3.2.2. En música.	43
4. Desarrollo del proyecto.....	46

4.1.	Diseño centrado en el humano.....	46
4.2.	Actores.....	48
4.3.	Análisis de requerimientos.....	49
4.4.	Diagrama de contexto.....	49
4.5.	Arquetipos.....	50
4.6.	Arquitectura.....	51
4.7.	Desarrollo de interfaz NUI con reconocimiento de gestos.....	53
4.8.	Desarrollo de interfaz NUI con interactividad kinestésica.....	55
4.9.	Desarrollo de prototipo de aplicación Web.....	56
4.9.1.	Algoritmo para Reconocimiento de Gestos en interfaces 3D.....	58
4.9.2.	CREA tus propios gestos.....	65
4.9.3.	JUEGA y diviértete.....	66
5.	Pruebas.....	68
5.1.	Caso de estudio: Creación y ejecución de reconocimiento de gestos.....	68
5.2.	Caso de estudio: Uso de interfaces de usuario naturales para la interacción dentro de un entorno de aprendizaje.....	69
5.3.	Pruebas de validez de GUI y NUIs del sistema.....	71
6.	Conclusiones y trabajo futuro.....	74
6.1.	Conclusiones.....	74
6.2.	Trabajo a futuro.....	75
	Bibliografía.....	76

Índice de figuras

Figura 2-1. Representación del pensamiento computacional como 3 etapas (Repenning et al., 2016).	8
Figura 2-2. Etapas del pensamiento espacial.	9
Figura 2-3. El ciclo de interactividad entre en hombre y la computadora (Bachmann, Weichert, & Rinkenauer, 2018).	10
Figura 2-4. Principios entre el hombre y la máquina (Bachmann et al., 2018).	11
Figura 2-5. Tópicos del HCI (Castro, Rodríguez, 2018).	12
Figura 2-6. Clasificación de tipo de sensores.	14
Figura 2-7. Ejemplos de dispositivos sin visión (wearables o de contacto).	15
Figura 2-8. Ejemplos de dispositivos de visión.	17
Figura 2-9. Clasificación de gestos (Aigner et al., 2012).	19
Figura 2-10. Representación de gestos de la mano a partir de sistemas de visión.	21
Figura 2-11. Estructura de datos del contenedor Frame (Davis, 2014).	22
Figura 2-12. Representación visual de los datos palmNormal y grabStrength (Davis, 2014).	22
Figura 2-13. Representación visual de <i>tipDirection</i> de cada dedo (Davis, 2014).	24
Figura 2-14. Visualización del campo de visión del sensor LMC.	26
Figura 2-15. Representación gráfica del proceso de comparación entre 2 nubes de puntos (Vatavu et al., 2012).	27
Figura 2-16. Pasos del proceso de reconocimiento en \$P recognizer. GSS significa Golden Section Search (Vatavu et al., 2012).	28
Figura 2-17. Ejemplo interfaz hecha en Blockly.	29
Figura 2-18. Editor de objetos 3D hecho en Babylon JS	30
Figura 3-1. Interfaz de programación en bloques en Scratch	32
Figura 3-2. Interfaz de programación en bloques en Alice 3	32
Figura 3-3. Interfaz de desarrollo de bloques en Snap!	33
Figura 3-4. Interfaz de MIT App Inventor	34
Figura 3-5. Ejemplo de aplicación en Pocket Code	34
Figura 3-6. GUI predeterminada de Tynker	35
Figura 3-7. GUI en GameMakerStudio 2	36
Figura 3-8. GUI en BeetleBlocks	37
Figura 3-9. GUI de AgentCubes	38
Figura 3-10. GUI de TurtleArt	39
Figura 4-1. Fases del proceso de diseño del DCH	46
Figura 4-2. Diagrama de contexto	50
Figura 4-3. Diagrama general de componentes	52
Figura 4-4. Archivo JSON como contenedor de gestos	54
Figura 4-5. Representación de las áreas de interacción con respecto a la profundidad entre el usuario y el LMC.	55
Figura 4-6. Representación de un gesto mediante trazos	58
Figura 4-7. Entorno de Desarrollo de CREA	65
Figura 4-8. Entorno de programación visual con Blockly	66
Figura 4-9. Escena 3D con interacción kinestésica desde un avatar	67
Figura 5-1. Segmentos para programar un entorno interactivo en JUEGA y diviértete	70

Índice de tablas

Tabla 2-1. Comparativa entre dispositivos con y sin visión.	18
Tabla 2-2. Valores de Finger y Pointable asociados a datos vectoriales de dedos	23
Tabla 2-3. Valores para acceso de datos vectoriales de Bone.....	24
Tabla 3-1. Herramientas de autor para el desarrollo del pensamiento computacional	40
Tabla 3-2. Proyectos interactivos enfocados al aprendizaje con LMC.....	42
Tabla 3-3. Proyectos interactivos educativos musicales	44
Tabla 4-1. Actores participantes de la aplicación	48
Tabla 4-2. Lista de requisitos funcionales.....	49
Tabla 4-3. tabla de componentes de la capa lógica.....	53
Tabla 5-1. Resultados de evaluación de aspectos de las interfaces del prototipo.	72

Índice de fórmulas

Fórmula (1)	59
Fórmula (2)	59
Fórmula (3)	60
Fórmula (4)	61
Fórmula (5)	61
Fórmula (6)	61
Fórmula (7)	62

Capítulo 1

1. Introducción

El sector educativo, busca más que nunca en la actualidad, diversas formas para el desarrollo del pensamiento computacional. Éste es una serie de habilidades que integran la resolución de problemas, el diseño de sistemas y algoritmos, así como el entendimiento del comportamiento humano, aplicando para ello conceptos estructurales de la computación. Este proceso se enfoca en el reconocimiento de aspectos de la informática en el mundo que nos rodea, y aplica herramientas y técnicas de la informática para comprender y razonar sobre los sistemas y procesos tanto naturales como artificiales (Grover & Pea, 2013).

Usualmente, cuando se habla de informática, computación, o de temas como la programación e interacción con una computadora, se visualiza a una persona sentada frente a una computadora, la cual está escribiendo en un teclado y moviéndose en pantalla por medio de un ratón o touchpad. Esta forma de interactuar con las computadoras no ha cambiado significativamente desde 1960 (Preece et al., 1994), época en la que estos periféricos fueron inventados.

Actualmente existen diversos dispositivos en el mercado tales como tabletas y teléfonos inteligentes, en donde se presentan interfaces más innovadoras: pantallas con capacidad táctil y el uso de gestos que representan una forma más natural de interacción entre el humano y la máquina. También se han realizado diversas implementaciones de reconocimiento de gestos en dispositivos con sensores o cámaras con la finalidad de ofrecer cierta simplicidad de uso (W. Lin, Du, & Harris-adamson, 2017).

El reconocimiento de gestos es un tema en las ciencias de la computación y de la tecnología del lenguaje con el objetivo de interpretar gestos humanos a través de algoritmos matemáticos. Los gestos pueden ser cualquier movimiento corporal o estado, pero comúnmente se originan a partir de un rostro o de las manos. El reconocimiento de gestos puede ser visto como la manera en que las computadoras interpretan el lenguaje corporal humano, considerándose como una interfaz natural humano-computadora.

Esto nos lleva a los conceptos de interacción humano-máquina (HMI por sus siglas en inglés) y las interfaces naturales de usuario (NUI por sus siglas en inglés), donde esta última define que los humanos se pueden comunicar con la máquina e interactuar naturalmente sin dispositivos mecánicos (Preece et al., 1994). Utilizando el concepto de reconocimiento de gestos, es posible usar los dedos en un espacio libre para relacionar el movimiento del cursor con los movimientos del usuario.

Actualmente, la presencia de interfaces humano-máquina más naturales se ha vuelto una necesidad, ya que estas pueden proporcionar nuevas capacidades y facilidad de uso en diversos dispositivos computacionales, ya sean computadoras, teléfonos inteligentes, tabletas, máquinas del sector industrial, etc. Estas interfaces ofrecen al usuario mejorar el tiempo de aprendizaje, la velocidad de desempeño, la tasa de error de uso, la forma en que un usuario opera el software de un dispositivo y la satisfacción que se experimenta en el uso (Wigdor & Wixon, 2011).

Por otra parte, el desarrollo del pensamiento computacional en un individuo para que éste adquiera la habilidad de crear programas para una computadora sigue siendo hoy en día un problema abordado por diversas investigaciones. Desde la creación de herramientas digitales llamadas e-learning (Cabero, 2006) para facilitar el aprendizaje, sistemas o mecanismos de asistencia en el aprendizaje de acuerdo al rendimiento del individuo como los Sistemas Tutores Inteligentes (STI) (Wenger, 2014), la comprensión de la conducta humana y de sus estados emocionales y cómo éstos afectan su rendimiento a la hora de aprender como los Sistemas Tutores Afectivos (STA) (H.-C. K. Lin, Wu, & Hsueh, 2014), son algunos de los enfoques que ofrecen diversas soluciones.

Uno de estos enfoques aborda el cómo un usuario podría interactuar con un dispositivo de cómputo, herramientas en las cuales se realiza el ejercicio cognitivo computacional.

Como se mencionaba anteriormente, algunos estudios indican que la utilización de interfaces humano-máquina con un enfoque a la interacción natural ayuda a la visualización, la comprensión y por ende el aprendizaje del usuario (Shneiderman & Plaisant, 2006) por lo que puede ser de gran ayuda para el sector educativo.

Por ello, esta investigación propone la exploración de nuevas formas de interacción implementadas en entornos de aprendizaje, así como el impacto sobre el proceso de aprendizaje, aumentando la motivación, el disfrute del estudio y el desarrollo del pensamiento computacional.

1.1. Objetivo general

El objetivo principal de la presente investigación se enfoca en desarrollar un prototipo de un ambiente de aprendizaje usando Blockly y un sistema de reconocimiento de gestos de la mano, para la programación de escenarios interactivos acerca de un sistema de simulación de eventos físicos. El sistema debe satisfacer las características básicas que componen a un entorno de aprendizaje orientado al desarrollo del pensamiento computacional, mediante el uso de interfaces de programación gráfica por bloques. Dicho ambiente de aprendizaje se complementará con la implementación de una interfaz natural de usuario compuesta por un sistema de reconocimiento de gestos e interactividad kinestésica, para proporcionar interacción natural con los elementos dentro del entorno.

1.2. Objetivos específicos

El objetivo principal se dividió en ocho objetivos específicos que se detallan a continuación:

1. Estudiar el estado del arte con respecto al uso de reconocimiento de gestos.
2. Estudiar las API de desarrollo de dispositivos y aplicaciones gestuales que existen actualmente.
3. Diseñar un entorno interactivo web que implemente Blockly y que posea las características básicas para la implementación de interfaces naturales de usuario.
4. Implementar algoritmos de reconocimiento de gestos, tales como reconocimiento de patrones y extracción de características.
5. Identificar características de los movimientos de las manos para su tratamiento.

6. Desarrollar la interfaz NUI con el sistema de reconocimiento de gestos de interactividad kinestésica.
7. Implementar representación gráfica de las manos del usuario para el entorno de aprendizaje en tiempo real.
8. Probar la usabilidad y satisfacción del usuario a dichas interfaces.

1.3. Hipótesis

La implementación de interfaces naturales de usuario dentro de un ambiente de aprendizaje, compuesto por interacción kinestésica y reconocimiento de gestos, así como las interfaces gráficas pensadas al desarrollo del pensamiento computacional y espacial, en conjunto cumplen con los aspectos de validez de interactividad natural necesarios para producir impactos positivos en factores como la usabilidad, motivación y satisfacción de uso del usuario.

Por lo tanto la hipótesis de este trabajo de investigación se establece a continuación: Es posible generar una herramienta para producir aplicaciones que permitan desarrollar el pensamiento computacional y espacial incluyendo interacción kinestésica y reconocimiento de gestos, que en conjunto cumplan con los aspectos de validez de interactividad natural requeridos en una herramienta de autor para aplicaciones educativas.

1.4. Justificación y contribuciones

Hoy en día, en donde las nuevas generaciones están inmersas en el uso de la tecnología, se siguen buscando diferentes formas para aprovechar el interés de diversas tecnologías para enfocar su uso en el ámbito educativo. La introducción de nuevas tecnologías dentro del campo educativo tiene el propósito de proveer nuevas formas que contribuyan a lograr el aprendizaje, así como un impactar positivamente en la motivación por aprender.

El presente trabajo, se enfoca en la interactividad de los usuarios con los entornos de aprendizaje, presenta una solución que intenta ofrecer al usuario la capacidad de interactuar

de forma natural con el software, con el fin de minimizar la dificultad, mejorar la comprensión y visualización de conceptos que se enseñan en una aplicación.

El ambiente de aprendizaje posee una interfaz de programación por bloques para la enseñanza de la lógica algorítmica y la abstracción de la programación, donde el usuario programa animaciones e interactividad entre objetos tridimensionales mediante un sistema de simulación de eventos físicos, representado dentro de un escenario interactivo 3D.

La característica de interactividad dentro de la escena 3D está dada mediante la implementación de una interfaz de interactividad natural compuesta por un sistema de reconocimiento de gestos y un sistema de interactividad kinestésica. Pretende demostrar que la implementación de nuevas tecnologías de interfaces hombre-máquina afecta positivamente el aprendizaje del usuario para usar la herramienta y, por ende, los conceptos que instruye el entorno de aprendizaje en sí.

La incorporación de interfaces naturales de usuario permite al entorno de aprendizaje reducir la dificultad de uso y mejorar la visualización y comprensión de conceptos que el usuario está por aprender.

1.5. Estructura de la tesis

En esta sección, se presentan los capítulos que componen esta tesis, describiendo brevemente su contenido.

Capítulo 1. Introducción. Este capítulo introduce los principales motivos que han llevado a la realización de esta tesis, así como la problemática e importancia de abordar este problema con un nuevo enfoque.

Capítulo 2. Marco teórico. Este capítulo describe los diversos temas requeridos para el desarrollo de este trabajo entre los que se encuentran: pensamiento computacional, interacción humano-máquina, sensores, gestos con las manos, etc.

Capítulo 3. Estado del arte. En este capítulo se muestran trabajos relacionados al desarrollo del pensamiento computacional y aplicaciones interactivas orientadas al aprendizaje.

Capítulo 4. Desarrollo del proyecto. En este capítulo se documenta el desarrollo del proyecto centrado en el humano (DCH) así como los pasos necesarios para el análisis de requerimientos y los casos de estudios pertinentes para el desarrollo de interfaces NUI y GUI de la aplicación.

Capítulo 5. Pruebas. Este capítulo analiza y discute en profundidad los resultados obtenidos en la evaluación presentada en el capítulo anterior.

Capítulo 6. Conclusiones y trabajo futuro. Este capítulo recopila las diferentes conclusiones extraídas del trabajo realizado, y propone algunas líneas de trabajo futuro.

Finalmente, la última sección corresponde a las referencias.

Capítulo 2

2. Marco teórico

En este apartado se presenta la parte teórica que da sustento a esta propuesta de investigación. Se exponen los principales conceptos relacionados a este campo de estudio como punto de partida para dar forma a la solución del problema. En este capítulo se abordan los temas de pensamiento computacional, pensamiento espacial e interfaces humano-maquina naturales.

2.1. Pensamiento Computacional

Jeannette Wing es la promotora del pensamiento computacional, su visión sobre ello es que, al igual que el conocimiento del idioma o la aritmética, el pensamiento computacional debería de ser una habilidad y una actitud de aplicación universal para todas las personas. Wing propone que las habilidades de abstracción y las técnicas para la resolución de problemas, utilizados por aquellos que practican las ciencias e ingenierías computacionales, deben de enseñarse y aplicarse en las disciplinas y actividades de la vida cotidiana.

El objetivo del pensamiento computacional es desarrollar sistemáticamente las habilidades del pensamiento superior, la abstracción, el pensamiento crítico y la resolución de problemas, con base de los conceptos de la computación (Wing, 2006).

Por lo general, las características que definen el pensamiento computacional son la descomposición, el reconocimiento de patrones, la abstracción y los algoritmos. Esto se refiere a las fases para descomponer un problema, identificar las variables involucradas a partir de la representación de datos/patrones, y la creación de algoritmos para finalmente obtener alguna solución genérica. Esta solución genérica es la generalización o la abstracción que puede ser utilizada para resolver múltiples variaciones del problema inicial.

Cabe aclarar que el pensamiento computacional no es el desarrollo de habilidades como la programación, sino al desarrollo y el ejercicio de capacidades de conceptualización del humano. Describe una forma de pensamiento a múltiples niveles de abstracción, no solo la

habilidad de programar. En otras palabras, es pensar con la computadora, no pensar como una.

El pensamiento computacional es considerado y utilizado como una forma de pensar, la cual utiliza a los dispositivos de cómputo como instrumentos que le dan soporte al proceso del pensamiento humano, con la finalidad de visualizar las consecuencias de dicho proceso de pensamiento y formular un problema, donde se pueda obtener una solución a partir de las capacidades de la computadora (Repenning, Basawapatna, & Escherle, 2016). Como se puede ver en la Figura 2-1, basándose en las definiciones del proceso del pensamiento computacional de Wing, éste puede ser segmentado en 3 etapas:

1. Formulación del problema (Abstracción): La formulación de una pregunta. El individuo visualiza el funcionamiento de algo, descomponiéndolo e identificando las variables involucradas, así como sus relaciones y patrones.
2. Expresión de la solución (Automatización): Etapa donde se determina la expresión no ambigua de la solución, de tal forma que pueda ser procesada por la computadora, mediante programación.
3. Ejecución y evaluación de la solución (Análisis): Ejecución de soluciones mediante la presentación de la computadora mostrando las consecuencias directas del pensamiento humano.

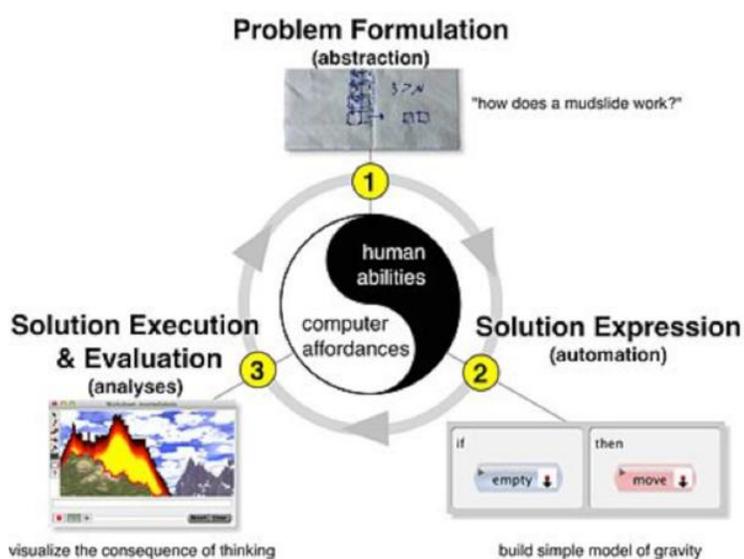


Figura 2-1. Representación del pensamiento computacional como 3 etapas (Repenning et al., 2016).

2.2. Pensamiento espacial.

El pensamiento espacial se considera como el conjunto de procesos cognitivos para la construcción y representación mental del espacio y de los objetos que se encuentran dentro del mismo, así como las relaciones y las transformaciones que existen entre ellos. (ver Figura 2-2).

La visualización espacial es un aspecto importante del pensamiento humano, se entiende como la construcción y la manipulación de representaciones mentales de objetos de dos y tres dimensiones y a la percepción de los objetos desde diferentes perspectivas.

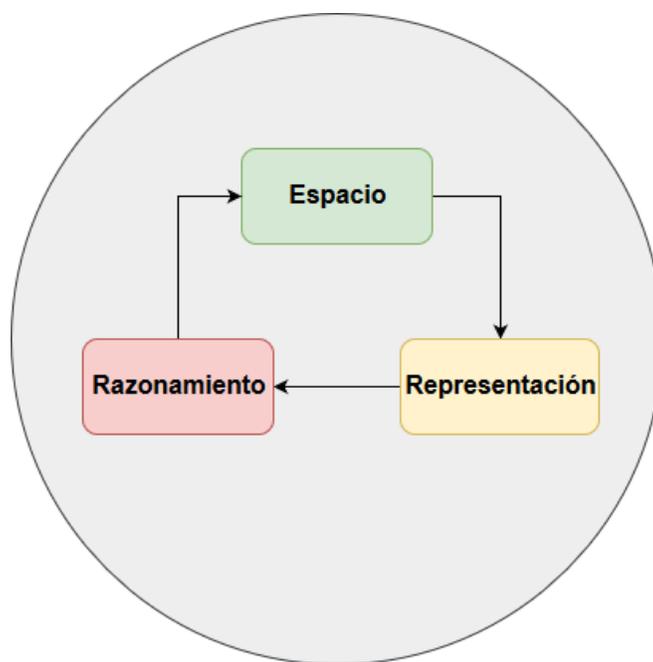


Figura 2-2. Etapas del pensamiento espacial.

El manejo de información espacial para resolver problemas de ubicación, orientación y distribución de espacios es peculiar a las personas que tienen desarrollada la inteligencia espacial. Se estima que la mayoría de las profesiones científicas y técnicas, tales como el dibujo técnico, la arquitectura, las ingenierías, la aviación, y muchas disciplinas científicas como química, física y matemáticas, requieren personas que tengan un alto desarrollo de inteligencia espacial (Mathewson, 1999).

Este tipo de pensamiento es soportado por los sistemas geométricos. En estos sistemas, se construyen conceptos y abstracciones geométricas espaciales a través de la exploración y

representación del espacio tanto para objetos en reposo o en movimiento. Esta construcción se entiende como un proceso cognitivo interactivo, que avanza de forma intuitiva y sensoriomotora, puesto que esto se relaciona con la capacidad práctica y natural de interacción con el entorno mediante manipulación de objetos, así como la ubicación y localización de los mismos mientras se efectúan cálculos espaciales y/o mediciones, etc.

2.3. Interacción Humano-Computadora.

La interfaz de usuario es el medio donde se desarrolla la interacción ente el humano y la computadora. La Interacción Humano-Computadora (IHC) es la disciplina dedicada al diseño, evaluación e implementación de sistemas informáticos interactivos para la interactividad del humano y así, estudiar la relación existente entre el hombre y la máquina.

