Universidad Nacional de La Plata

Facultad de Informática

TRABAJO FINAL PRESENTADO PARA OBTENER EL GRADO DE
ESPECIALISTA EN TECNOLOGÍA INFORMÁTICA APLICADA EN EDUCACIÓN

Programación tangible en el ámbito educativo. Análisis de experiencias

---

Estudiante

Prof. Leandro Javier Castro

Directora:

Dra. Verónica Artola

Co-Director:

Mg. Prof. Gustavo Javier Astudillo

Diciembre, 2022

Agradecimientos

A Verónica y Gustavo, por cada minuto, videollamada, mensaje, audio, comentario o sugerencia.

A Sofi, mi compañera, por simplemente escuchar cada novedad del trabajo final.

Resumen

Los saberes sobre programación comenzarán a ser parte de la currícula escolar en Argentina. Se espera a partir de las capacitaciones y mesas de trabajo realizadas en algunas provincias, que en el año 2023 en educación inicial, primaria y secundaria se desarrollen conceptos relacionados con la programación. En este contexto, los lenguajes de programación tienen un rol fundamental para abordar los saberes propuestos en los Núcleos de Aprendizajes Prioritarios aceptados en 2018. Existen lenguajes de programación textuales que derivan muchas veces en errores de sintaxis y requieren un esfuerzo extra por quien debe aprender a utilizarlo. Así mismo, también existen en menor medida, lenguajes de programación del tipo icónico (mediante el uso de bloques) como el conocido Scratch o Pilas Bloques. Independientemente de si es a través de escritura o mediante bloques, para poder hacer uso de estas herramientas, es necesario poseer conocimientos mínimos sobre informática, ya que para ello se requieren un conjunto de comandos aprendidos, como palabras claves que se deben ingresar o teclas de función que se deben presionar. Aquí es donde la incorporación de las Interfaces de Usuario Tangibles podrían hacer su aporte al permitir una forma de interacción diferente. En este trabajo se presenta la investigación sobre el estado del arte en el uso de interfaces de programación tangible en el contexto educativo.

Palabras claves: educación; programación tangible; interfaces de usuario tangible, estado del arte.

Índice general

Resumen **4**

Índice general **6**

Introducción **7**

Capítulo 1. Programación, bits y átomos **10**
1.1 Algunos Desarrollos previos 14
1.1.1 AlgoBlocks 14
1.1.2 Tangible Programming Bricks 15
1.1.3 Quetzal 15

Capítulo 2. Metodología **16**
2.1 Preguntas de Investigación 17
2.2 Estrategia de búsqueda 17
2.2.1 Resultados obtenidos 18
2.3 Criterios de inclusión y exclusión 19

Capítulo 3