El objetivo principal de las interfaces es abstraer el funcionamiento de una computadora en una orden o un conjunto de ellas, con el propósito de evitarle al usuario la necesidad de saber el funcionamiento interno de dicho sistema (ver Figura 2-3), La interacción puede ser definida como el intercambio entre dos o más participantes activos.

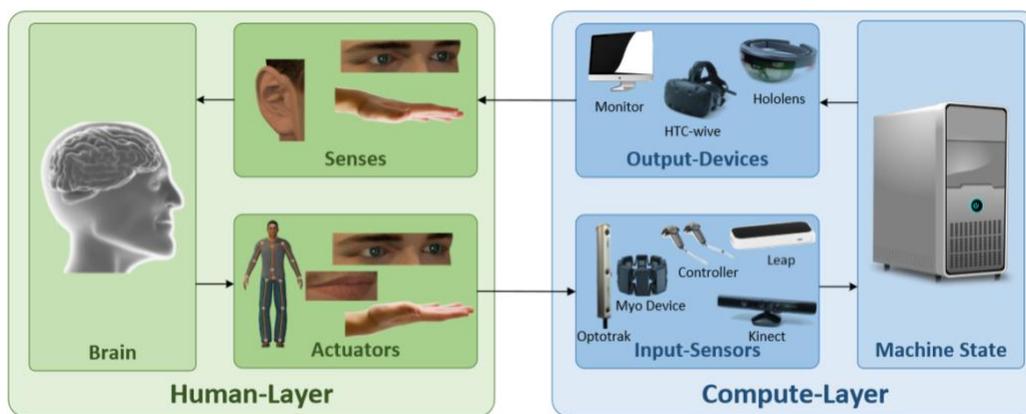


Figura 2-3. El ciclo de interactividad entre en hombre y la computadora (Bachmann, Weichert, & Rinckenauer, 2018).

En términos de la programación de interfaces, la interacción está dada entre un sistema computacional y el usuario (ver Figura 2-4). Existe una condición importante, debe de existir en todo momento la retroalimentación en cuanto la interacción entre el sistema y el usuario se presente. Si no se presenta ninguna retroalimentación, esto causaría una confusión al

usuario sobre el funcionamiento del sistema (Bachmann, Weichert, & Rinkenauer, 2018). Esto podría significar que el sistema funciona de forma incorrecta o que no se presenta una respuesta correcta.

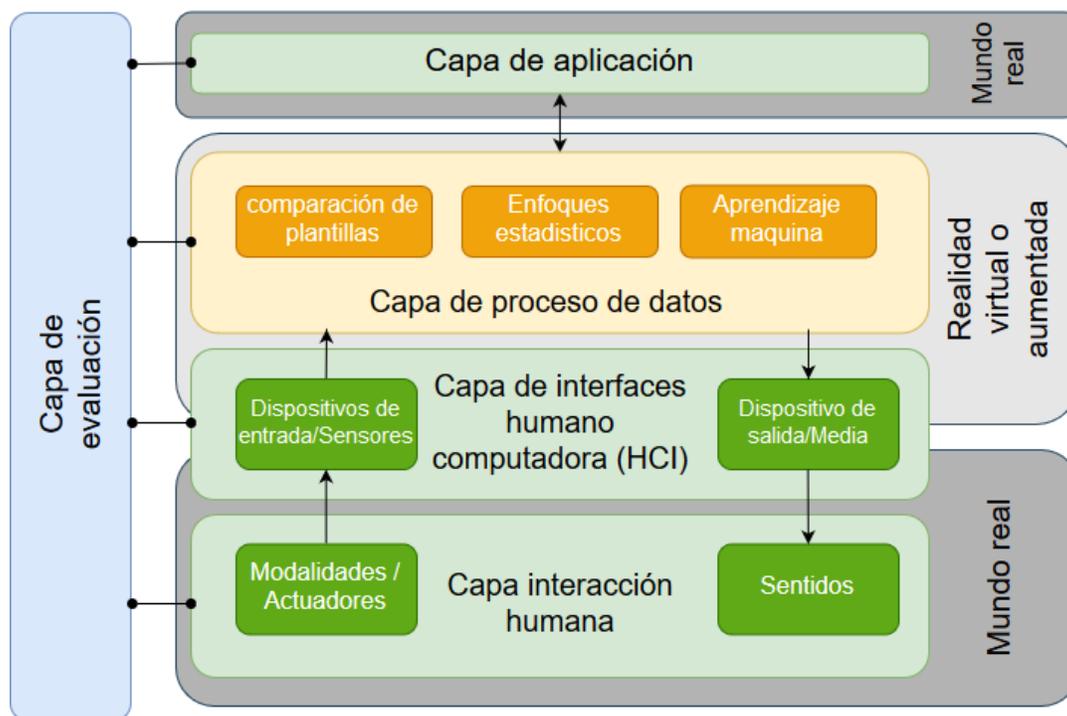


Figura 2-4. Principios entre el hombre y la máquina (Bachmann et al., 2018).

En esta área el actor principal es el usuario, ya que se requiere entender su comportamiento durante el proceso de construcción del software para comprender los procesos y capacidades que están asociados a las tareas que estos desempeñan (Preece et al., 1994).

El uso de HCI, para elaborar sistemas computacionales genera un grado de eficiencia mediante la reducción del número de errores ocasionados por el usuario durante el desarrollo de una tarea. De la misma forma se logra eficiencia, en términos del tiempo que requiere el usuario en aprender a utilizar el sistema, el tiempo de realización de cierta tarea y esto en consecuencia, logra un incremento en la satisfacción de uso.

La figura 2-5 muestra los tópicos importantes del HCI: la consideración de las capacidades y limitaciones de la tecnología (Computadora), la aplicación sistemática de conocimientos sobre las características y metas humanas (proceso de desarrollo), la relación entre los

aspectos sociales, organizacionales y físico del entorno de trabajo del usuario (uso y contexto), para que finalmente el software pueda ofrecer un sistema de calidad al actuador principal (usuario).

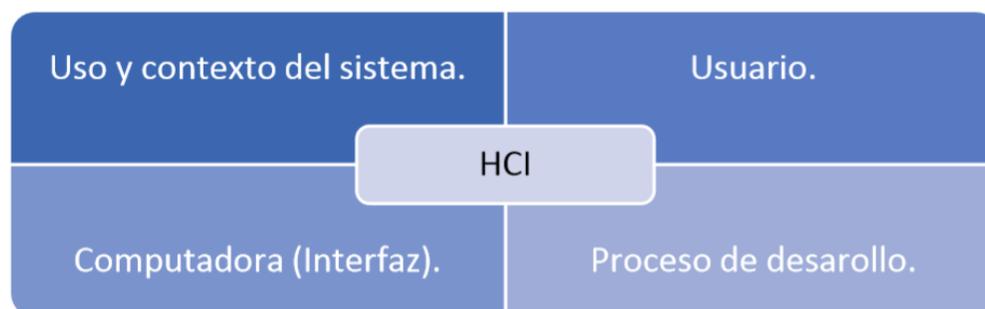


Figura 2-5. Tópicos del HCI (Castro, Rodríguez, 2018).

HCI es el diseño e implementación de sistemas informáticos interactivos para que los usuarios puedan llevar a cabo el proceso de interacción. Esto incluye a sistemas de escritorio, así como los sistemas integrados en diferentes dispositivos (Castro, Rodríguez, 2018).

El éxito de una tecnología es el resultado de la facilidad con la que el usuario puede interactuar con la misma. Si la interfaz es pobre o difícil de usar, el usuario simplemente ignorará el producto o la tecnología. Es por eso la importancia de que, al crear un sistema, la implementación de HCI sea realizada teniendo en cuenta la funcionalidad y facilidad del uso.

Para el usuario, lo importante es poder realizar sus actividades de manera eficiente, sin tener que pasar por un entrenamiento previo sobre comandos básicos, medios y avanzados de cómo utilizar el software, es por tal motivo que se está usando esta área como apoyo para realizar interfaces de lenguaje natural (NUI).

De ahí que exista la necesidad de los elementos como la usabilidad y amigabilidad de un programa de cómputo proporcionando un fácil acceso a los usuarios. Cuando se refiere a amigabilidad se consideran los factores de tiempo de aprendizaje, la velocidad de desempeño, la tasa de error de uso, el tiempo del usuario, la forma en que el usuario opera el software y la satisfacción que se experimenta en el uso (Shneiderman & Plaisant, 2006).

2.4. Interfaz Natural de Usuario.

La Interfaz Natural de Usuario (NUI por sus siglas en inglés) es el siguiente cambio en el paradigma metafísico sobre la interacción entre el hombre y la máquina. Ésta busca investigar modelos y técnicas computacionales inspiradas en la naturaleza de la interactividad del humano, pretende entender cómo el hombre realiza el procesamiento de la información de su entorno, utilizando tecnologías emergentes donde las habilidades y sensaciones son mucho más precisas y optimizadas para la interacción entre objetos físicos y digitales.

Está definida en términos de experiencia: cómo el humano se comunica por medio de gestos, expresiones y movimientos y cómo éste descubre el mundo mientras observa su alrededor manipulando objetos físicos.

Las NUI son interfaces diseñadas para que el usuario actúe y se sienta natural durante la interacción. Dependen de que el usuario sea capaz de aprender rápidamente el uso de una interfaz, sin la necesidad de utilizar sistemas de mando o dispositivos de entrada, como ratones, teclado, o lápiz óptico. En su lugar, se utilizan partes del cuerpo, como las manos o las yemas de los dedos, las cuales son los medios para el control del sistema y/o aplicación.

La NUI surge motivada por el estudio e investigación de la forma en que se interactúa con el ambiente de trabajo en el contexto computacional y las interfaces naturales del usuario, en las cuales buscan reemplazar las interfaces de usuario actuales, por mecanismos que permitan interactuar con la computadora de la misma forma en la que se interactúa diariamente con el entorno, es decir, volviendo transparentes los dispositivos de entrada y facilitando el aprendizaje al usuario para interactuar mediante la interfaz (Jain, Lund, & Wixon, 2011).

Diseñar interfaces NUI de forma correcta implica involucrarse con el usuario de forma más directa. Esto implica, analizar y evaluar las diferentes formas en que el usuario puede interactuar con el sistema, y tomar decisiones de diseño que logren un producto que genere una experiencia agradable. Esta consideración refleja que es más complejo diseñar interfaces NUI que diseñar las tradicionales interfaces GUI, lo cual obliga a tomar en cuenta una serie de consideraciones y criterios para lograr una aplicación exitosa, enfatizando en una mejor calidad de experiencia de usuario, con respecto a la usabilidad.

Para que una interfaz sea considerada natural, debe cumplir con las siguientes consideraciones (Santana, 2014):

- Crear una experiencia que de la sensación de ser una extensión del cuerpo.
- Crear una experiencia que le sea natural tanto a usuarios expertos, como a los usuarios nuevos.
- Crear una experiencia que sea auténtica al medio.
- Crear una interfaz de usuario que considere el contexto, tomando en cuenta las metáforas correctas, indicaciones visuales, realimentación y métodos de entrada y salida.
- Evitar caer en la trampa de copiar los paradigmas de interfaz de usuario existentes.

2.5. Sensores.

Hay varios tipos de dispositivos que son capaces de llevar a cabo un seguimiento en los movimientos de una persona y determinar el gesto que realiza.

A continuación, en la Figura 2-6 se presenta dos categorías de los planteamientos existentes en reconocimiento de gestos humanos de acuerdo al tipo de sensor usado:

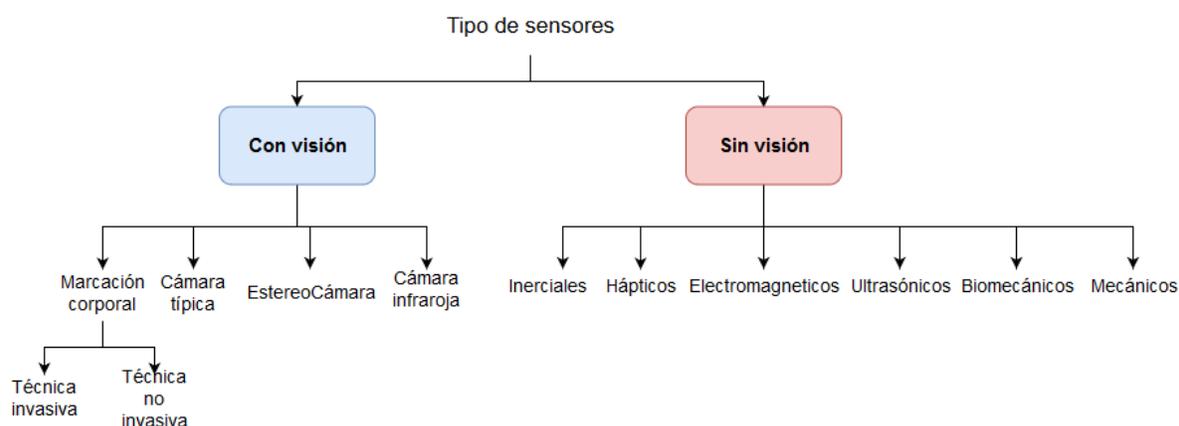


Figura 2-6. Clasificación de tipo de sensores.

2.5.1. Dispositivos sin visión.

Los dispositivos sin visión son tecnologías que utilizan sensores para rastreo y detección de movimiento, estas incluyen acelerómetros, sensores electromagnéticos, sensores hápticos, inerciales, magnéticos, ultrasónicos, etc. (Rautaray & Agrawal, 2015).

Por lo general se presentan en dispositivos basados en contacto, casos populares son en forma de guantes, pantallas multitouch, joysticks, brazaletes, etc. (ver Figura 2-7).



Figura 2-7. Ejemplos de dispositivos sin visión (wearables o de contacto).

Mecánicos: este tipo de dispositivos por lo general se encuentran en forma de prenda de vestir (wearable), la cual incluye múltiples sensores como potenciómetros, giroscopios y rastreadores magnéticos necesarios para el reconocimiento de movimientos examinados en una parte del cuerpo. Se encuentran en forma de guantes (CyberGlove), brazaletes (Myo) o traje completo (IGS-190).

Biomecánicos: Dispositivos en los cuales se utilizan técnicas biomecánicas como la electromiografía, para medir los parámetros del gesto. Consisten en la adquisición, registro

y análisis de la actividad eléctrica generada en los nervios y músculos a través de la utilización de electrodos (superficiales, de aguja, implantados).

Inerciales: Estos dispositivos miden la variación del campo magnético de la tierra para detectar el movimiento. Este tipo de dispositivos utilizan acelerómetros y giroscopios para las mediciones. Los teléfonos inteligentes (*smartphones*) modernos, así como dispositivos para juegos (Wii mote y PS move, etc.) son ejemplo de ello.

Háptico: esta tecnología se puede observar en *gadgets* que se utilizan en la vida cotidiana moderna (tabletas, PC, *smartphones*, etc.). Dispositivos que poseen pantallas táctiles que ofrecen capacidades de reconocimiento e interactividad mediante gestos multi-touch,

Electromagnéticos: Estos dispositivos miden la variación de un campo electromagnético artificial generado por redes inalámbricas, dispositivos electrónicos o producidos por ellos mismos con la finalidad de detectar movimiento.

Ultrasónicos: Proveen rastreo de movimiento mediante sistemas que emiten y reciben frecuencias de sonido ultrasónicas, miden el tiempo de retorno de pulsos y la obstrucciones encontradas.

2.5.2. Dispositivos con visión

Los dispositivos con visión son tecnologías de detección de movimiento a partir del uso de una o varias cámaras de video, en conjunto con otros elementos como sensores infrarrojos, marcas de color RGB marcadores laser los cuales en conjunto ayudan a la asistencia de la detección de movimientos. (Rautaray & Agrawal, 2015) En la Figura 2-8 se muestran algunos ejemplos de dispositivos que utilizan cámaras como medio sensorial.



Figura 2-8. Ejemplos de dispositivos de visión.

Estos dispositivos incluyen uno o varias cámaras con el fin de proveer datos a partir de las secuencias de video capturadas. El procesamiento de *frames* está basado en filtros, análisis e implementación de datos. Es posible distinguir los siguientes tipos de tecnologías basadas en visión:

Cámaras de video típicas: técnicas de reconocimiento de gestos basadas en los datos obtenidos por una cámara monocular usando métodos de detección de colores o formas, detectores de aprendizaje a partir de valores de píxeles o detección basada en modelos 3D.

Estereocámaras: técnicas de captura de imágenes a partir de 2 cámaras, los cuales proveen una aproximación de los datos grabados para poder representar un modelo 3D.

Cámaras infrarrojas: Comúnmente usadas para visión nocturna, las cámaras infrarrojas generalmente dan una visión más clara de la silueta del cuerpo humano.

Sensores de marcadores corporales: Algunos sistemas requieren de aplicar marcadores en el cuerpo con el fin de detectar movimiento del cuerpo humano. Existen dos técnicas de marcadores: invasivas y no invasivas. Las técnicas no invasivas son aquellas donde se realiza alguna forma de proyección de luz estructurada. Kinect de Microsoft o el uso de luces

infrarrojas por parte de Leap Motion son un ejemplo de ello. Las técnicas invasivas requieren el uso de marcadores en el cuerpo para poder detectar el movimiento del cuerpo humano, estos pueden ser etiquetas de colores o luces LED.

2.5.3. Ventajas y desventajas entre las categorías de dispositivos.

Ambos enfoques tecnológicos tienen sus pros y contras. Por ejemplo, dispositivos de contacto (wearables) durante largos periodos de uso pueden ser incómodos. Dispositivos basados en visión no requieren de cooperación del usuario, pero son más difíciles de configurar y pueden sufrir problemas de obstrucción.

La Tabla 2-1 hace un resumen comparativo de los criterios de ambos enfoques. Por ejemplo, dispositivos sin visión (de contacto) son más precisos a excepción de los ultrasónicos. Además, estos no poseen problemas de oclusión a excepción de los sensores magnéticos (obstáculos metálicos) y los sensores ultrasónicos. Sobre el criterio de problemas de salud, algunos dispositivos de contacto provocan alergia por materiales y en el uso de dispositivos magnéticos existe el riesgo de cáncer (Rautaray & Agrawal, 2015).

Tabla 2-1. Comparativa entre dispositivos con y sin visión.

Criterio	Dispositivos de visión	Dispositivos sin visión
Cooperación del usuario	no	si
Intrusivo con el usuario	no	si
Precisión	No/si	Si/no
Configuración flexible	no	si
Fácil de usar	si	no
Problemas de oclusión	si	no(si)
Problemas de salud	no	Si(no)

2.6. Tipo de gestos.

En la figura 2-9, se muestra una taxonomía de los tipos de gestos clasificados por (Aigner et al., 2012) los cuales estudian las preferencias del humano en el uso de gestos de la mano cuando ésta se encuentra en el aire.

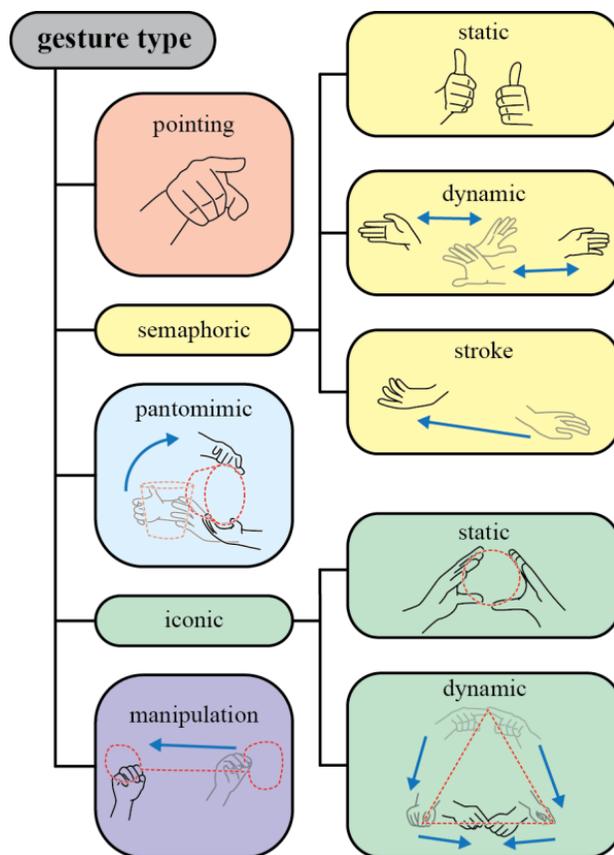


Figura 2-9. Clasificación de gestos (Aigner et al., 2012).

A continuación, se describe cada uno de los tipos de gestos presentados en la clasificación de Aigner.

Apuntar (Pointing): Uso de objeto punto o indicador de dirección. Formalmente consiste en apuntar y establecer la identidad o la localización espacial de un objeto en el contexto del dominio de la aplicación. Esto no solo implica el uso del dedo índice de una mano, sino de cualquier dedo o cualquier número de dedos a la vez.

Semafóricos (Semaphoric): Grupo que consiste en la postura gestual y la dinámica de los gestos, los cuales con utilizados para determinar significados específicos. Formalmente pueden denominarse como “comunicativos” en el sentido en que los gestos sirven como un universo de símbolos que deben comunicarse a una máquina. Se dividen en tres subtipos: estáticos, dinámicos y trazos. Como estos gestos tienen un significado meramente simbólico, su trazado no necesariamente está relacionado con su significado.

Icónico (Iconic): Son usados para demostrar tamaño, forma, curvatura de objetos o entidades. En contraste con los gestos semafóricos, su trazado o trayectoria de movimiento está estrictamente relacionado a su significado. Estos se dividen en 2 subtipos: estáticos y dinámicos.

Pantomímico (Pantomimic): Usado para imitar la realización de una tarea o actividad sin herramientas u objetos. El gesto pantomímico se caracteriza por una alta variabilidad de postura y movimientos.

Manipulación (Manipulation): Usado en el control de la posición, rotación y escala de un objeto o entidad en el espacio. Gestos de manipulación constituyen una interacción directa entre el objeto manipulado y la mano o herramienta utilizadas para realizar el gesto.

2.7. Modelos de representación.

En este tema se presenta una vista general de los modelos espaciales para la representación de gestos en entornos gráficos. Dependiendo del tipo de datos de entrada (Input Data), el enfoque del reconocimiento gestual puede ser realizado de diferentes formas. Existen dos tipos principales de representación gestual definidos en la literatura: el modelado 2D limitado a la obtención y representación de un gesto de 2 dimensiones espaciales y el modelado 3D que posee 3 dimensiones útiles para casos donde se requiere unas capacidades de interactividad más ricas (ver Figura 2-10).

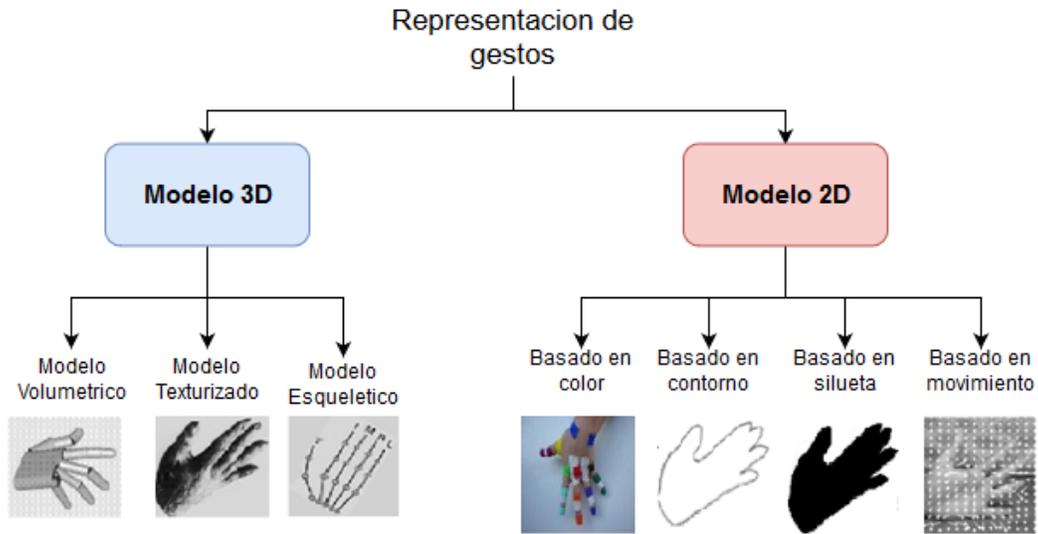


Figura 2-10. Representación de gestos de la mano a partir de sistemas de visión.

2.8. Controlador Leap Motion para interfaces naturales de usuario

La compañía LeapMotion fue fundada en el año 2010 por Michael Buckwald y David Holz. El primer dispositivo comercial fue lanzado por primera vez en julio del 2013 bajo el nombre “The Leap”. Aunque sus inicios y primeros prototipos se remontan al año 2008, cuando su cofundador David Holz estaba estudiando un doctorado en matemáticas.

2.8.1. Acceso a datos

En Leap Motion (LMC), los datos capturados son procesados por drivers propietarios por el vendedor y accesibles a partir de una API. Leap Motion tiene como propósito de ser una interfaz humano-máquina, no un escáner 3D multipropósito, por lo cual está optimizado para el reconocimiento de manos humanas y objetos punzantes.

El contenedor principal de datos que se obtiene a partir de la API de LMC es un *frame* (ver Figura 2-11). Un *frame* consiste en las manos, dedos, pointables (objetos visibles por el controlador) e información extra, tales como los gestos reconocidos por un sencillo mecanismo de reconocimiento integrado, marca de tiempo del marco, rotación, traslación y datos de escalado.

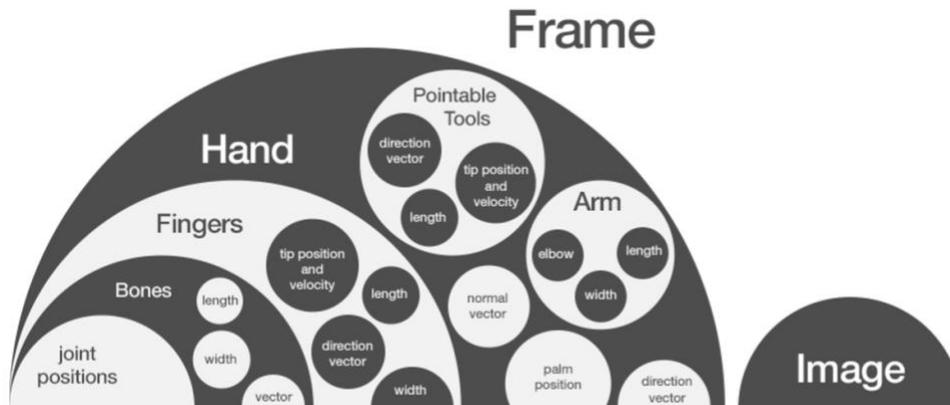


Figura 2-11. Estructura de datos del contenedor Frame (Davis, 2014).

2.8.2. Clase Hand

Hand es la entidad principal que es rastreada por el controlador Leap Motion (LMC) (ver Figura 2-12). El LMC mantiene un modelo interno de una mano humana y la asocia a la información obtenida del sensor. Esto tiene como propósito realizar el seguimiento de la posición de los dedos de la mano incluso cuando no es completamente visible.

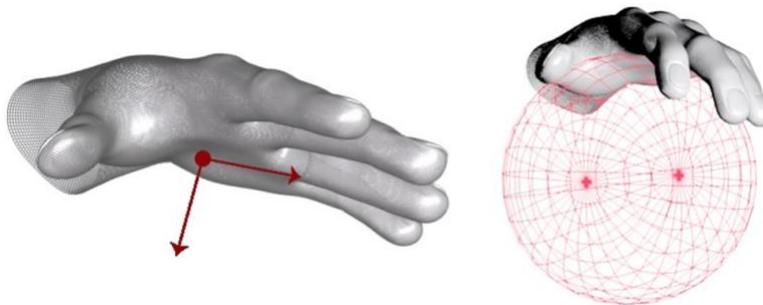


Figura 2-12. Representación visual de los datos palmNormal y grabStrength (Davis, 2014).

Este proceso de asociación de datos también permite al dispositivo determinar si se trata de la mano izquierda o derecha y que atributos poseen ambas:

- Posición del centro de la palma (*palm position*) en un vector en milímetros
- Velocidad de la palma (*palm velocity*) en milímetros/segundo.
- Vector perpendicular a la palma (*palm normal*) vector.
- Dirección de la mano (*direction*) vector.

- Información de la esfera que describe la postura de la mano (*grabStrength*, *pinchStrength*)
- Factores de movimiento como escala, rotación y translación (Motion Factors).

2.8.3. Clase Finger y Pointable.

Tanto la clase Finger como Pointable poseen una estructura similar. Estos elementos están asociados con Hand y a partir del mismo son consultados. Tanto de Finger como Pointable se obtiene un vector que contiene los datos de cada uno de los 5 dedos de las manos. Se puede acceder a un dedo concreto usando los valores mostrados en la Tabla 2-2.

Tabla 2-2. Valores de Finger y Pointable asociados a datos vectoriales de dedos

TYPE_THUMB == 0 (Dedo pulgar)
TYPE_INDEX = 1 (Dedo índice)
TYPE_MIDDLE = 2 (Dedo corazón)
TYPE_RING = 3 (Dedo anular)
TYPE_PINKY = 4 (Dedo pequeño)

Los atributos que se obtienen en un objeto del tipo Finger son los siguientes:

- Posición del extremo del dedo (*Tip position*) Vector en milímetros.
- Velocidad del extremo (*Tip velocity*) del dedo en milímetros/s.
- Posición estabilizada del extremo (*Stabilized tip position*) del dedo usando la velocidad y la posición en el *frame* anterior. Vector.
- Dirección (*Tip direction*) del dedo (ver Figura 2-13). Vector.
- Longitud (*Length*) del dedo.
- Anchura (*Width*) del dedo.

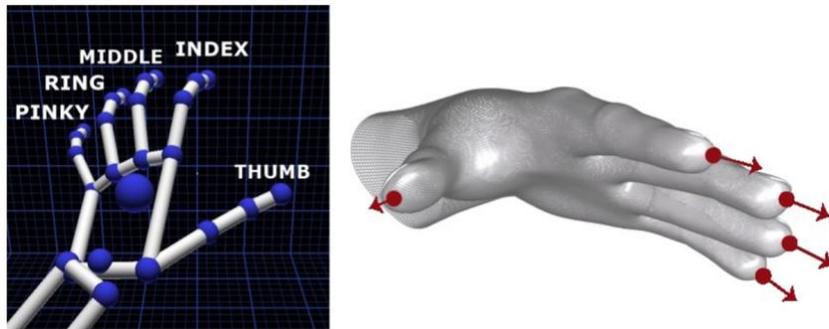


Figura 2-13. Representación visual de *tipDirection* de cada dedo (Davis, 2014).

2.8.4. Clase Bone.

De forma similar a como se obtiene la información de los dedos, se accede a la información a los huesos que la componen. Para acceder al objeto tipo Bone, se realiza a través de Frame/Hand/Finger. Y análogamente a los dedos, se obtiene un vector que contiene los 4 huesos de cada dedo. Los huesos de los dedos de una mano pueden accederse a partir de la numeración mostrada en la Tabla 2-3:

Tabla 2-3. Valores para acceso de datos vectoriales de Bone

TYPE_METACARPAL = = 0 (Hueso que conecta con la muñeca).
TYPE_PROXIMAL = = 1 (Hueso de la base del dedo)
TYPE_INTERMEDIATE = = 2 (Hueso ente el extremo y la base del dedo)
TYPE_DISTAL = = 3 Hueso en el extremo del dedo)

Los atributos que posee el objeto Bone son los siguientes:

- Dirección del hueso (*Direction*) vector.
- Longitud del hueso (*Length*).
- Anchura del hueso (*Width*).
- El punto medio del hueso (*Center*).

Para crear la aplicación que se quiere desarrollar en este trabajo, es fundamental conocer el SDK de Leap Motion. Sus clases, se acceden con lenguajes de programación entre los que están C++, C #, Unity, Objective-C, Java, JavaScript, Python y Unreal Engine.

El controlador aplica algoritmos a los datos brutos del sensor con el fin de extraer la posición precisa de las manos y los dedos. El software llamado Leap Motion Service, es el encargado del procesamiento de imágenes y de la reconstrucción de una representación en 3D en forma de datos vectoriales de acuerdo a lo que el dispositivo ve.

Luego, el software realiza el seguimiento y compara los datos, aplicando algoritmos de seguimiento e inferencia de los objetos reconocidos. Aquí es donde se aplican técnicas de filtrado para asegurar que los datos sean coherentes respecto al tiempo. Finalmente, el dispositivo emite datos resultantes vectoriales a través de un protocolo de transporte que contiene los datos de seguimiento y dependiente del lenguaje de programación o plataforma, una serie de imágenes.

Mediante este protocolo, el servicio se comunica con el panel de control de Leap Motion, así como con bibliotecas cliente nativas y clientes web, mediante una conexión de socket local (TCP para nativo, WebSocket para web). Esta librería organiza los datos en una estructura API orientada a objetos, gestiona el historial de *frames* y proporciona funciones y clases de ayuda.

El sensor de Leap Motion cubre aproximadamente un área de 60 cm por encima del dispositivo y otros 60 cm a cada lado con un ángulo de visión de hasta 150° grados. Sin embargo, la nueva versión beta del software denominada Orión, ha expandido esos límites hasta los 80 cm. Esta limitación ocurre a causa del uso de la luz LED y su propagación en el espacio, ya que se vuelve difícil de inferir la posición de las manos en un análisis estereoscópico o 3D hasta cierta distancia.

2.8.5. Sistema de coordenadas.

El controlador Leap Motion provee coordenadas en unidades reales en milímetros entre el frame de referencia de Leap Motion. El controlador Leap Motion y su centro representan también el centro de referencia para un frame. Su origen se muestra en la Figura 2-14.

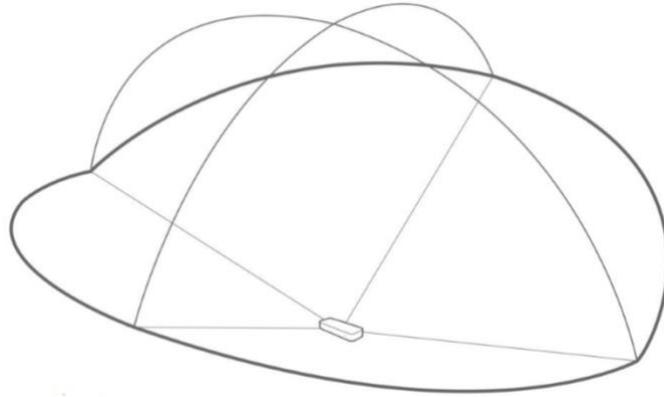


Figura 2-14. Visualización del campo de visión del sensor LMC.

A causa de que las coordenadas en milímetros que provee el dispositivo deben de ser interpretadas en forma que tengan sentido a las dimensiones dentro de un entorno de una aplicación, las diferencias existentes entre el sistema de coordenadas del hardware y del software podría requerir cambios de orientación de los ejes y del sistema de coordenadas o ignorar datos relacionados con el espacio.

2.9. Algoritmo \$P Point-Cloud para el Reconocimiento de gestos.

El reconocimiento de gestos es un tópico de las ciencias computacionales y un tipo de interfaz computacional que trata de la captura del lenguaje corporal humano y su interpretación por la computadora para ser usado como comandos. Se aplican algoritmos matemáticos para generar modelos que puedan ser interpretados por la computadora y así generar una forma de interactividad entre el hombre y la máquina sin el uso de ningún dispositivo mecánico. Generalmente, el algoritmo o técnica de reconocimiento de gestos a implementar (árboles de decisión, clasificadores estadísticos, redes neuronales, vecinos cercanos, etc.) depende de la representación que se desea y los atributos de implementación (rápida respuesta, reconocimiento más exacto, fácil implementación, bajo uso de recursos computacionales, etc.).

El *\$P Point-Cloud Recognizer* es una librería diseñada para aportar los algoritmos y técnicas necesarias para el desarrollo de prototipado rápido de aplicación e interfaces basadas en el reconocimiento de gestos mediante trazos 2D. Este reconocedor representa un gesto como una nube de puntos que, en conjunto, forman un trazo o trayectoria que representa el

movimiento realizado por las yemas de los dedos de una mano. El algoritmo “\$P” tiene como propósito ofrecer una solución rápida de implementación de reconocimiento de gestos a proyectos que requieran interacciones humana-máquina naturales (Vatavu, Anthony, & Wobbrock, 2012).

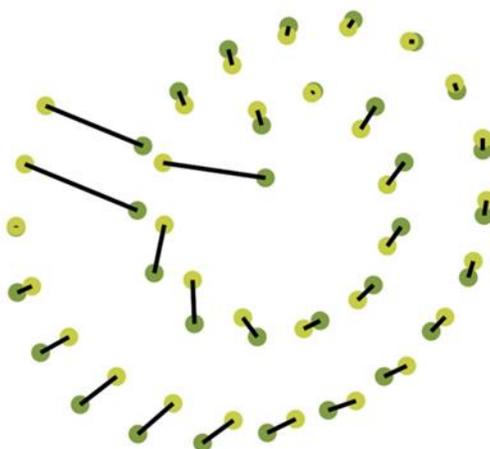


Figura 2-15. Representación gráfica del proceso de comparación entre 2 nubes de puntos (Vatavu et al., 2012).

Representando un gesto como una nube de puntos (ver Figura 2-15), es posible manejar gestos formados por una trayectoria o varias trayectorias por igual, sin embargo, se ignoran el orden y la dirección. Para comparar dos nubes de puntos y obtener un resultado, \$P encuentra una solución al clásico problema de la asignación entre 2 gráficas bipartitas, utilizando una aproximación del algoritmo húngaro con el uso de heurísticas para “clouds matching”.

En conjunto, dicha nube de puntos forma un trazo o trayectoria geométrica que después es rotada, escalada y normalizada, para que sea comparada con una base de datos que almacene gestos (ver Figura 2-16).

En términos de aprendizaje máquina, \$P es un clasificador con función de distancias euclidianas 2D, es decir, un igualador de plantillas geométricas.

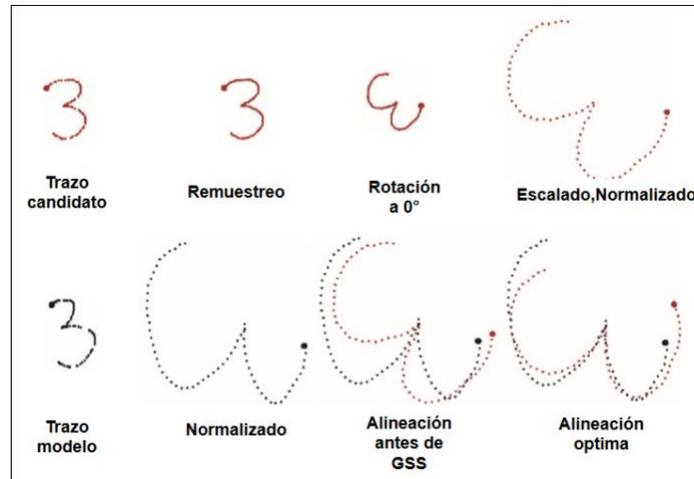


Figura 2-16. Pasos del proceso de reconocimiento en SP recognizer. GSS significa Golden Section Search (Vatavu et al., 2012).

2.10. Interfaz Gráfica de Usuario.

La Interfaz Gráfica de Usuario (GUI por sus siglas en inglés) es un programa que sirve de intermediario entre el usuario y la máquina. Este software muestra de forma visual todas las acciones posibles de interacción en una plataforma, utilizando un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz. Su principal uso, consiste en proporcionar un entorno visual sencillo e intuitivo para permitir la comunicación con el sistema a usuarios que no poseen conocimientos de informática.

Actualmente existe software orientado al desarrollo de GUI que buscan lograr mejores formas de representar simbólicamente y visualmente las acciones e interacciones posibles dentro de un software, ya sea dentro de entornos gráficos orientados a dimensiones 2D, como el concepto escritorio 2D se presenta en el sistema operativo de una computadora, o software orientado al desarrollo de entornos gráficos interactivos 3D con el uso de tecnologías de realidad virtual o aumentada y/o para el renderizado de gráficos 3D en videojuegos.

2.10.1. Blockly

Blockly es una librería JavaScript para la creación de interfaces gráficas que implementan un lenguaje de programación visual mediante bloques (ver Figura 2-17), donde un conjunto de bloques realiza la representación de forma abstracta con los pasos necesarios para ejecutar

un ejercicio de programación. Blockly es un proyecto de Google y es código abierto bajo la licencia Apache 2.0 (Fraser & others, 2013).

Blockly permite que los programadores principiantes puedan centrarse en la lógica, ofrece una serie de comandos y reglas de relaciones entre ellos. Provee bloques que funcionan como componentes de control, de lógica, de operaciones matemáticas y de procesos en una interfaz gráfica intuitiva y amigable con el usuario.

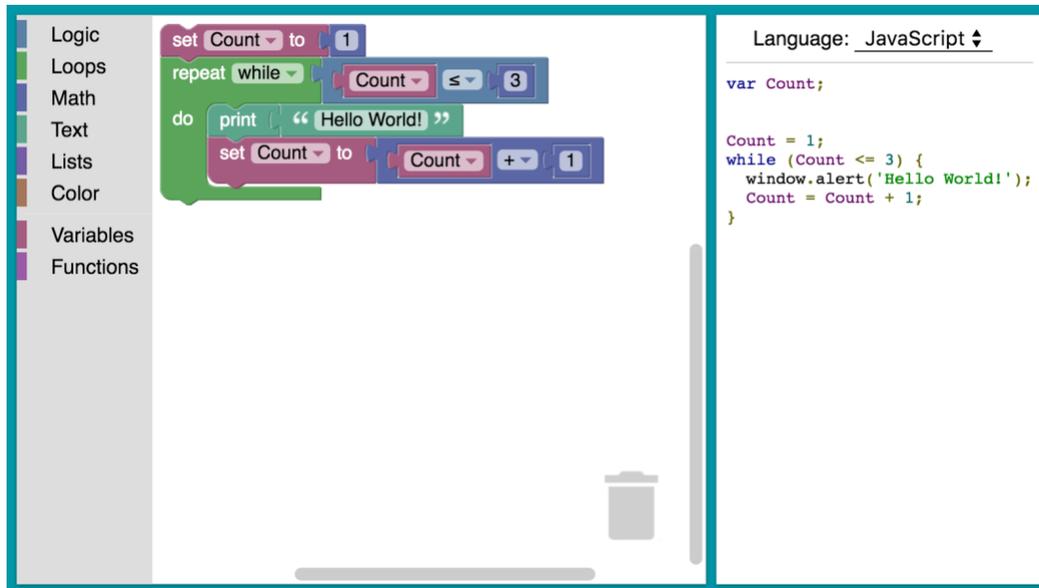


Figura 2-17. Ejemplo interfaz hecha en Blockly.

Blockly por defecto presenta una interfaz de usuario compuesta por una caja de herramientas (*toolbox*), la cual contiene los bloques disponibles y un espacio de trabajo (*workspace*), donde el usuario puede interactuar con los bloques dentro del área con solo arrastrar y soltar (*drag & drop*), así como realizar relaciones entre elementos que se pueden unir o ser insertados uno sobre el otro, pero también puede ser extensible y configurable, ya que permite la creación de nuevos bloques y la modificación del aspecto gráfico de la interfaz.

2.10.2. Babylon js

En un motor gráfico para el renderizado de gráficos 3D en navegadores web vía HTML5 utilizando las librerías JavaScript WebGL. El código es abierto y distribuido bajo la licencia Apache 2.0 (Catuhe, Rousseau, Lagarde, & Rousset, 2014).

Permite la creación de escenarios 3D dentro de una página web y este es renderizado dentro de un elemento HTML 5 *canvas*. En la Figura 2-18 se puede observar un editor de objetos 3D hecho en Babylon JS ejecutándose en un navegador web.

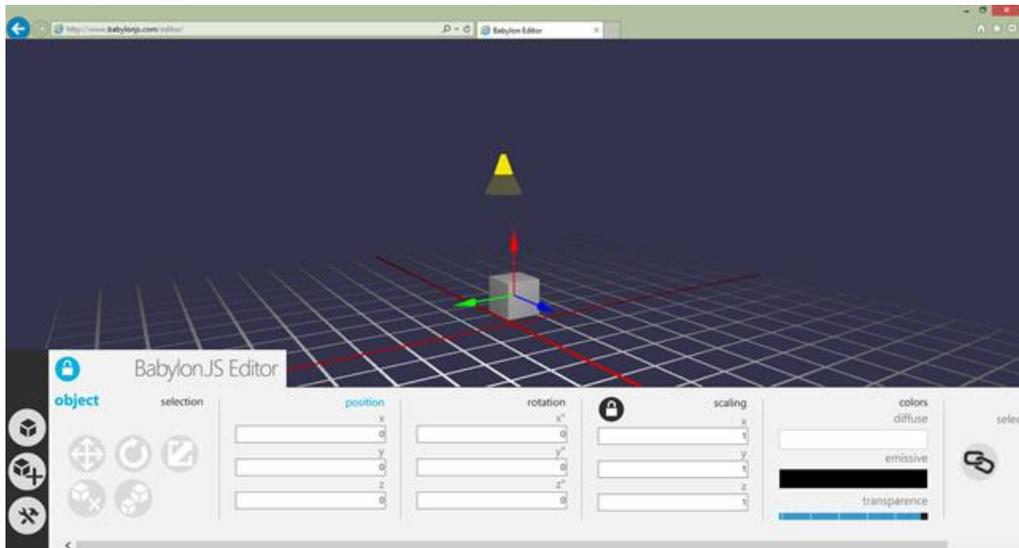


Figura 2-18. Editor de objetos 3D hecho en Babylon JS

A partir de un programa Shader que determina la posición de píxeles de los objetos dentro del escenario, características como las texturas, colores, así como sombras son aplicados a cada modelo 3D. En cada escenario se representa un objeto cámara y un objeto que controla la iluminación, así como un sistema global de matrices 4x4 que controla la posición, la rotación y la escala de los modelos 3D. Posee técnicas de post-procesado, soporte de sistemas de simulación de físicas y de partículas, así como control de animación de objetos mediante key-frames o sistemas esqueléticos.

Capítulo 3

3.Estado del arte

En esta sección se describen investigaciones y trabajos en áreas relacionadas con el campo del pensamiento computacional en la educación, se presentan algunos proyectos y aplicaciones interactivas orientadas al aprendizaje con enfoque al dispositivo Leap Motion.

3.1. Ambientes de aprendizaje orientados al desarrollo del pensamiento computacional.

Actualmente, existen diversas herramientas o sistemas e-learning de apoyo para la programación, donde se facilitan la creación de programas y algoritmos a los estudiantes. Estas denominadas herramientas de autor permiten a los usuarios diseñar y construir animaciones, juegos, simulaciones, etc. La mayoría pretende que el estudiante desarrolle lo que se le conoce como “pensamiento computacional” (Wing, 2006).

Una de las herramientas orientadas al desarrollo del pensamiento computacional más importantes que se puede mencionar es Scratch (Resnick et al., 2009), una comunidad de aprendizaje creativo en línea, donde los usuarios pueden crear sus proyectos a partir de bloques gráficos, los cuales representan elementos de lenguaje de la programación, como son: control, operadores, variables, funciones, etc. (Ver Figura 3-1).

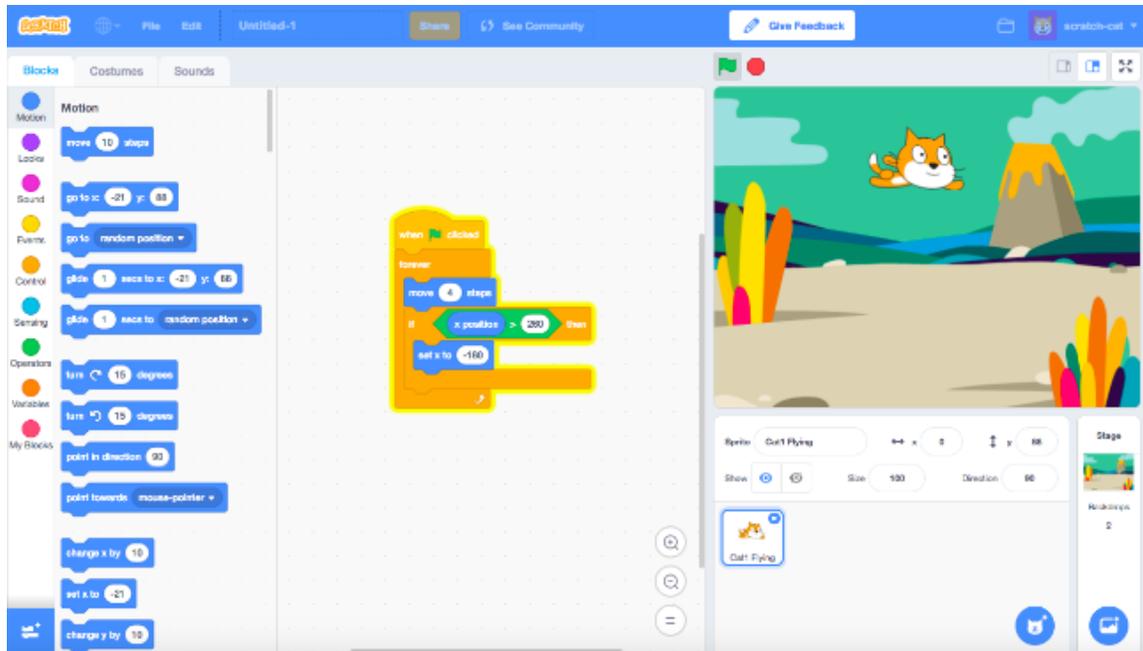


Figura 3-1. Interfaz de programación en bloques en Scratch

Otra herramienta importante es Alice (Werner, Campe, & Denner, 2012), en donde principiantes pueden programar animaciones y juegos 3D utilizando modelos ya predefinidos a partir de una interfaz visual basada en la programación por bloques (Ver Figura 3-2).



Figura 3-2. Interfaz de programación en bloques en Alice 3

Snap! Llamado antes Build Your Own Blocks (Harvey, Garcia, Paley, & Segars, 2012) es una versión de Scratch extendida que tiene como característica principal la creación de bloques personalizados para que sean implementados en creaciones con la misma herramienta (Ver Figura 3-3).

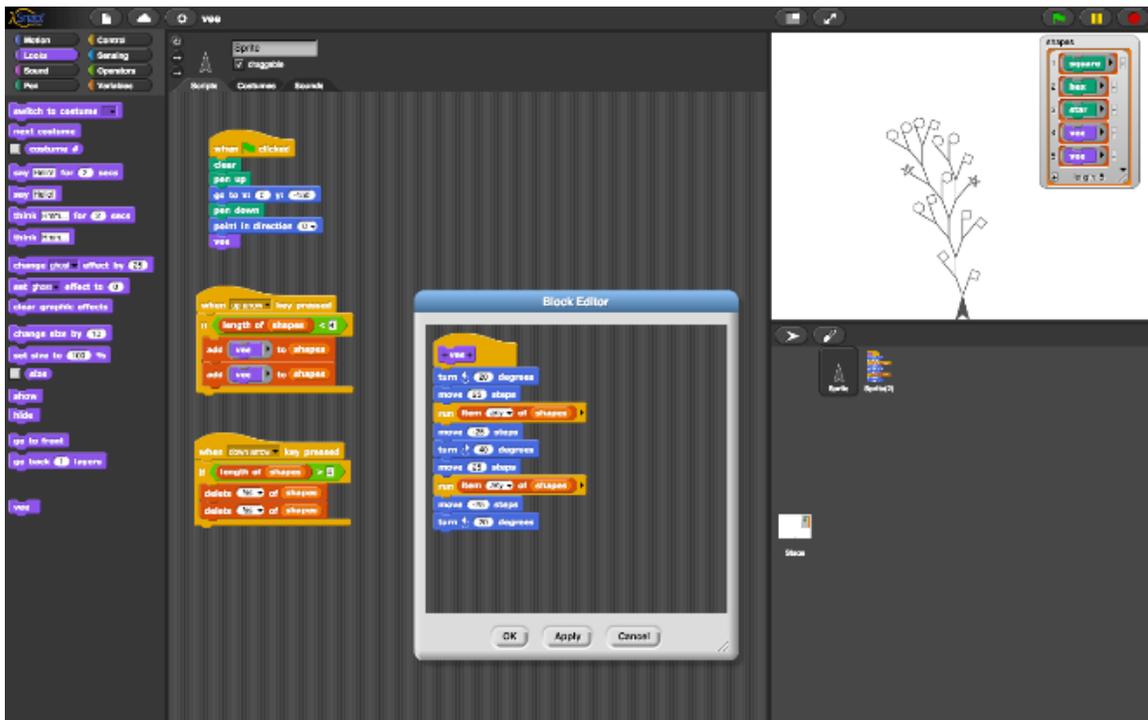


Figura 3-3. Interfaz de desarrollo de bloques en Snap!

MIT Android App Inventor (Pokress & Veiga, 2013) es una aplicación orientada al desarrollo de aplicaciones Android creadas a partir de una simple interfaz de programación basada en bloques (ver figura 3-4) para convertir dicho lenguaje a lenguaje basado en texto en tiempo real. Este puede ser útil en el aprendizaje de lenguajes tradicionales como JavaScript o Python.

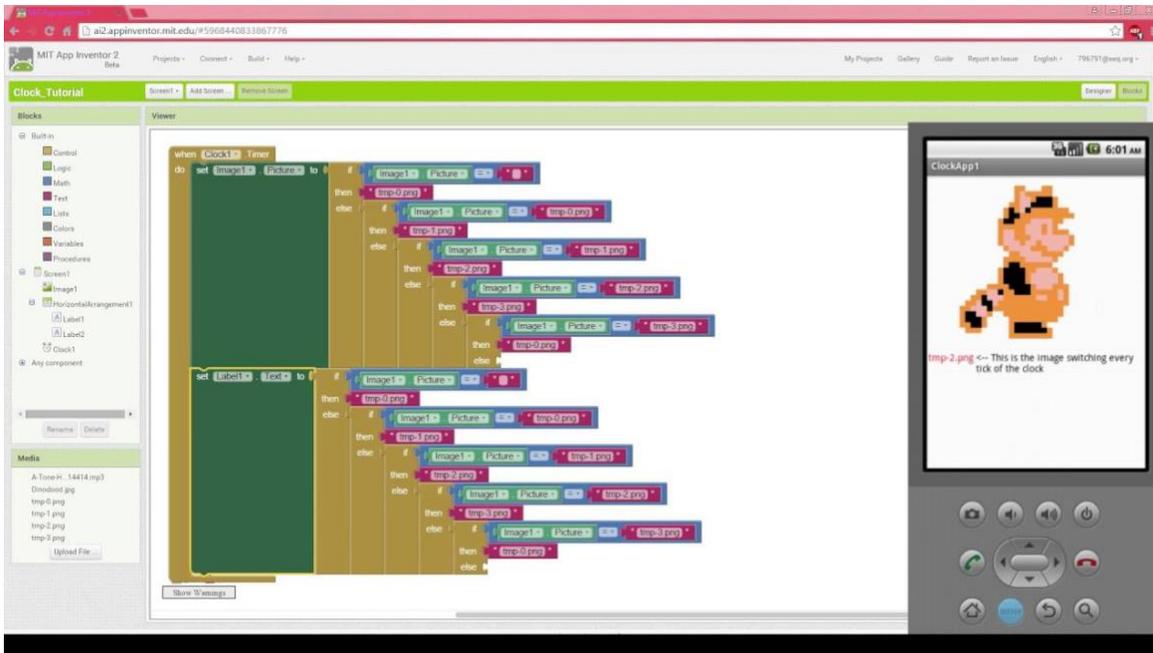


Figura 3-4. Interfaz de MIT App Inventor

GameFroot (Gamefroot , 2017) es una plataforma de creación de juegos online para crear juegos side-scrolling. Utiliza un editor con bloques y posee capacidades de arrastrar y soltar (*drag and drop*) y scripting avanzado para la ejecución de eventos en las aplicaciones. Pocket Code (Slany, 2014) es un lenguaje de programación visual para crear aplicaciones en dispositivos móviles. Posee soporte para Android, iOS y dispositivos que poseen soporte con HTML5. Inspirado en Scratch y desarrollado por Catrobat team como software de código abierto (*open source*) (Ver Figura 3-5).

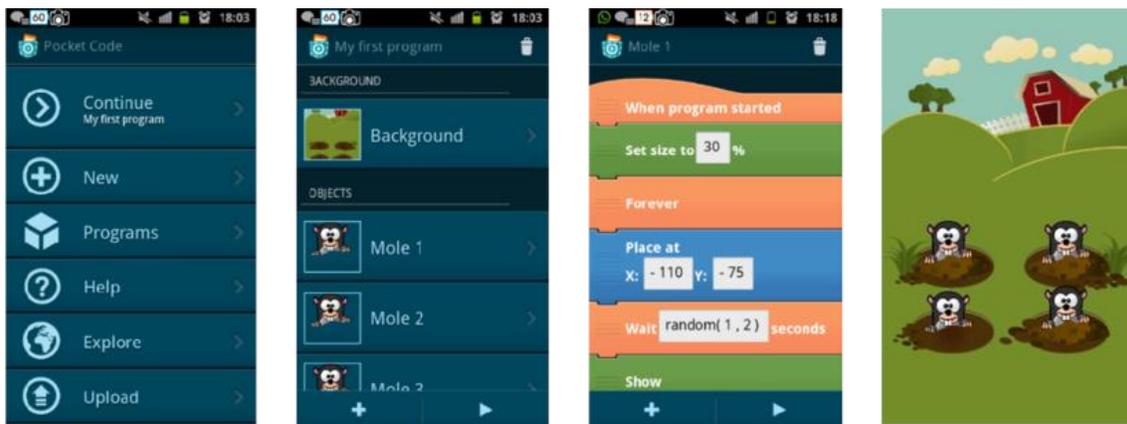


Figura 3-5. Ejemplo de aplicación en Pocket Code

Tynker (Fuel, 2018) es un lenguaje de programación visual con bloques orientado a niños, el cual permite que cualquiera pueda crear juegos fácilmente (figura 3-6). Posee funcionalidad a diversos dispositivos e incluso un modo para funcionar en conjunto con el videojuego Minecraft.

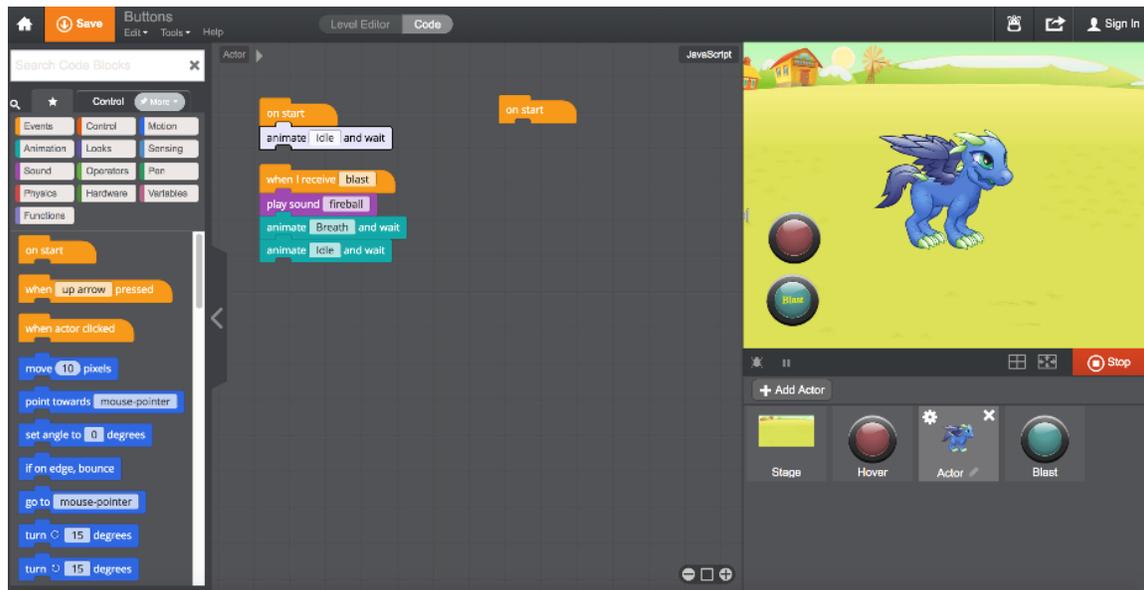


Figura 3-6. GUI predeterminada de Tynker

GameSalad (Guerineau & Abrams, 2012) es una herramienta de autor para programación de videojuegos que posee una interfaz de programación con capacidades de arrastrar y soltar (*drag and drop*). Enfocado a los más inexpertos permite que cualquiera pueda crear juegos fácilmente. GameMakerStudio (Yoyo Games, 2015) es un software para la creación de videojuegos de diversos géneros para programadores inexpertos (figura 3-7). Utiliza un lenguaje de texto CUI llamado “Game Maker Language” pero también posee características de arrastrar y soltar (*drag and drop*), en la mayoría de sus elementos de programación.

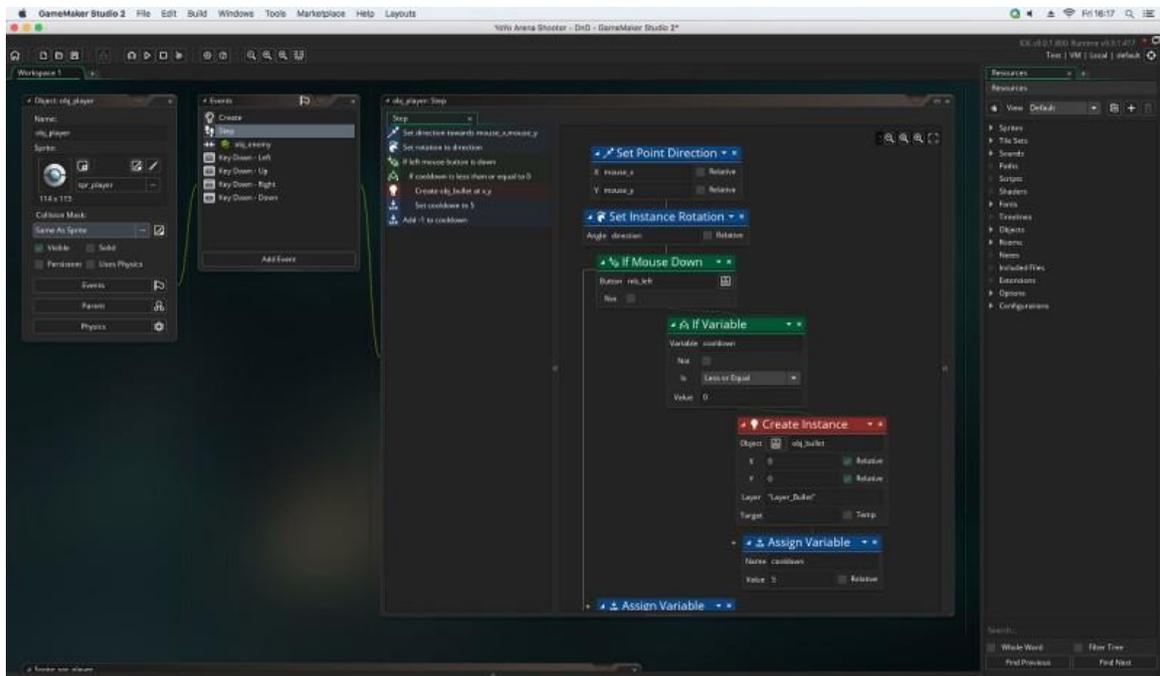


Figura 3-7. GUI en GameMakerStudio 2

BeetleBlocks (Koschitz, Ramagosa, & Rosenbaum, 2016) es un ambiente de programación visual mediante bloques para el diseño y creación de estructuras 3D (figura 3-8). Soporta tanto interfaces de desarrollo mediante bloques y texto escrito para scripts Desarrollado en JavaScript como una aplicación web y funciona para cualquier navegador con soporte HTML5.

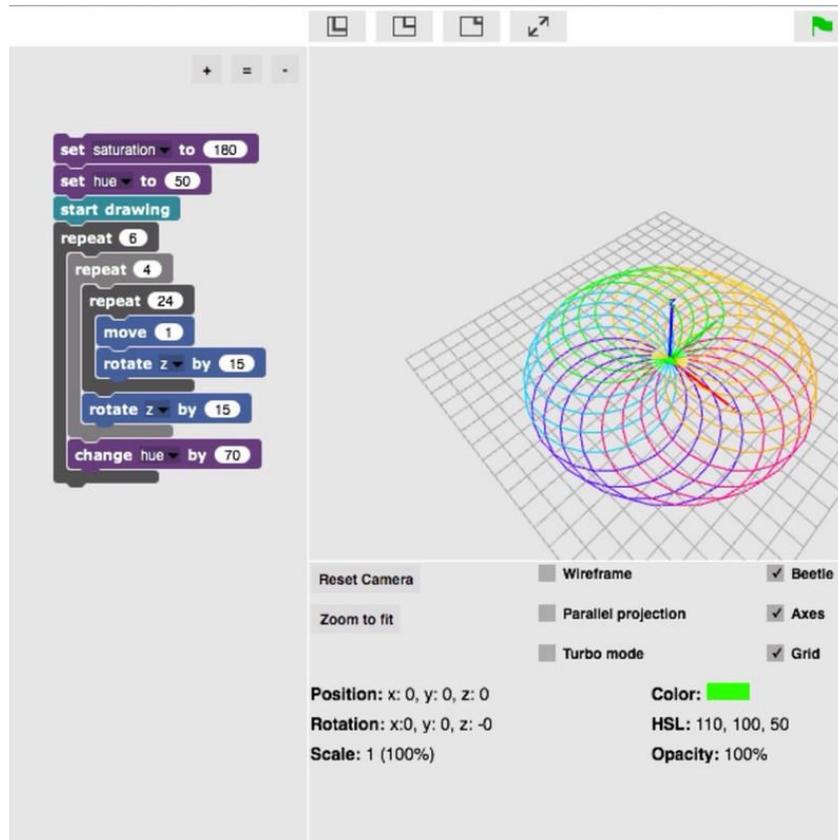


Figura 3-8. GUI en BeetleBlocks

AgentCubes (Ioannidou, Repenning, & Webb, 2009) en un lenguaje de programación educativo para niños para la creación de juegos y simuladores en entornos 2D y 3D. Utiliza un interfaz de programación visual con bloques (figura 3-9) y el diseño de las aplicaciones está basada en Scalable Game Design.



Figura 3-9. GUI de AgentCubes

ToonTalk Reborn (Kahn, 2014) es un software de programación enfocado a los niños para la creación de animaciones utilizando un lenguaje visual por medio de bloques. Existe una versión para Windows, así como una desarrollada en JavaScript para su ejecución en navegadores con soporte HTML5. TurtleArt (Bontá, Papert, & Silverman, 2010) Es una aplicación para el aprendizaje por medio de programación por bloques parecida a Blockly (figura 3-10) para la creación de imágenes artísticas.

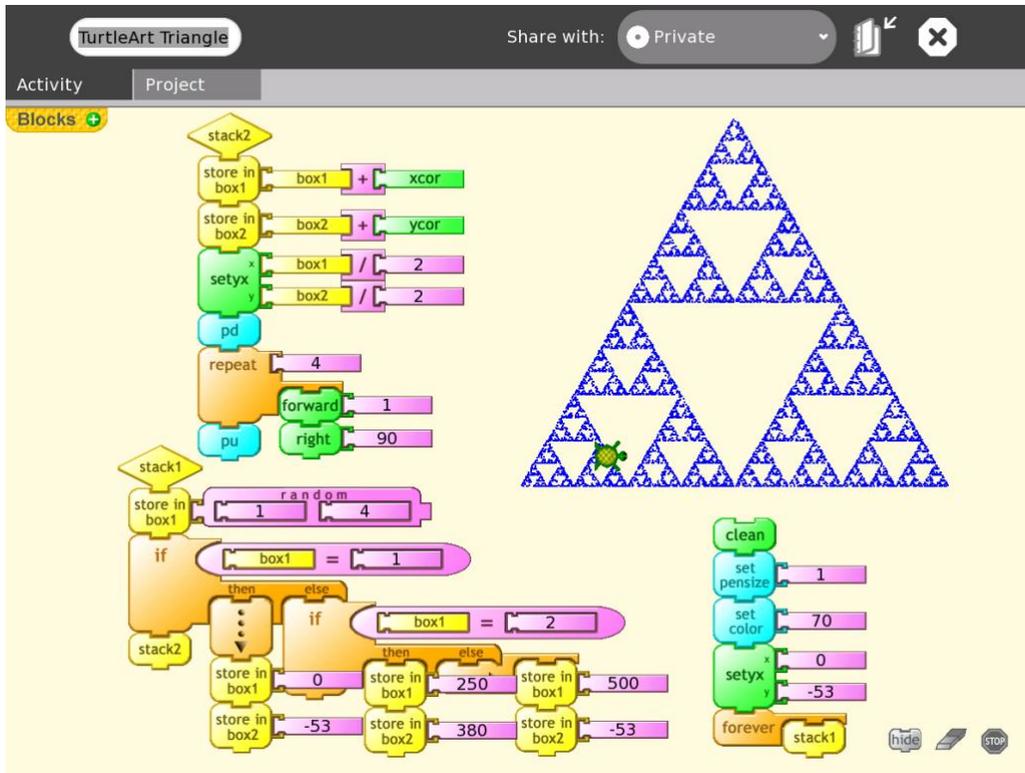


Figura 3-10. GUI de TurtleArt

En la Tabla 3-1 se presenta una comparación de las herramientas de autor por tipo de entorno, plataforma de desarrollo y dominio. Como se puede observar, la mayoría de los proyectos están enfocados al desarrollo de aplicaciones o juegos, a excepción de algunos que tienen otros enfoque como el desarrollo artístico de estructuras graficas 3D, 2D y animaciones.

Tabla 3-1. Herramientas de autor para el desarrollo del pensamiento computacional

Nombre de proyecto	Autores y año	Tipo de entorno	Plataformas	Dominio
Scratch	Resnick et al. (2009)	2D	Web	Programación por bloques
AgentCubes	Ioannidou, Repenning, & Webb (2009)	2D y 3D	Desktop	Creación de juegos y simuladores de entornos
TurtleArt	Bontá, Papert, & Silverman (2010)	2D	Desktop	Creación de imágenes artísticas
Alice	(Werner, Campe, & Denner (2012)	2D y 3D	Desktop	Creación de juegos y animaciones
Snap!	Harvey, Garcia, Paley, & Segars (2012)	2D	Desktop Móviles Android iOS	Creación de bloques personalizados y programación.
GameSalad	Guerineau & Abrams, 2012	2D	Desktop Android iOS	Plataforma de creación de juegos.
MIT Android App Inventor	Pokress & Veiga (2013)	2D	Android	Creación de aplicaciones
Pocket Code	Slany (2014)	2D	Android iOS Web	Creación de aplicaciones
ToonTalk Reborn	Kahn (2014)	2D	Web	Creación de animaciones
GameMakerStudio	Yoyo Games (2015)	2D	Desktop	Creación de juegos multiplataforma
BeetleBlocks	Koschitz, Ramagosa, & Rosenbaum (2016)	3D	Desktop	Creación y diseño de estructuras 3D.
GameFroot	Gamefroot (2017)	2D	Web	Creación de juegos side-scrolling online
Tynker	Fuel (2018)	2D y 3D	Desktop Android iOS	Creación de juegos multiplataforma y MOD de Minecraft

3.2. Aplicaciones interactivas orientadas al aprendizaje.

En el marco del uso de interfaces naturales, mediante el reconocimiento de gestos en el campo de la educación, los trabajos de investigación más importantes han estado dedicados principalmente a estudiar factores como la usabilidad y la motivación en prototipos educativos. S. Bowe, M. Antoniou, C. Garrett et al. (Bowe et al., 2015) presentan el aprendizaje de niños y jóvenes utilizando dispositivos con pantalla táctil, donde se realiza una actividad con un juego de la Torre de Hanoi, dentro de una aplicación Android, para luego resolver uno real, concluyendo que el conocimiento y habilidades aprendidas en una

tarea desde un dispositivo móvil pueden ser transferidas en el mundo real. De acuerdo con el trabajo de M. Ebner et al. (Ebner & Spot, 2015) el uso de interfaces innovadoras como el controlador Leap Motion (LMC), mejora significativamente el proceso de aprendizaje dentro de aplicaciones que educan mediante el juego. Estas mejoran la motivación y la diversión de los estudiantes mientras juegan aprendiendo. G. Zhu, S. Cai, y. Ma et al. (Zhu, Cai, Ma, & Liu, 2015) usan LMC para analizar la efectividad para la rehabilitación motriz de niños con autismo a partir de una serie de ejercicios en aplicaciones para el sector escolar.

3.2.1. Sobre realidad virtual.

Para el campo de realidad virtual (VR por sus siglas en inglés) y realidad aumentada (AR por sus siglas en inglés) también existen trabajos que buscan el estímulo educativo por medio de la interactividad de este tipo de interfaces. Una interfaz de usuario intuitiva es un factor importante en las plataformas virtuales educativas. Casos como el LMC y el Kinect muestran bases sólidas respecto al diseño de interfaces para este tipo de aplicaciones.

Moriarty y colegas presentan diversas pruebas para interactividad en ambientes educativos utilizando el sensor Kinect (Moriarty et al., 2012) donde se aplican ajustes a la experiencia de usuario utilizando inteligencia artificial. También, un sistema de realidad virtual, presentando el parque zoológico Elmwood (Chifor & Stefanut, 2015), provee una nueva forma de observar la vida salvaje animal.

Por otra parte, para la educación en medicina se presenta una plataforma de aprendizaje en VR (Nainggolan, Siregar, & Fahmi, 2016) donde se muestra un modelo anatómico del cuerpo humano en 3D que puede ser explorado interactivamente usando el LMC. En este trabajo, se mide la satisfacción del usuario a partir de un cuestionario. Los resultados muestran que el sistema puede apoyar eficazmente el proceso de aprendizaje sobre temas de anatomía en 3D. Otra plataforma, la zSpace es una evidencia del potencial de la combinación de interfaces gráficas, VR y sensores modernos (Noor & Aras, 2015), donde se utiliza el LMC y el *headset* EEG Emotiv para representar procesos complejos de interacción humana. El usuario puede explorar modelos de sistemas espaciales, por ejemplo, misiones de la NASA o la ESA. Salvadori y colegas presentan un enfoque similar en (Salvadori et al., 2014).

Diversas representaciones gráficas de sistemas moleculares se visualizan en el sistema de realidad virtual CAVE el cual permite que múltiples usuarios interactúen con ciencias

moleculares y analicen las estructuras de moléculas (Morse, Reading, Lueg, & Kenderdine, 2015).

V. Tran, J. Lee, D. Kim et al (Tran, Lee, Kim, & Jeong, 2016) en colaboración con The LEGO Group, LEGO System A/S Denmark usaron los dispositivos LMC y Kinect para presentar en VR una actividad interactiva que permitía al usuario diseñar y construir edificaciones con piezas LEGO. También en (Clarke, Dass, & Chau, 2016) se investigan enfoques de exploración de datos en gráficas que evolucionan con el tiempo. Por otra parte, se realizaron pruebas del rendimiento al realizar actividades en un tablero scrum digital usando el LMC en el trabajo de (Rittitum, Vatanawood, & Thongtak, 2016).

En la Tablas 3-2 se muestra una comparación entre las investigaciones de interactividad y educación mediante el uso de interfaces utilizando al LMC.

Tabla 3-2. Proyectos interactivos enfocados al aprendizaje con LMC

Investigación	Autor y año	Dispositivos	Propósito
Utilizing depth based sensors and customizable software frameworks for experiential application.	Moriarty et al. (2012)	Kinect	Mejora de experiencia de usuario con IA en ambientes educativos.
Graphical interfaces and virtual reality for molecular sciences.	Salvadori et al. (2014)	LMC	Interactividad en VR para aprendizaje.
Young children's transfer of learning from a touchscreen device.	Bowe et al. (2015)	Pantalla touchscreen	Evaluación de aprendizaje en niños.
Game-Based Learning with the Leap Motion Controller.	Ebner, M., & Spot, N. (2015)	Leap Motion Controller	Gamificación e interactividad.
A Series of leap motion-based matching games for enhancing the fine motor skills of children with autism.	Zhu, G., Cai, S., Ma, Y., & Liu, E. (2015)	Leap Motion Controller	Mejora motriz en niños con autismo.
Potential of multimodal and multiuser interaction with virtual holography.	Noor, A. K., & Aras, R. (2015).	LMC	Interactividad VR para exploración virtual.
Immersive Virtual Reality application using Google Cardboard and Leap Motion technologies.	Chifor, M., & Stefanut, T. (2015)	Google cardboard y LMC	Interactividad en VR para aprendizaje.

TaggerVR: interactive data analytics for geoscience-a novel interface for interactive visual analytics of large geoscientific datasets in cloud repositories.	Morse et al (2015)	CAVE y LMC	Interactividad en VR para aprendizaje.
Anatomy learning system on human skeleton using Leap Motion Controller.	Nainggolan, F. L., Siregar, B., & Fahmi, F. (2016).	LMC	Interactividad educativa en AR.
Easy-to-use virtual brick manipulation techniques using hand gestures.	Tran, Lee, Kim, & Jeong (2016)	LMC y Kinect	Gamificación e interactividad en VR para aprendizaje.
Naturalmotion: Exploring gesture controls for visualizing time-evolving graphs.	Clarke, S., Dass, N., & Chau, D. H. P. (2016)	LMC	Interactividad educativa y reconocimiento de gestos.
Digital scrum board using leap motion.	Rittitum, P., Vatanawood, W., & Thongtak, A. (2016)	LMC	Interactividad educativa y reconocimiento de gestos.

3.2.2. En música.

Otro campo de aplicación de dispositivos libres de contacto (sensores de visión) es la transferencia de gestos a música. Esto puede ser considerado como una forma especial de educación. En este caso, el sensor LMC es puesto a prueba como una interfaz de gestos para la creación de nuevos instrumentos musicales digitales (DMI por sus siglas en inglés) simulando un teclado musical virtual (Silva, de Abreu, de Almeida, Teichrieb, & Ramalho, 2013).

Los principales problemas tratados son la pérdida del rastreo cuando los dedos de la mano se encuentran muy cercanos entre sí, problemas de oclusión que obligan al usuario a tomar posturas incómodas e inusuales, así como la variación de latencia y la retroalimentación entre acciones.

(Moore, Howell, Stiles, Herrera, & McMahan, 2015) construyeron un instrumento musical virtual (VMI por sus siglas en inglés) que consiste de un Oculus Rift y un LMC. La interfaz diseñada permitió con un gesto componer música mediante el arreglo de notas en un ambiente virtual y con un gesto secundario en control de reproducción de composiciones. Otro trabajo de investigación donde se discutió la efectividad de enseñanza y rendimiento de aprendizaje

en una clase de música utilizando el LMC en una escuela primaria es (Perdana, 2014). Los resultados muestran que el reconocimiento de movimiento es ineficiente para el uso diario. Las tareas realizadas dentro de la investigación fueron calificadas como muy demandantes y la pérdida de rastreo fue considerada como el problema principal. (Hemery, Manitsaris, Moutarde, Volioti, & Manitsaris, 2015) propusieron la ejecución y aprendizaje de gestos musicales usando Kinect y LMC. Los autores presentan un concepto y un prototipo de una NUI que soporta el reconocimiento de gestos al utilizar un piano, para transformar dichos gestos en sonidos. En (Volioti et al., 2015) se diseñó una plataforma para el aprendizaje de habilidades motrices de gestos musicales. Un sistema basado en LMC para interactuar con partículas que representan arreglos orquestales fue presentado por (Fonteles, Sousa, & Rodrigues, 2015). El sistema funcionaba para la visualización de composiciones musicales para la audiencia y el conductor.

A continuación, se muestra la Tabla 3-3 que contiene una lista de los proyectos interactivos educativos musicales que se apoyaban en el uso del LMC.

Tabla 3-3. Proyectos interactivos educativos musicales

Investigación	Autor y año	Dispositivos	Propósito
Teaching elementary school students new method of music performance with Leap Motion.	Perdana, I. (2014)	LMC	Aprendizaje de música utilizando interfaces interactivas.
Wedge: A musical interface for building and playing composition-appropriate immersive environments.	Moore, A. G et al (2015)	LMC y Oculus Rift	Interface para instrumento musical virtual (VMI).
Towards the design of a natural user interface for performing and learning musical gestures.	Hemery et al (2015)	LMC y Kinect	Aprendizaje de gestos musicales utilizando reconocimiento de gestos.
Music gestural skills development engaging teachers, learners and expert performers.	Volioti, C. et al (2015)	LMC	Desarrollo de habilidades motrices de gestos musicales.
Visual and Interactive Performance of Particles Conducted by the Leap Motion for an Orchestral Arrangement.	Fonteles, J. H., Sousa, É. S., & Rodrigues, M. A. F. (2015)	LMC	Interactividad y visualización de composiciones musicales.

De acuerdo a las investigaciones realizadas, no existe un sistema enfocado al estudio de la interactividad del usuario con un dispositivo de cómputo, con la premisa de que el usuario aprenda y pueda dominar conceptos que componen el pensamiento computacional y espacial, a partir de la programación e implementación de interfaces naturales de usuario dentro de entornos de aprendizaje.

Por ello, la contribución principal de este proyecto consiste en presentar un prototipo de entorno de aprendizaje que englobe diferentes interfaces de interactividad natural, así como el uso de interfaces gráficas de programación mediante bloques y de entornos gráficos interactivos tanto para 2D como 3D. Esto con el fin de ofrecer al usuario final, capacidades de programación de reglas y modos de interacción entre un dispositivo de cómputo, donde la interactividad entre el hombre y la máquina pueda ser personalizada. Se trata de convertir el proceso de aprendizaje en una tarea intuitiva, natural e interactiva.

Capítulo 4

4. Desarrollo del proyecto

4.1. Diseño centrado en el humano.

El Diseño Centrado en el Humano (DCH) es un enfoque para el desarrollo de sistemas interactivos que tienen como propósito el crear sistemas útiles y usables para las necesidades y requisitos del usuario, aplicando los factores humanos, la ergonomía, usabilidad etc. La finalidad es mejorar la efectividad y eficiencia de uso, la satisfacción, la accesibilidad, el rendimiento, la seguridad y sustentabilidad del usuario (Sharp, Preece, & Rogers, 2019).

El DCH tiene como objetivo el diseñar productos interactivos que sean fáciles de usar, efectivos en su uso y con una experiencia de uso que se disfrute, así como la optimización de la interacción del usuario con un sistema y su entorno. El DCH está enfocado en entender el espacio del problema para proponer tecnologías innovadoras. En contraste con la ingeniería de software, donde los requerimientos de un programa de software se obtienen con un carácter de contrato entre el desarrollador y el cliente, el DCH enfatiza involucrar a los usuarios potenciales en el proceso de diseño para ayudarlos a establecer requerimientos que de otra manera serían muy difíciles de identificar (ver figura 4-1).

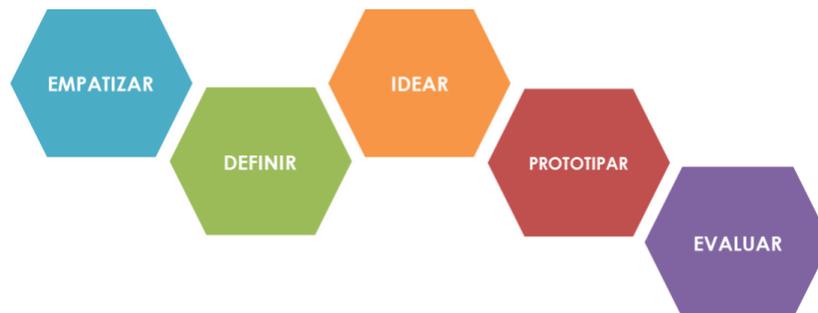


Figura 4-1. Fases del proceso de diseño del DCH

La literatura en DCH ha propuesto abstracciones acerca de los aspectos o principios de diseño como guías de lo que se debe y no se debe hacer al diseñar un sistema. Estos principios se

derivan de una mezcla de diferentes teorías basadas en conocimiento, experiencia de uso y sentido común.

Principios básicos de diseño de interacción:

1. **Visibilidad:** Hacer visibles las interacciones de los humanos con la computadora en la medida de lo posible. Mientras más visibles sean las funciones, más probable será que los humanos realicen la acción apropiada.
2. **Retroalimentación:** Proporcionar al usuario información inmediata acerca de la acción que se está ejecutando o que se acaba de ejecutar. En DCH existen diferentes tipos de retroalimentación que involucran el uso de sonidos, animaciones, vibraciones y combinaciones de dichos estímulos sensoriales.
3. **Restricciones:** Es limitar las opciones que los usuarios pueden elegir al ejecutar una acción con la finalidad de que el usuario no cometa errores, ofreciendo solo opciones relevantes.
4. **Consistencia:** Utilizar operaciones o elementos similares para tareas similares, ya que esto permite que los sistemas sean más fáciles de utilizar porque el usuario aprende un camino aplicable a varios objetos.
5. **Asequibilidad:** Indicar o dar pistas acerca de las acciones que se deben de realizar sobre un objeto. Este principio de diseño se ha utilizado mayormente en el diseño de objetos físicos ya que el mapeo es natural e involucra el uso de metáforas basadas en interacciones reales.

Si bien estos principios son generales, se debe de tomar en cuenta en que el proceso de DCH es altamente empírico, lo cual significa que los diseñadores deben de tomar decisiones basadas en el conocimiento que se tiene de los usuarios y en el contexto en que se utilizará el producto. Se debe de escuchar que quieren los usuarios, tomar en cuenta sus habilidades y considerar lo que puede ayudar a mejorar la forma en que realizan sus tareas.

De manera general, el proceso de diseño se desenvuelve en las siguientes actividades (Norman, 2013):

- **Identificar las necesidades y establecer los requerimientos para la experiencia de usuario.** Estudios empíricos (entrevistas, encuestas) para representar modelos conceptuales que describen las necesidades, estrategias y metas de los usuarios.
- **Desarrollar diseños alternativos que satisfagan los requerimientos.** Reflexiones sobre las ventajas y desventajas de alternativas de diseño en la búsqueda de lo que mejor satisfaga las necesidades del usuario.
- **Construir versiones interactivas de los diseños para ser comunicados y evaluados.** Creación de escenarios de uso que muestren como el producto será utilizado en la práctica a partir de prototipos que permitan al usuario final interactuar con diferentes versiones del diseño.
- **Evaluar el prototipo a través del proceso y la experiencia de usuario.** Evaluar la usabilidad y experiencia de uso del prototipo a partir de técnicas cualitativas o cuantitativas.

4.2. Actores

Durante el proceso de diseño de la NUI y tomando como base los requisitos funcionales de desarrollo de interfaces interactivas se definieron los actores que interactúan en el ambiente de aprendizaje, los cuales se presentan en la Tabla 4-1, (ver Tabla 4-1).

Tabla 4-1. Actores participantes de la aplicación

Actor	Descripción
Usuario/Alumno	Persona que interactúa con el sistema y que recibe retroalimentación al usar las interfaces NUI que son implementadas en el ambiente de aprendizaje.
Reconocedor de gestos	Sistema que recibe imágenes de las manos del usuario y la procesa para realizar acciones y actividades relacionadas con el funcionamiento del ambiente de aprendizaje.
Interfaz kinestésica	Sistema encargado de la representación 3D de las manos del usuario dentro del ambiente de aprendizaje y dedicado a la interacción del usuario con los objetos 3D dentro de un entorno interactivo.

4.3. Análisis de requerimientos

Una de las actividades principales dentro de este proyecto es analizar e identificar los procesos que se llevan a cabo dentro de los ambientes de aprendizaje, así como la forma en que los usuarios llegan a interactuar en entornos gráficos 2D y 3D. Por ello, se especifican una serie de requisitos funcionales (ver Tabla 4-2) para el desarrollo de un prototipo de ambiente de aprendizaje, que posea características de interactividad mediante reconocimiento de gestos y kinestesia. Los requisitos funcionales se obtuvieron a través de una lluvia de ideas de diseño de software e interfaces centrado en el humano (DCH), donde existen principios de diseño que sirven como base esencial para la creación de software orientado en la mayor satisfacción y mejor experiencia de uso posible, con un mínimo esfuerzo por parte del usuario.

Tabla 4-2. Lista de requisitos funcionales

Requisito	Descripción	Prioridad
RF-01	El entorno debe de contar con una GUI fácil de utilizar.	Alta
RF-02	La NUI debe de ser fácil de usar.	Alta
RF-03	El ambiente de aprendizaje debe de contar con un escenario gráfico 3D interactivo con un sistema de simulación de físicas.	Alta
RF-04	El Sistema de interfaces NUI deberá de ser capaz de controlar la ejecución de actividades dentro del entorno interactivo.	Alta
RF-05	El ambiente de aprendizaje debe de contar con capacidad de guardado y carga de escenarios interactivos.	Media
RF-06	El ambiente de aprendizaje debe de contar con capacidad de ejecución y detención de código JavaScript para la creación de escenarios interactivos.	Media

4.4. Diagrama de contexto

A continuación, se presenta el diagrama de contexto del prototipo. Aquí se representa el contexto general del uso de las interfaces NUI y GUI con las que el usuario interactúa con el ambiente de aprendizaje que implementa reconocimiento de gestos. El ambiente depende de un banco de gestos para el funcionamiento de las interfaces interactivas (Ver Figura 4-2).

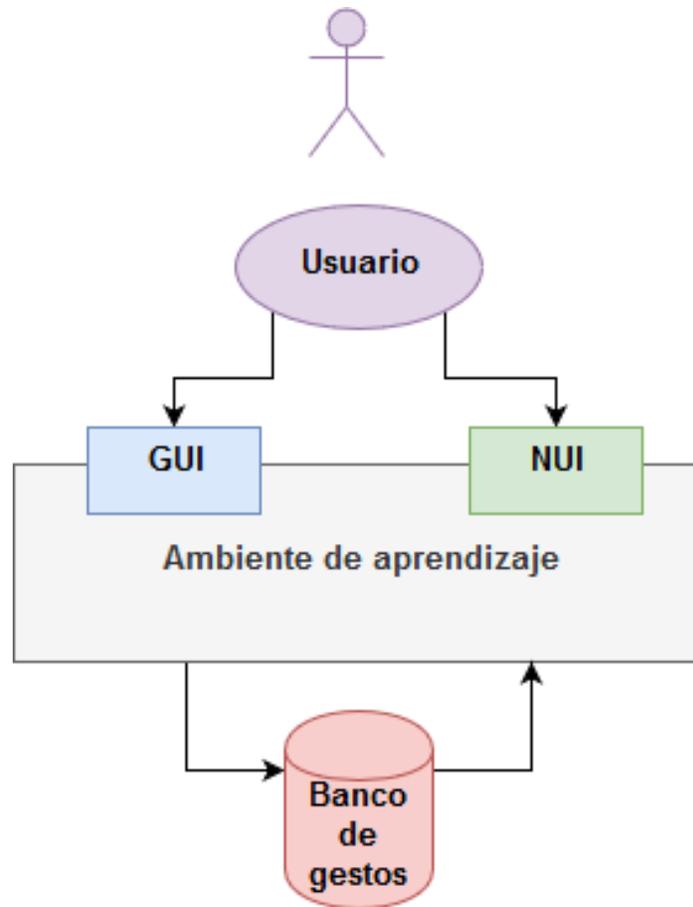


Figura 4-2. Diagrama de contexto

4.5. Arquetipos

Usuario: un usuario puede ser un estudiante o interesado en el área de estudio de la aplicación. A partir de la GUI el usuario navega a través de las opciones para el control y la interacción, se dedica a interactuar con la NUI del ambiente de aprendizaje y los recursos que la componen. La relación más importante del usuario está definida por la capacidad de este en utilizar en conjunto las interfaces NUI y GUI para la creación de escenarios interactivos personalizados dentro del software.

Escenario interactivo: la aplicación le muestra al usuario una GUI con las herramientas para la creación de escenarios interactivos a partir de la programación visual utilizando bloques. Se implementa las interfaces NUI basadas en el reconocimiento de gestos e interactividad kinestésica. Al escenario interactivo se le puede asignar un nombre, así como ser guardados

dentro del computador para después ser cargados cuando se desee. Permite la carga de una base de datos para la implementación de reconocimiento de gestos.

LeapLoop: Componente encargado en el control y obtención de datos de entrada del sensor Leap Motion.

GestureRecognition: Es el componente donde se lleva a cabo la detección de manos a través de Leap Motion para el proceso de datos que da como resultado el sistema de reconocimiento de gestos.

GesturePlayback: Componente encargado en la reproducción y grabación de gestos 2D y 3D dentro de un banco de gestos en formato JSON.

GesturesBank: Componente donde se registra una serie de gestos etiquetados y clasificados para ser utilizados a partir de las funciones del ambiente de aprendizaje.

HandsInteractivity: Componente que tiene las funciones para la aplicación de interactividad kinestésica en cuerpos y objetos dentro de los escenarios interactivos con simulación de físicas. También se encarga de generar la representación 3D de las manos del usuario dentro de un escenario.

4.6. Arquitectura

En la Figura 4-3 se muestra la arquitectura general del sistema, la cual contiene los componentes del prototipo del ambiente de aprendizaje, específicamente se enfoca en los componentes de las NUI y GUI. Cada capa y su contenido se describen a continuación:

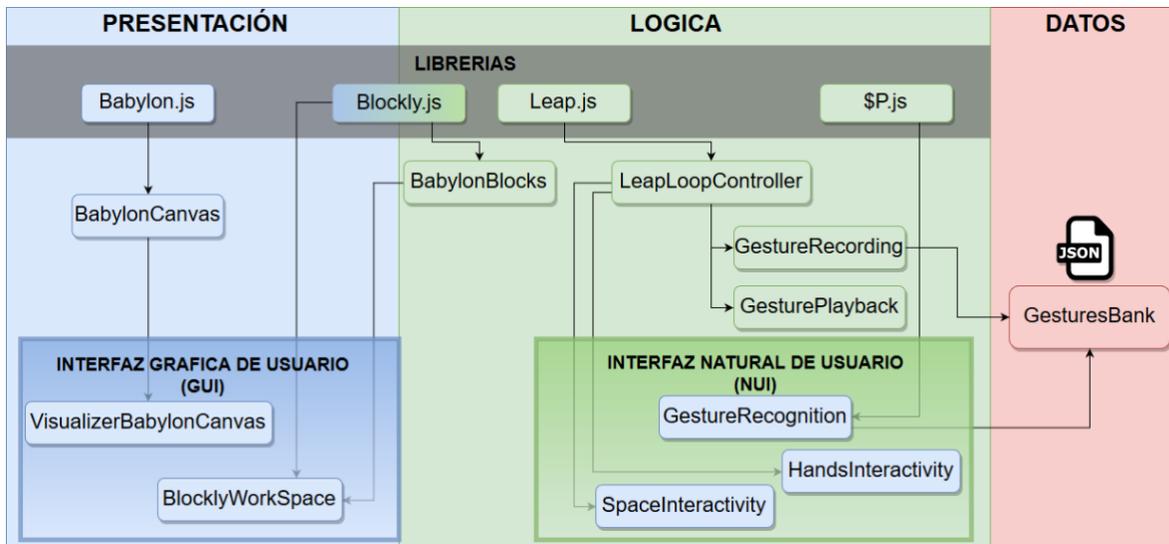


Figura 4-3. Diagrama general de componentes

Capa presentación: Dentro de la capa de presentación se presentan las interfaces en las que está formada la aplicación web, definidas por el uso de diseño de páginas web mediante el uso de HTML 5 y los *canvas* 2D, así como espacios donde se utiliza *canvas* para la presentación de escenarios 3D utilizando Babylon JS y un *framework* de renderizado de gráficos 3D basado en WebGL.

La GUI de la aplicación implementa las librerías de Blockly para ofrecer al usuario una interfaz de programación visual mediante bloques, mientras que Babylon JS está encargado del renderizado de gráficos 3D para presentar un escenario interactivo 3D con un sistema de simulación de eventos físicos.

Capa lógica: La capa lógica está compuesta por los componentes necesarios para la implementación de NUIs dentro de la aplicación. Dichos componentes están dedicados a la captura, procesamiento y reconocimiento de gestos, a la interacción kinestésica dentro del entorno interactivo 3D, así como la representación de los mismos. En la tabla 4-3 se muestra una lista de descripciones breves de los componentes de la capa lógica.

Tabla 4-3. Tabla de componentes de la capa lógica

Componente	Descripción
LeapLoopController	Encargado de la ejecución y control de datos del dispositivo LMC. Manipula los datos frame recibidos desde un webSocket por la API Leap.js.
GestureRecording	Componente que contiene funciones de interpretación y registro de datos transmitidos del componente LeapLoopController. Permite normalizar y guardar datos de gestos en formato JSON en una base de datos.
GesturePlayback	Provee las funciones necesarias para la reproducción de <i>frames</i> de datos del dispositivo. Se encarga de la lectura de etiquetas de tiempo de captura (<i>timestamps</i>) generadas por el LMC o almacenadas en un archivo JSON. Permite simular animaciones 2D y 3D de gestos.
GestureRecognition	Componente compuesto por funciones que implementan el algoritmo de reconocimiento de gestos \$P. Es el encargado de generar y procesar trazos de nubes de puntos de trayectorias realizadas por gestos.
HandsInteractivity	Encargado de ofrecer las funciones de interactividad kinestésica. Procesa información relacionada con la abertura de la palma de la mano (<i>grabbing</i>), distancia entre el dedo índice y pulgar (<i>picking</i>) o si el dedo índice apunta hacia alguna dirección (<i>pointing</i>).
SpaceInteractivity	Su funcionalidad consiste en delimitar las áreas de acción del dispositivo de acuerdo a parámetros como rangos y escalas dimensionales de un espacio interactivo a implementar. Ofrece áreas de acción: activas, pasivas o límite.
BabylonBlocks	Componente que permite la implementación de bloques de Blockly en el entorno gráfico desarrollado en Babylon JS. Posee un paquete de bloques para programar reglas de interactividad para las interfaces NUI y para programación de simulación de físicas dentro de la aplicación.

Capa datos: Esta capa está compuesta por un banco de gestos almacenados en un archivo JSON, en donde su lectura y escritura se lleva a cabo a partir de la interfaz natural de usuario con reconocimiento de gestos. Estos gestos son registrados a partir del algoritmo \$P.

4.7. Desarrollo de interfaz NUI con reconocimiento de gestos.

Este proyecto se enfoca principalmente en realizar reconocimiento de gestos de acuerdo a las trayectorias realizadas por el movimiento de las puntas de los dedos de una mano humana, simulando la información que se recibe al utilizar un dispositivo táctil, pero en este caso, que el proceso de reconocimiento sea aplicado dentro de un entorno 3D.

Esto último implica que debe de existir alguna forma de registrar la trayectoria de una mano no solo en una perspectiva 2D, sino también 3D.

Para el problema de localización de la posición de una mano de acuerdo a la profundidad de un entorno, este proyecto se ha enfocado en dar a conocer 2 propuestas: utilizar un algoritmo que pueda reconocer gestos realizados tanto en 2D como 3D y determinar campos de profundidad de un entorno a partir de áreas de interacción.

Para el reconocimiento de gestos se ha adaptado el algoritmo \$P para su uso en 3D, representando a un gesto como el desplazamiento de las puntas de los dedos de una mano, formando un trazo o una serie de puntos en un plano. A partir del dispositivo LMC, se obtienen características de las manos humanas que se encuentren dentro de su campo de visión. La información es almacenada en forma de series de fotogramas o *frames*, parecidos a tomar video, pero en su lugar se registran datos vectoriales de la posición, orientación, grados de inclinación de cada parte que compone a la mano humana, desde los huesos de la muñeca, las articulaciones y huesos de los dedos.

Esta información es procesada y registrada en un archivo en formato JSON, el cual contiene los datos vectoriales de cada *frame*, con su respectivo tiempo de captura. Esto permite que se creen animaciones de acuerdo a la duración del movimiento o gesto (ver Figura 4-4).



Figura 4-4. Archivo JSON como contenedor de gestos

Para el reconocimiento de los gestos 3D, se ha optado por separar el gesto realizado por las puntas de los dedos de una mano en sus diferentes perspectivas de acuerdo a los planos ortogonales XY y XZ. Sin embargo, implementar un sistema de reconocimiento de gestos dentro de entornos 3D implica también reconocimiento de la posición de la mano dentro de la misma en coordenadas 3D, por lo que es necesario delimitar en qué momento un gesto es usado. Se definen 3 estados dentro del área de visión del sensor; un estado donde las manos son representadas, pero la interactividad y gestos están desactivados (PASIVO), el estado en la que la interactividad esta activada, pero el reconocimiento de gestos se mantiene desactivado (ACTIVO) y el estado cuando se toca el límite frontal del LMC para activar reconocimiento de gestos (LIMITE) (ver Figura 4-5).



Figura 4-5. Representación de las áreas de interacción con respecto a la profundidad entre el usuario y el LMC.

4.8. Desarrollo de interfaz NUI con interactividad kinestésica

Si bien, el reconocimiento de gestos forma parte de una interfaz humana-máquina natural, este no proporciona capacidades de interacción kinestésicas dentro del entorno. Es decir, que el usuario pueda interactuar con los elementos que se encuentre dentro de un escenario virtual de la misma manera que se realiza en la vida real, como levantar un objeto con sus manos, moverlo de un lugar a otro, lanzarlo, etc. Esta es una característica que un sistema de reconocimiento de gestos no puede atender, puesto que este no utiliza información

relacionada a por ejemplo verificar, si las manos dentro de un entorno virtual colisionan o se encuentran cerca de un objeto interactivo. Por lo cual, crear un componente de interacción para esta clase de entornos también forma parte de la interfaz de este proyecto.

Para que el usuario final dentro de la aplicación pueda interactuar directamente con sus manos, se utiliza una representación tridimensional de las manos, que consiste de un modelo esquelético para el entorno gráfico 3D. Además, la interface cuenta con un componente dedicado a la interacción mediante eventos a partir de ciertas posturas y acciones de las manos. Este componente primero utiliza la información recibida por el sensor para evaluar la distancia entre el dedo pulgar y el dedo índice (*picking*). Después, evalúa que tanto la palma de la mano está cerrada o abierta (*grabing*) y si el dedo índice está apuntando a un objeto a distancia (*pointing*) mientras se determina si la posición de la mano virtual se encuentra cerca de un elemento interactivo con el que se pueda realizar alguna acción. Esto ofrece formas en la que el usuario puede interactuar con los elementos dentro del entorno y también están disponibles para ser utilizados para programar eventos personalizados.

4.9. Desarrollo de prototipo de aplicación Web

Al principio se construyó una página web para realizar pruebas y calibración del funcionamiento del sensor. Utilizando la API Leap JS se obtuvieron datos representativos desde un websocket en formato JSON. Los datos muestran la posición de las manos registradas en el campo de visión del sensor en coordenadas 3D.

Como el propósito principal del proyecto es el reconocimiento de gestos, se depura los datos obtenidos por relevancia. Realizar un gesto con las manos es una acción realizada principalmente a partir de los dedos de las manos. En el lenguaje de señas, por ejemplo, diferentes posiciones de los dedos, en conjunto con el desplazamiento de la palma de la mano representan una palabra. Al utilizar una pantalla táctil, la punta de los dedos de una mano es la principal parte del cuerpo que realiza la interacción, por lo que son características importantes a considerar. En este caso, solo se consideró la posición de la palma de la mano y la posición de las puntas de los dedos para la generación de características que representarían un gesto, haciendo una comparación homóloga al utilizar una pantalla táctil.

A partir de JavaScript, se programó una pequeña aplicación para la visualización de los datos en una representación 3D. Para ello se utilizó un motor de renderizado gráfico 3D para páginas web llamado Babylon JS. Babylon JS fue capaz de crear escenas 3D y de instancias de objetos tridimensionales, así como el control de físicas y partículas. Esto nos permitió el desarrollo de una GUI interactiva.

La información obtenida ya depurada por relevancia es registrada en un nuevo archivo JSON. Este nuevo archivo contiene una serie de fotoramas registrados al realizar un gesto sobre el sensor, donde cada frame es etiquetada por el tiempo de captura en milisegundos. Este archivo representa una animación 3D de un gesto y a partir de este archivo se desarrolló un reproductor de gestos que reproduce la animación registrada en el JSON creado.

A partir de este punto, lo siguiente consiste en crear un grabador de gestos que implementa el reproductor de animaciones de gestos, con la finalidad de registrar de forma individual el trayecto producido por los dedos de la mano al realizar un gesto. Esta información de desplazamiento es registrada de acuerdo a las dos perspectivas XY y XZ. Una perspectiva frontal y una perspectiva perpendicular a la mano.

El sistema de reconocimiento de gestos está desarrollado para que funcione mediante el dispositivo Leap Motion. Para ello se aplican algoritmos y técnicas de Aprendizaje Máquina utilizando una familia de librerías dedicadas al reconocimiento de gestos para prototipado rápido llamado \$P family.

El algoritmo, basado en el método “vecinos más cercanos” (KNN) con un sistema de comparación de plantillas geométricas, está enfocado principalmente al reconocimiento de patrones de gestos de la mano humana, más precisamente, de las trayectorias realizadas con los dedos al realizar un gesto. Originalmente pensado para su uso para interfaces 2D, procesa y visualiza el gesto de una mano como la interacción con una pantalla táctil de un dispositivo móvil o Tablet. Esto significa que la representación de un gesto es bidimensional.

En este proyecto a dicho algoritmo se le da un uso dentro de un ambiente de aprendizaje con entornos interactivos 3D, por lo que el algoritmo se adaptó para que funcionara dentro de entornos 3D, pero respetando sus funcionalidades básicas sobre la interpretación de un gesto, como si se tratase de una pantalla táctil. Para ello, se define una área dentro de la profundidad

del espacio 3D del sensor, para que este determine cuando se desea realizar un gesto e interactuar con la aplicación. Dentro de dicho límite, el usuario puede realizar un gesto, y en el momento de retirar su mano sobre el espacio para generar un gesto, este se procesa, se reconoce y se realiza una acción relacionada con dicho gesto.

La representación de gestos está compuesta por una serie de puntos que son interconectados entre sí. Para ello se genera la trayectoria de los dedos de la mano que luego es comparada por un banco de gestos, luego se determina que el gesto candidato sea aproximado a uno de los gestos modelo y se recibe un resultado mediante un puntaje de aproximación. Para identificar los gestos modelo, a estos antes se les asigna un nombre durante el proceso de etiquetación y se registra dentro del banco de gestos.

4.9.1. Algoritmo para Reconocimiento de Gestos en interfaces 3D

En el algoritmo \$P\$, el concepto de nube de puntos se describe como la representación de un trazo o trayectoria, que ignora el concepto de tiempo y ve a un gesto como un conjunto de puntos no ordenados, pero agrupados entre sí. Adoptando una vista libre del concepto de tiempo, permite que aspectos como el número de trazos, el orden y dirección se vuelvan irrelevantes. (Vatavu et al., 2012) Para hacer una analogía, esta representación es parecida al reconocimiento de caracteres utilizando sistemas de reconocimiento mediante mapas de bits utilizando un sensor óptico.

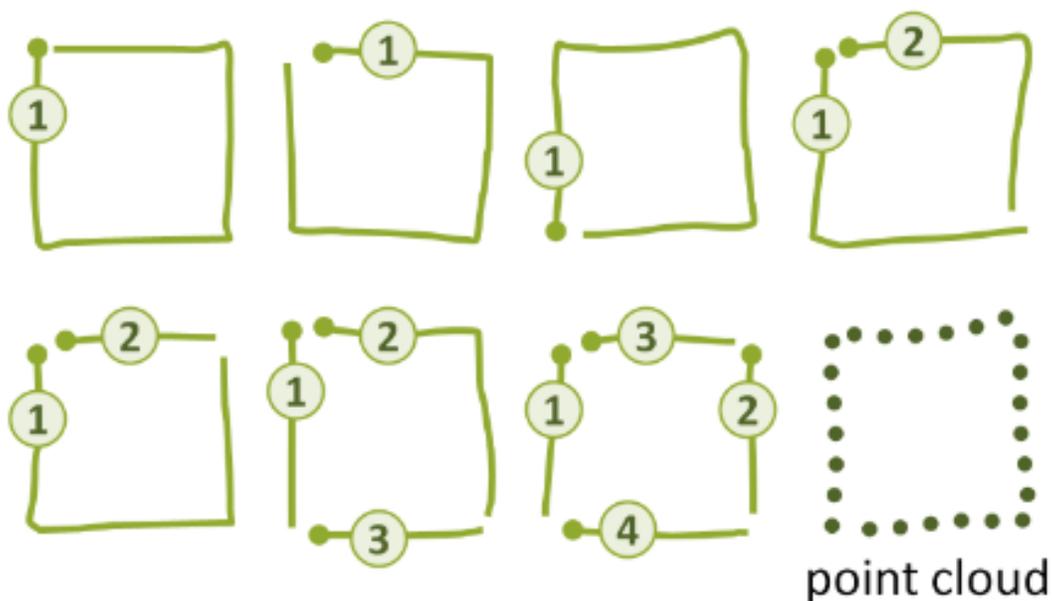


Figura 4-6. Representación de un gesto mediante trazos

Para demostrar mejor las ventajas al descartar el tiempo de ejecución de un gesto, la figura 4-6 ilustra diferentes formas para dibujar un cuadrado. Un cuadrado puede ser dibujado mediante gestos de 1, 2, 3 o 4 trazos con variaciones en cada trazo de forma individual en términos de orden y dirección. Cuando se observa dicho dibujo como una nube de puntos, esos detalles de ejecución pierden su significancia. Esto trae como consecuencia, la reducción de la complejidad de la estructura de datos y del proceso de clasificación (Vatavu et al., 2012).

El algoritmo \$P funciona tomando como base el concepto de trazo que a su vez está compuesto por una serie de puntos que representan una trayectoria de algún gesto realizado dentro de un espacio bidimensional expresada en la fórmula (1):

$$\{P_i = (x_i, y_i) | i = 1..n\} \quad (1)$$

Donde P_i es un punto dentro de un espacio de dos dimensiones definidas por (x, y) y que en conjunto con otros P_n puntos forman una nube de puntos que representan un trazado.

Como el tiempo de ejecución del gesto es descartado, la tarea del reconocedor se mantiene en emparejar una nube de puntos de un gesto candidato (C) con la nube de puntos de cada template (T) existente en un set de datos (corpus) y calcular una distancia de emparejamiento. Dichas distancias entre puntos son sumadas para obtener una distancia general M que determina el mejor resultado. En tradición al método de vecinos cercanos (Nearest-Neighbor), el template T más cercano de C es el que ofrece el resultado de una clasificación.

En \$P se define el emparejamiento entre dos nubes de puntos C y T como una función M , que asocia a cada punto $C_i \in C$ con un punto $T_j \in T$, $T_j = \Sigma(C_i)$. Ambas nubes de puntos C y T son normalizadas a una cantidad de puntos representativos n , para realizar pares exactos de dichos n puntos. En este caso se calcula el mejor caso de emparejamiento M de acuerdo a la suma de las distancias euclidianas para cada par de puntos realizados por M .

$$M = \sum_{i=1}^n \|C_i - T_j\| = \sum_{i=1}^n \sqrt{(C_{ix} - T_{jx})^2 + (C_{iy} - T_{jy})^2} \quad (2)$$

En la fórmula (2), j depende de i , pero por facilidad de notación solo iteramos sobre i y descartamos cualquier otra formalidad de notación considerando simplemente a C_i como un punto de la primer nube C (candidata) con el emparejamiento a un punto T_j de una segunda nube T (Témlate).

El algoritmo funciona a base de heurística egoísta (Greedy) denominada GREEDY-5 que busca solucionar el problema de asignación de forma eficaz y funciona con un tiempo lineal de $O(n^{2+\epsilon})$ de complejidad a base de los siguientes pasos:

Para cada punto en la primera nube de puntos (C_i), encontrar el punto más cercano respecto a una segunda nube (T_j) que aún no ha sido pareada. Una vez que dicho punto C_i posee pareja, se continua con C_{i+1} hasta que todos los puntos de C son pareados. La complejidad de este ejercicio de cómputo se encuentra en el tiempo de $O(n^2)$ hasta que la búsqueda lineal entre C y T sea finalizada.

Este algoritmo se ejecuta múltiple veces en diferentes puntos de inicio con la finalidad de obtener el mínimo de intentos de emparejamiento en cada ejecución. Si C_k es el punto de inicio entonces M se expresa como:

$$M = \sum_i \|C_i - T_j\| = \sum_{i=k}^n \|C_i - T_j\| + \sum_{i=1}^{k+1} \|C_i - T_j\| \quad (3)$$

De acuerdo a la fórmula (3), i es donde se pasa circularmente por todos los puntos en C . Por ello se presenta un parámetro ϵ para controlar el número de ejecuciones. Si $\epsilon = 0$, el algoritmo solo se ejecuta una vez (por el punto $C_1, C_2 \dots C_n$ en orden). Si $\epsilon = 1$ entonces el algoritmo se ejecuta n veces (donde cada punto C_i tiene la primera oportunidad de emparejarse).

Si $\epsilon < 1$ entonces el algoritmo se ejecuta $n^\epsilon < n$ veces. Usando este formalismo, la complejidad de la heurística GREEDY-5 se expresa como $O(n^{2+\epsilon})$, la cual se encuentra entre $O(n^2)$ y $O(n^3)$.

Otro aspecto importante de la heurística GREEDY-5 es la implementación de pesos de confianza a las distancias euclidianas entre cada par de puntos:

$$M \sum_i w_i \cdot \|C_i - T_j\| \quad (4)$$

En la fórmula (4), los pesos w_i registran la confianza de cada par (C_i, T_j) calculado durante una ejecución. El primer emparejamiento se le asigna un peso $w_1 = 1.0$ (de confianza alta) a causa de que confiamos en ella: el primer punto contiene todos los datos necesarios en orden para realizar decisiones y elegir su par T_j más cercano. Mientras el algoritmo progresa, las opciones restantes de emparejamiento de la primera nube C para emparejarse con una segunda nube T se ven reducidas. Por lo tanto, no se pueden confiar de estos pares completamente por lo que se les asigna pesos de confianza en valores entre 0 y 1. Por ejemplo, el último punto en ser pareado solo tiene una única opción (el último punto de una segunda nube que falta ser pareado), por lo que la confianza en esta alineación también debería ser pequeña. Por lo tanto, se adopta un esquema de ponderación lineal mostrada en la fórmula (5):

$$Mw_i = 1 - \frac{i - 1}{n} \quad (5)$$

Donde $i = 1 \dots n$ almacena el paso actual del algoritmo y n representa la cantidad de pasos (puntos) que componen una nube de puntos.

Como se ha mostrado anteriormente, \$P ofrece una solución de reconocimiento de gestos expresadas como coordenadas (x, y) en una nube de puntos en 2D, sin embargo, \$P puede ser adaptado para aceptar una tercera dimensión y así ser implementado dentro de entornos gráficos 3D. Para ello, se le agrega una propiedad z a cada uno de los puntos que componen a una nube de puntos. Esto se representa mediante la fórmula (6):

$$M = \sum_{i=1}^n \|C_i - T_j\| = \sum_{i=1}^n \sqrt{(C_{ix} - T_{jx})^2 + (C_{iy} - T_{jy})^2 + (C_{iz} - T_{jz})^2} \quad (6)$$

En donde M es la suma de las distancias euclidianas para cada par de puntos en el espacio, la cual es representada como un vector entre cada par de puntos (C_i, T_j) por lo tanto cada punto $C_i \in C$ como de $T_i \in T$ se representa como en la fórmula (7):

$$\{P_i = (x_i, y_i, z_i) | i = 1..n\} \quad (7)$$

Donde P_i es un punto dentro de un espacio de tres dimensiones definidas por (x, y, z) y que en conjunto con otros P_n puntos forman una nube de puntos que representan un trazado, ya sea para C como de T .

A continuación, se muestra el funcionamiento del algoritmo a través de pseudocódigo. En este caso, **punto** es una estructura compuesta por las propiedades x, y, z e ID . **ID** es el índice $\{1, 2, \dots\}$ que se le asigna a un punto dentro de una nube de puntos. Una lista de puntos se representa en **puntos** y **templates** es una lista de puntos que posee una clase de gesto representativo.

El pseudocódigo **Algoritmo \$P** muestra la función principal del algoritmo. Consiste en ejecutar el emparejamiento de puntos entre un conjunto **templates** a partir de las reglas de clasificación de vecinos cercanos. Da como resultado un puntaje normalizado entre 0 y 1 donde 1 significa un emparejamiento perfecto.

```

Algoritmo $P (puntos, templates)
1: n = 32
2: score = ∞
3: Normalizar (puntos, n)
4: FOR template en templates DO
5:     Normalizar (template, n)
6:     m = GreedyCloudMatch (puntos, template, n)
7:     IF score > m THEN
8:         score = m
9:         resultado = template.ID
10: RETURN (resultado, score)

```

El pseudocódigo **GreedyCloudMatch** es una función encargada de emparejar dos nubes de puntos (**puntos** y **template**) realizando alineaciones repetidas entre sus puntos. Cada alineación empieza con un índice i diferente. El parámetro ϵ entre el 0 y el 1 controla el número de alineamientos realizado durante pruebas. Por defecto a ϵ se le asigna el valor 0.5.

```

GreedyCloudMatch (puntos, template, n)
1:  $\epsilon = 0.5$ 
2: paso =  $\lfloor n^{1-\epsilon} \rfloor$ 
3: min =  $\infty$ 
4: Normalizar (puntos, n)
5: FOR i = 0 TO n-1 paso DO
6:     d1 = CloudDistance (puntos, template, n, i)
7:     d2 = CloudDistance (puntos, template, n, i)
8:     IF score > m THEN
9:         score = m
10:         resultado = template.ID
11: RETURN (resultado, score)

```

El pseudocódigo **CloudDistance** es la función encargada del cómputo del coste mínimo de las alineaciones entre **puntos** y **template** empezando desde el punto inicio. Asignando pesos de confianza decrecientes a cada emparejamiento de puntos.

```

CloudDistance (puntos, template, n, inicio)
1: pareado = new bool [j]
2: suma = 0
3: i = inicio //punto de partida
4: DO
5:     min =  $\infty$ 
6:     FOR EACH j que no esta pareado[j] DO
7:         d = EuclideanDistance (puntos i , template j)
8:         IF d < min THEN
9:             min = d
10:            Indice = j
11: pareado[indice] = true
12: confianza =  $1 - (i - inicio + n) \text{MOD } n / n$ 
13: suma = suma + (confianza * min)
14: i = (i+1) MOD n
15: UNTIL i == inicio
16: RETURN suma

```

El pseudocódigo **Normalizar** consiste en el proceso de **remuestreo**, **reescalado** y **trasladado** hacia el origen de nubes de puntos de gestos.

```

Normalizar (puntos, n)
1: puntos = Remuestreo (puntos, n)
2: Escala (puntos)
3: TrasladarOrigen (puntos, n)

```

El pseudocódigo **Remuestreo** toma los datos de un trazado de un gesto y lo generaliza a una n cantidad de puntos uniformemente separados. Por defecto el parámetro $n = 32$. Para ello,

se requiere obtener las distancias euclidianas entre dichos puntos, proceso realizado por la función **getLongitud**.

```

Remuestreo (puntos,n)
1: I = getLongitud(puntos) / (n-1)
2: D = 0
3: puntosNuevos = puntos[0]
4: FOR EACH  $p_i$  en puntos tal que  $i \geq 1$  DO
5:     IF  $P_i.ID == P_{i-1}.ID$  THEN
6:         d = DistanciaEuclidiana ( $P_{i-1}, P_i$  )
7:         IF  $(D+d) \geq I$  THEN
8:              $q.x = P_{i-1}.x + ((I-D)/d) * (P_i.x - P_{i-1}.x)$ 
9:              $q.y = P_{i-1}.y + ((I-D)/d) * (P_i.y - P_{i-1}.y)$ 
10:             $q.z = P_{i-1}.z + ((I-D)/d) * (P_i.z - P_{i-1}.z)$  //si se implementa
            dimensiones 3D
11:         Adjuntar (puntosNuevos, q)
12:         Insertar (puntos, i, q) // q sera el siguiente punto  $p_i$ 
13:         D = 0
14:     ELSE D = D+d
15: RETURN puntosNuevos

```

```

getLongitud (puntos)
1: d = 0
2: FOR EACH  $P_i$  en puntos de forma que  $i \geq 1$  DO
3:     IF  $P_i.ID == P_{i-1}.ID$  THEN
4:         d = d + DistanciaEuclidiana ( $P_{i-1}, P_i$ )
5: RETURN d

```

El pseudocódigo **Escala** es una función que computa los valores mínimos y máximos de las dimensiones x, y, z de puntos, para realizar un reescalado de las dimensiones de una nube de puntos, preservando la geometría del gesto en un cuadro (o cubo) dimensional delimitado entre valores del 0 *al* 1.

```

Escala (puntos)
1: xmin = ∞, xmax = 0, ymin = ∞, ymax=0, zmin = ∞, zmax=0 //si se implementa
   dimensiones 3D
2: FOR EACH P en puntos DO
3:     xmin = MIN(xmin, P.x)
4:     ymin = MIN(ymin, P.y)
5:     zmin = MIN(zmin, P.z) //si se implementa dimensiones 3D
6:     xmax = MAX(xmax, P.x)
7:     ymax = MAX(ymax, P.y)
8:     zmax = MAX(zmax, P.z) //si se implementa dimensiones 3D
9: escala = MAX(xmax - xmin, ymax - ymin, zmax - zmin)
10: FOR EACH p en puntos DO
11:     P = ((p.x - xmin)/escala, (p.y - ymin)/escala, (p.z - zmin)/escala)

```

El pseudocódigo **TrasladarOrigen** como su nombre lo indica, traslada los puntos a un mismo origen 2D (0,0) o 3D (0,0,0).

```

TrasladarOrigen (puntos,n)
1: c = (0,0)
2: FOR EACH p en puntos DO
3:     c = (c.x + p.x, c.y + p.y, c.z + p.z) //si se implementa dimensiones 3D
4: c = (c.x/n, c.y/n, c.z/n)
5: FOR EACH p en puntos DO
6:     P = (px -cx, py- c.y, p.z - c.z)

```

4.9.2. CREA tus propios gestos.

CREA consiste en la creación de un corpus de reconocimiento de gestos, donde a partir de la técnica de *programación por ejemplos* (Lieberman, 2001), se crea un archivo JSON que representa el corpus de gestos con su respectiva etiquetación, la cual puede ser cargado en los entornos del apartado JUEGA. Para crear gestos, el usuario usa sus manos sobre el sensor LMC, mientras que el software automáticamente procede a capturar datos y etiquetar por tiempo (en milisegundos) los fotogramas (frames) de lo que dura el gesto. La figura 4-7 muestra la interfaz del entorno de desarrollo CREA.



Figura 4-7. Entorno de Desarrollo de CREA

CREA permite al usuario visualizar los gestos producidos por medio de un reproductor gráfico que muestra una representación animada del gesto en 3D, que incluye la visualización

de las trayectorias realizadas por las puntas de sus dedos en perspectivas XY y XZ. El visualizador también apoya al usuario con un verificador incorporado que mide la exactitud de los gestos, comparándolo con el contenido del corpus generado.

Para crear un archivo JSON que represente el corpus de gestos, el usuario deberá de generar un gesto y después etiquetarlo con un nombre. Se puede repetir el registro del gesto y el proceso de etiquetación las veces que se desee. El archivo JSON resultante se guarda en la computadora y este puede ser utilizado dentro del entorno JUEGA como parte de la interfaz de interacción, la cual implementa el reconocimiento de los gestos.

4.9.3. JUEGA y diviértete.

JUEGA es el entorno interactivo que implementa una interfaz de interacción mediante reconocimiento de gestos y una interfaz visual orientada a la programación mediante bloques (ver Figura 4-8).

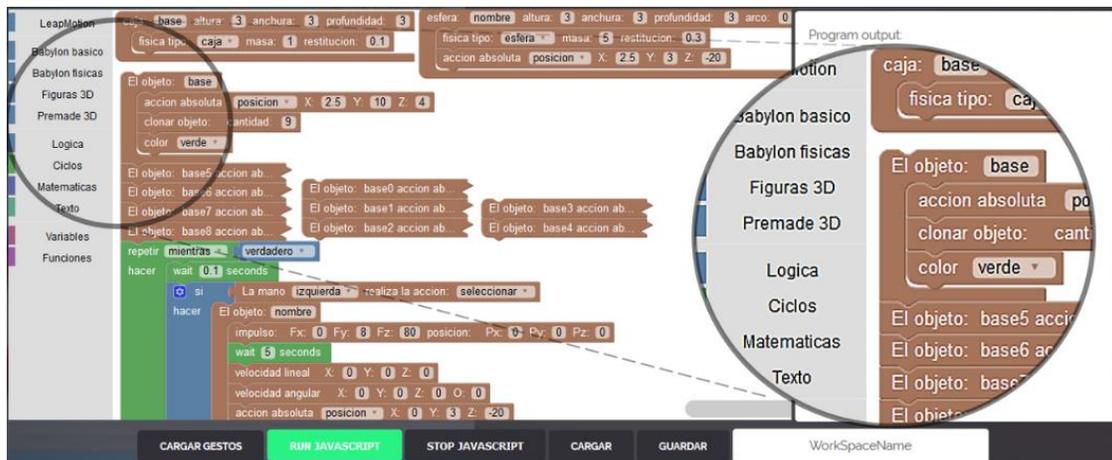


Figura 4-8. Entorno de programación visual con Blockly

A partir de los bloques, el usuario final construye elementos personalizados dentro del entorno, los cuales poseen físicas y pueden interactuar entre sí. También hay bloques para programación de la lógica y control de eventos que implementan la interfaz de interacción. Esto con el fin de que el usuario mediante gestos manipule e interactúe con lo que ocurre en el entorno 3D utilizando directamente sus manos de una forma más kinestésica, o bien utilizando el sistema de reconocimiento de gestos y relacionarlos a eventos o acciones en pantalla, a partir de la perspectiva de un avatar (ver Figura 4-9).

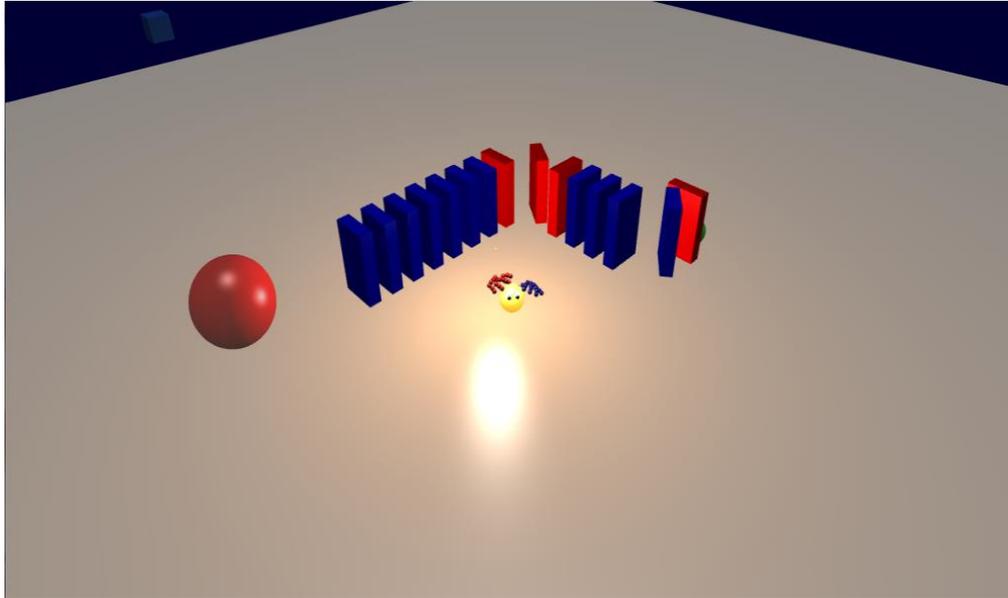


Figura 4-9. Escena 3D con interacción kinestésica desde un avatar.

JUEGA posee opciones para guardar y cargar el trabajo realizado. También posee opciones para cargar un archivo JSON para la asignación del corpus del sistema de reconocimiento de gestos, por lo que CREA y JUEGA pueden ser utilizados en conjunto.

Capítulo 5

5. Pruebas

En este capítulo se presenta la metodología de evaluación, así como los resultados obtenidos a partir de pruebas para verificar la validez de las interfaces GUI y NUI del ambiente de aprendizaje a través de heurísticas basadas en el diseño centrado en el humano.

5.1. Caso de estudio: Creación y ejecución de reconocimiento de gestos

Cuando un usuario quiere utilizar el ambiente de aprendizaje, debe de acceder primero al área de **CREA tus propios juegos**, donde la plataforma le permite al estudiante en una página web crear sus propios gestos a partir de programación por ejemplos. La programación por ejemplos se refiere a la capacidad de programar el comportamiento del software a partir de ejemplos o demostraciones. El usuario con la repetición de gestos dentro del campo de visión del sensor puede crear un corpus de gestos que pueden ser interpretados por la aplicación y ser utilizados para actividades o acciones personalizadas.

Una vez que el usuario haya realizado un gesto sobre el sensor, el sistema muestra en pantalla una animación 3D esquelética de la mano, así como animaciones 2D de las trayectorias de las puntas de sus dedos mediante un reproductor incorporado. Este reproductor permite reproducir y guardar los datos de gestos realizados en un archivo en formato JSON.

Para crear un corpus de gestos, el usuario deberá de generar un gesto y después etiquetarlo con un nombre. Esto se realiza escribiendo un nombre en el apartado etiqueta y después se presiona el botón agregar.

Se puede repetir el proceso de registro de gestos y la etiquetación las veces que se desee. Realizar esto varias veces da como resultado la creación de un corpus de gestos clasificados y listos para su uso dentro del ambiente de aprendizaje en el apartado JUEGA y diviértete, como parte de la interfaz de interacción, la cual implementa el reconocimiento de gestos. Este corpus de gestos se almacena en un archivo JSON el cual se descarga y guarda en la computadora.

5.2. Caso de estudio: Uso de interfaces de usuario naturales para la interacción dentro de un entorno de aprendizaje.

Para usar las interfaces interactivas, el usuario deberá de posicionar sus manos encima del sensor Leap Motion. Dentro de la interfaz JUEGA y diviértete, aparecerá dentro del entorno interactivo una representación 3D de las manos del usuario en tiempo real. El usuario debe de programar dentro de un escenario 3D los objetos interactivos con simulación de físicas. Para ello utiliza una interfaz de programación por bloques que implementa Blockly.

El escenario 3D representa el espacio en el que el usuario puede crear objetos e interactuar con sus manos a partir de las interfaces NUI. Este escenario posee un avatar, un personaje redondo de color amarillo que representa al usuario dentro del entorno interactivo. A partir de este avatar, el usuario puede identificar su posición y orientación dentro del espacio y puede cambiar la perspectiva de la cámara con respecto al avatar, desde una perspectiva aérea en tercera persona a una perspectiva más cercana en primera persona. El movimiento del personaje se controla a partir del AWSD del teclado, mientras que la cámara en primera persona se controla con el mouse. Para la interactividad con los elementos, así como la representación de las manos dentro del entorno, el usuario deberá de colocar sus manos sobre el sensor y estas aparecerán como las manos del avatar dentro del escenario interactivo 3D.

La interactividad se lleva a cabo gracias a 2 interfaces NUI: el sistema de reconocimiento de gestos y el sistema de interactividad kinestésica. Ambas en conjunto le proporcionan al usuario diversas formas de interacción con los elementos en pantalla. El usuario puede crear reglas de interacción a partir de la programación por bloques. La interacción kinestésica permite interactuar con los elementos en pantalla, ya sea con el dedo índice o pulgar (*picking*), con la palma de la mano (*grabing*) o realizar un gesto de apuntar (*pointing*). Estas opciones también pueden ser utilizadas para programar reglas dentro del entorno interactivo.

Por otra parte, el uso del reconocimiento de gestos del usuario es más completo. Se permite la carga de un banco de gestos creado por el mismo usuario a partir de CREA, así como la capacidad de utilizar dicho banco de gestos dentro del ambiente interactivo, creando una serie de reglas para asociar gestos con alguna acción o proceso que afecte el comportamiento de objetos y del entorno interactivo en sí.

El diseño y creación de un ambiente interactivo se lleva a cabo a partir del uso de la interfaz de programación visual por bloques. Esta forma de programación permite al usuario desarrollar su capacidad de abstracción y algorítmica para la creación de entornos interactivos. Esta interfaz le plantea al usuario la metáfora de que un programa está compuesto por una serie de pasos o trozos de código que en conjunto y en cierto orden forman un proceso computacional. Esta representación metafórica de la programación le permite al usuario visualizar fácilmente los pasos que son usados para crear algo dentro del entorno interactivo.

Se le presenta al usuario dentro de la interfaz de programación, una serie de bloques que involucra el uso de las interfaces NUI desarrolladas en este proyecto. Los bloques son para la implementación de reconocimiento de gestos, detección de las manos dentro del campo de visión, la interactividad kinestésica con los objetos del entorno, etc. Así como bloques constructores de objetos, bloques de propiedades básicas e implementación de físicas, así como bloques que definen funciones lógicas, algorítmicas y de relación entre objetos.

Respecto al ejercicio de programar un entorno interactivo, esto se puede dividir en 3 segmentos (ver figura 5-1):

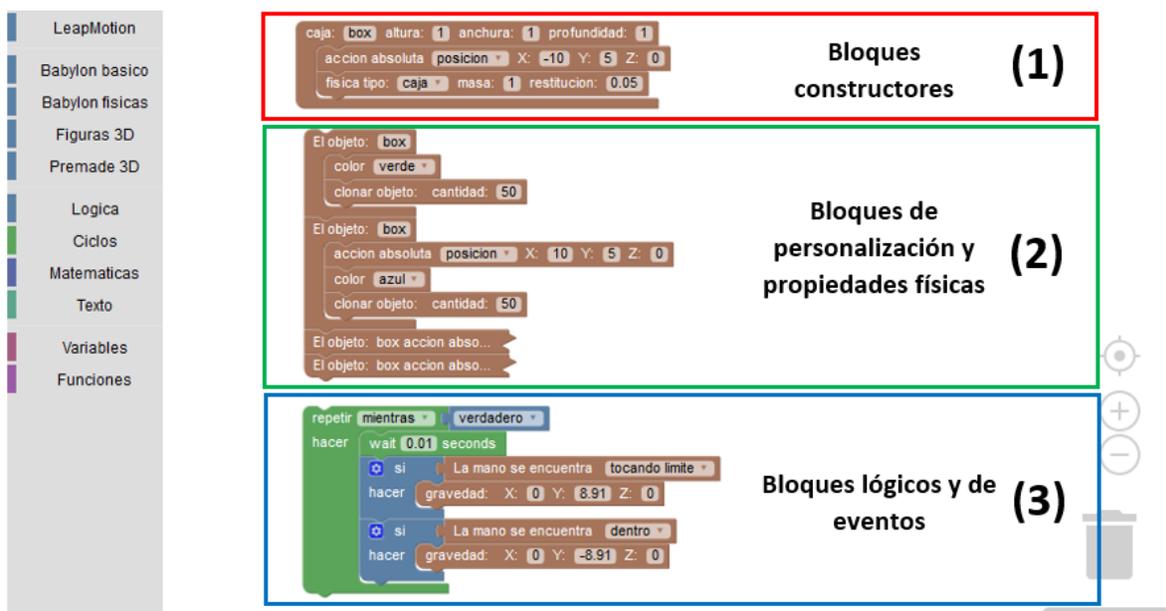


Figura 5-1. Segmentos para programar un entorno interactivo en JUEGA y diviértete.

1. **Creación de constructores de elemento 3D:** Se utilizan bloques para la construcción de figuras geométricas con sus respectivas dimensiones. Se crea instancias de figuras 3D y se les asigna un nombre.
2. **Personalización de elementos 3D y del entorno:** Ya sea durante o después del proceso de construcción, se usan bloques que permiten personalizar el aspecto y las propiedades físicas de una figura. Permiten definir el comportamiento del objeto a partir del sistema simulador de físicas. Para que la gravedad simulada haga efecto, por ejemplo, se necesita que al objeto se le asigne una masa.
3. **Creación del comportamiento lógico de entorno mediante eventos:** Se realiza la programación de la lógica de eventos y las interacciones que pueden ocurrir por las acciones del usuario. Se programa a partir de bloques de secuencias y funciones lógicas y permite crear animaciones en el entorno, o programar eventos que utilicen la interfaz de interacción e implementar reconocimiento de gestos.

5.3. Pruebas de validez de GUI y NUIs del sistema

El diseño centrado en el humano (DCH) tiene como objetivo el diseñar productos interactivos que sean fáciles, efectivos y disfrutables de usar, así como óptimos de la interacción del usuario con un sistema y su entorno. La literatura en DCH ha propuesto abstracciones acerca de los aspectos o principios de diseño como guías de lo que se debe y no se debe hacer al diseñar un sistema (Castro et al., 2018; Norman, 2013).

Por ello, nuestra investigación evalúa las interfaces propuestas de acuerdo a los principios básicos de interacción a partir de una matriz ponderada. Este método de evaluación tiene una larga tradición en la ingeniería de software y en la literatura de sistemas de información (Wieringa, Maiden, Mead, & Rolland, 2006).

Para la evaluación, se utiliza la siguiente escala de calificación a partir del cumplimiento de los aspectos o principios mencionados anteriormente.

- Si cumple: +2 puntos.
- Parcialmente cumple: +1 punto.
- No cumple: 0 puntos.

Tabla 5-1. Resultados de evaluación de aspectos de las interfaces del prototipo.

Aspectos	Criterios			
	GUI		NUI	
	Entorno gráfico interactivo 3D	Interfaz de programación gráfica por Bloques	Interfaz de interacción kinestésica 3D	Interfaz de interacción por reconocimiento de gestos.
Visibilidad	2	2	2	2
Retroalimentación	2	2	2	1
Restricciones	2	1	1	1
Consistencia	2	2	2	2
Asequibilidad	2	1	2	2

De acuerdo a los resultados de evaluación mostrados en la tabla 5-1, la visibilidad se cumple en las NUI y GUI de la aplicación. El uso de interfaces gráficas 3D permite al usuario observar con claridad el comportamiento de la aplicación con sus acciones. La representación de las manos en 3D y el avatar ayudan a la orientación visual de donde se encuentra el usuario dentro del entorno interactivo y brindan asistencia de cómo actuar en determinadas circunstancias (+2 puntos). La GUI de programación por bloques cumple este aspecto, presenta bloques para la codificación clasificados por tipo en una lista de forma organizada (+2 puntos).

El aspecto de la retroalimentación cumple en la mayoría de las interfaces de la aplicación. En conjunto las NUIs y el entorno interactivo 3D retroalimentan al usuario de sus acciones, a partir de la representación de un modelo 3D de manos en tiempo real (+2 puntos). Por otra parte, la interfaz de reconocimiento de gestos no posee ninguna forma de representación visual y depende de la activación de eventos para producir retroalimentación al usuario (+1 punto). La GUI de programación por bloques, reproduce efectos de audio y retroalimentación

visual cuando se conectan los bloques, e indicaciones cuando se está ejecutando código o existe algún error, por lo que cumple con este aspecto (+2 puntos).

Las restricciones son limitadas en las interfaces naturales, puesto que su objetivo es el de ofrecer libertad creativa para crear e interactuar. Esto ocasiona que no se definan con exactitud las posibilidades de uso de la aplicación (+1 punto). Sin embargo, existen excepciones en la GUI del entorno interactivo 3D, donde su propósito es el de delimitar las dimensiones del escenario y las áreas en las que el usuario puede llegar a interactuar desde la perspectiva de un avatar (+2 puntos).

La consistencia cumple dentro de las GUI, donde el mismo proceso de codificación de escenarios y los eventos es realizado sin importar el tipo de ejercicio, utilizando botones y funciones vistas en otras GUI fáciles de familiarizar como guardar, reproducir, editar y borrar. El usuario puede personalizar como interactúa con el entorno mediante las NUI, pero el proceso de creación e implementación de kinestesia y gestos es natural y siempre es el mismo (+2 puntos).

Las NUI cumplen con el aspecto de la asequibilidad, ya que la implantación de reconocimientos de gestos, así como la representación de la mano dentro del entorno interactivo, son por naturaleza, mecanismos de representación de como el humano interactúa con el entorno (+2 puntos). La GUI de programación por bloques, por otra parte, representa metafóricamente a la programación como la unión lógica de distintos elementos relacionados entre sí. Los bloques poseen señalamientos de que se pueden unir o ser insertados uno sobre el otro. Sin embargo, no todos los usuarios poseen la capacidad de abstracción para este tipo de conceptos, ocasionando ciertas dificultades de uso (+1 punto).

Capítulo 6

6. Conclusiones y trabajo futuro

6.1. Conclusiones

El desarrollo del pensamiento computacional y la búsqueda de mejores formas de educar a las nuevas generaciones sigue siendo una tarea demandante. En este trabajo, se presenta un enfoque de las implementaciones de interfaces humano-máquina naturales para el aprendizaje de lógica algorítmica y programación. Aplicando evaluaciones de diseño centrado en el humano, se demuestra que las NUI y GUI presentadas en este proyecto cumplen con los principios básicos de interacción natural. Sin embargo, el proceso de DCH es altamente empírico, donde la toma de decisiones es basada en el conocimiento que se tiene de los usuarios y del contexto en el que se utilizará un producto.

La aplicación fue desarrollada en JavaScript, utilizando la API de Leap Motion para la recepción de los datos de entrada. La interfaz de programación visual se implementó con Blockly (Fraser & others, 2013). Para la presentación del entorno gráfico, se utilizó el framework de gráficos 3D para aplicaciones web llamada Babylon JS (Catuhe et al., 2014), componente principal para la creación de escenarios tridimensionales, así como la representación de manos humanas dentro del entorno.

En este proyecto se realizaron investigaciones respecto al uso de nuevas tecnologías de interfaces de usuario naturales, así como la búsqueda de principios básicos y heurísticas necesarias para el desarrollo y análisis de un prototipo de un ambiente de aprendizaje, que cumpla con dichos requisitos para la implementación de interfaces naturales.

Se desarrolló un componente de software, que consiste en un algoritmo para reconocimiento de gestos en interfaces 2D y 3D el cual funciona a partir del procesamiento de datos generados por un gesto como una trazo conformado por una nube de puntos utilizando una aproximación del algoritmo húngaro con el uso de heurísticas para “clouds matching” en un sistema de comparación de plantillas geométricas.

El prototipo resultante, el cual se diseño y se analizo a base de los principios básicos del diseño orientado con el humano, consiste en la creación de escenarios 3D interactivos con simulación de eventos de física mediante la creación de elementos tridimensionales con un lenguaje de programación visual, esta también compuesto por un entorno de desarrollo de corpus de gestos, permitiendo al usuario crear sus propios modelos de gestos. Esto ultimo tiene como fin en que el corpus de gestos generado sea el medio de interacción personalizada entre el uso y la aplicación,

6.2. Trabajo a futuro

Como se mencionaba anteriormente, el proceso del DCH es un proceso empírico que requiere de iteraciones de desarrollo para cumplir los requerimientos de los usuarios y del contexto de uso del software interactivo. La creación de un prototipo con implementación de GUI e NUI de uso natural es solo la herramienta necesaria para realizar posteriores evaluaciones. Por ello, como trabajo futuro se planea realizar evaluaciones con grupos de estudiantes de educación básica a nivel secundaria, para analizar mejor el impacto de la herramienta y medir aspectos de usabilidad, motivación y satisfacción que se experimenta con el uso de este tipo de interfaces.

Bibliografía

- Aigner, R., Wigdor, D., Benko, H., Haller, M., Lindlbauer, D., Ion, A., ... Koh, J. T. K. V. (2012). Understanding Mid-Air Hand Gestures : A Study of Human Preferences in Usage of Gesture Types for HCI. Microsoft Research Tecnical Report MSR-TR-2012-111, (November), 10. Retrieved from <http://research.microsoft.com/apps/pubs/?id=175454>
- Bachmann, D., Weichert, F., & Rinkeauer, G. (2018). Review of three-dimensional human-computer interaction with focus on the leap motion controller. *Sensors (Switzerland)*, 18(7), 1–39. <https://doi.org/10.3390/s18072194>
- Bontá, P., Papert, A., & Silverman, B. (2010). Turtle, art, turtleart. In *Proc. of Constructionism 2010 Conference*.
- Bowe, S. J., Antoniou, M. N., Garrett, C., Huber, B., Kaufman, J., & Tarasuik, J. (2015). Young children's transfer of learning from a touchscreen device. *Computers in Human Behavior*, 56, 56–64. <https://doi.org/10.1016/j.chb.2015.11.010>
- Cabero, J. (2006). Bases pedagógicas del e-learning. *Revista de Universidad y Sociedad Del Conocimiento*, 3, 1–10.
- Castro, L. A., Rodríguez, M. D., & Computación, A. M. D. E. (2018). *Interacción Humano-Computadora y Aplicaciones en México* (primera ed). Academia Mexicana De Computación, A,C.
- Catuhe, D., Rousseau, M., Lagarde, P., & Rousset, D. (2014). Babylon.js a 3D engine based on webgl and javascript.
- Chifor, M., & Stefanut, T. (2015). Immersive Virtual Reality application using Google Cardboard and Leap Motion technologies. *Romanian Conference on Human-Computer Interaction*, 114–120.
- Clarke, S., Dass, N., & Chau, D. H. P. (2016). Naturalmotion: Exploring gesture controls for visualizing time-evolving graphs. *Proceedings of IEEE VIS (Poster Session)*.
- Davis, A. (2014). Leap Motion SDK. Retrieved from blog.leapmotion.com/
- Ebner, M., & Spot, N. (2015). Game-Based Learning with the Leap Motion Controller. (October), 555–565. <https://doi.org/10.4018/978-1-4666-9629-7.ch026>
- Fonteles, J. H., Sousa, É. S., & Rodrigues, M. A. F. (2015). Visual and Interactive Performance of Particles Conducted by the Leap Motion for an Orchestral Arrangement. In *2015 XVII Symposium on Virtual and Augmented Reality* (pp. 255–264).
- Fraser, N., & others. (2013). Blockly: A visual programming editor. URL: <https://code.google.com/p/blockly>.

- Fuel, N. (2018). Tynker. Retrieved July 25, 2019, from <https://www.tynker.com/>
- Gamefroot - game creator social and mobile gaming. (2017). Retrieved July 25, 2019, from <https://www.gamefroot.com/>
- Games, Y. (2015). GameMaker Studio. Retrieved July 25, 2019, from <https://www.yoyogames.com/gamemaker>
- Grover, S., & Pea, R. (2013). Computational thinking in K--12: A review of the state of the field. *Educational Researcher*, 42(1), 38–43.
- Guerineau, D., & Abrams, H. (2012). *Learn GameSalad for IOS: Game Development for iPhone, iPad, and HTML5*. Springer.
- Harvey, B., Garcia, D., Paley, J., & Segars, L. (2012). Snap!:(build your own blocks). In *Proceedings of the 43rd ACM technical symposium on Computer Science Education* (p. 662).
- Hemery, E., Manitsaris, S., Moutarde, F., Volioti, C., & Manitsaris, A. (2015). Towards the design of a natural user interface for performing and learning musical gestures. *Procedia Manufacturing*, 3, 6329–6336.
- Ioannidou, A., Repenning, A., & Webb, D. C. (2009). AgentCubes: Incremental 3D end-user development. *Journal of Visual Languages & Computing*, 20(4), 236–251.
- Jain, J., Lund, A., & Wixon, D. (2011). The future of natural user interfaces, 211. <https://doi.org/10.1145/1979742.1979527>
- Kahn, K. (2014). TOONTALK REBORN-Re-implementing and re-conceptualising ToonTalk for the Web. *Constructionism, Vienna, Austria*.
- Koschitz, D., Ramagosa, B., & Rosenbaum, E. (2016). Beetle Blocks: A New Visual Language for Designers and Makers.
- Lieberman, H. (2001). *Your wish is my command: Programming by example*. Morgan Kaufmann.
- Lin, H.-C. K., Wu, C.-H., & Hsueh, Y.-P. (2014). The influence of using affective tutoring system in accounting remedial instruction on learning performance and usability. *Computers in Human Behavior*, 41, 514–522.
- Lin, W., Du, L., & Harris-adamson, C. (2017). Human-Computer Interaction. User Interface Design, Development and Multimodality. *Human-Computer Interaction. User Interface Design, Development and Multimodality*, 10271(December), 584–592. <https://doi.org/10.1007/978-3-319-58071-5>
- Mathewson, J. H. (1999). Visual-spatial thinking: An aspect of science overlooked by educators. *Science Education*, 83(1), 33–54.

- Moore, A. G., Howell, M. J., Stiles, A. W., Herrera, N. S., & McMahan, R. P. (2015). Wedge: A musical interface for building and playing composition-appropriate immersive environments. In *2015 IEEE Symposium on 3D User Interfaces (3DUI)* (pp. 205–206).
- Moriarty, B., Lennon, E., DiCola, F., Buzby, K., Manzella, M., & Hromada, E. (2012). Utilizing depth based sensors and customizable software frameworks for experiential application. *Procedia Computer Science*, *12*, 200–205. <https://doi.org/10.1016/j.procs.2012.09.054>
- Morse, P., Reading, A., Lueg, C., & Kenderdine, S. (2015). TaggerVR: interactive data analytics for geoscience—a novel interface for interactive visual analytics of large geoscientific datasets in cloud repositories. In *2015 Big Data Visual Analytics (BDVA)* (pp. 1–2).
- Nainggolan, F. L., Siregar, B., & Fahmi, F. (2016). Anatomy learning system on human skeleton using Leap Motion Controller. In *2016 3rd International Conference on Computer and Information Sciences (ICCOINS)* (pp. 465–470).
- Noor, A. K., & Aras, R. (2015). Potential of multimodal and multiuser interaction with virtual holography. *Advances in Engineering Software*, *81*, 1–6.
- Norman, D. (2013). *The design of everyday things: Revised and expanded edition*. Basic books.
- Perdana, I. (2014). Teaching elementary school students new method of music performance with Leap Motion. In *2014 International Conference on Virtual Systems & Multimedia (VSMM)* (pp. 273–277).
- Pokress, S. C., & Veiga, J. J. D. (2013). MIT App Inventor: Enabling personal mobile computing. *ArXiv Preprint ArXiv:1310.2830*.
- Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S., & Carey, T. (1994). *Human-Computer Interaction*. Essex, UK, UK: Addison-Wesley Longman Ltd.
- Rautaray, S. S., & Agrawal, A. (2015). Vision based hand gesture recognition for human computer interaction: a survey. *Artificial Intelligence Review*, *43*(1), 1–54.
- Repenning, A., Basawapatna, A., & Escherle, N. (2016). Computational thinking tools. In *2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (pp. 218–222).
- Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., ... others. (2009). Scratch: Programming for all. *Commun. Acm*, *52*(11), 60–67.
- Rittitum, P., Vatanawood, W., & Thongtak, A. (2016). Digital scrum board using leap motion. In *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)* (pp. 1–4).
- Salvadori, A., Licari, D., Mancini, G., Brogni, A., De Mitri, N., & Barone, V. (2014). Graphical interfaces and virtual reality for molecular sciences.

- Santana, P. (2014). Interfaces Naturales de Usuario: La Experiencia de la Universidad de Colima. *Researchgate.Net*, 36–37. Retrieved from http://www.researchgate.net/profile/Pedro_Santana2/publication/261985554_Interfaces_Naturales_de_Usuario_La_Experiencia_de_la_Universidad_de_Colima/links/0046353611c8ce457c000000.pdf
- Sharp, H., Preece, J., & Rogers, Y. (2019). *Interaction design: beyond human-computer interaction*. John Wiley & Sons.
- Shneiderman, B., & Plaisant, C. (2006). Diseño de interfaces de usuario. *Estrategias Para Una Interacción Persona-Computador Efectiva, Cuarta Edición Ed: Pearson Education*.
- Silva, E. S., de Abreu, J. A. O., de Almeida, J. H. P., Teichrieb, V., & Ramalho, G. L. (2013). A preliminary evaluation of the leap motion sensor as controller of new digital musical instruments. *Recife, Brasil*.
- Slany, W. (2014). Tinkering with Pocket Code, a Scratch-like programming app for your smartphone. *Proceedings of Constructionism*.
- Tran, V. T., Lee, J., Kim, D., & Jeong, Y. S. (2016). Easy-to-use virtual brick manipulation techniques using hand gestures. *Journal of Supercomputing*, 72(7), 2752–2766. <https://doi.org/10.1007/s11227-015-1588-4>
- Vatavu, R.-D., Anthony, L., & Wobbrock, J. O. (2012). Gestures as point clouds, 273. <https://doi.org/10.1145/2388676.2388732>
- Volioti, C., Hemery, E., Manitsaris, S., Teskouropoulou, V., Yilmaz, E., Moutarde, F., & Manitsaris, A. (2015). Music gestural skills development engaging teachers, learners and expert performers. *Procedia Manufacturing*, 3, 1543–1550.
- Wenger, E. (2014). *Artificial intelligence and tutoring systems: computational and cognitive approaches to the communication of knowledge*. Morgan Kaufmann.
- Werner, L., Campe, S., & Denner, J. (2012). Children learning computer science concepts via Alice game-programming, 427. <https://doi.org/10.1145/2157136.2157263>
- Wieringa, R., Maiden, N., Mead, N., & Rolland, C. (2006). Requirements engineering paper classification and evaluation criteria: A proposal and a discussion. *Requirements Engineering*, 11(1), 102–107. <https://doi.org/10.1007/s00766-005-0021-6>
- Wigdor, D., & Wixon, D. (2011). *Brave NUI world: designing natural user interfaces for touch and gesture*. Elsevier.
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.
- Zhu, G., Cai, S., Ma, Y., & Liu, E. (2015). A Series of leap motion-based matching games for enhancing the fine motor skills of children with autism. *Proceedings - IEEE 15th International Conference on*

Advanced Learning Technologies: Advanced Technologies for Supporting Open Access to Formal and Informal Learning, ICALT 2015, 430–431. <https://doi.org/10.1109/ICALT.2015.86>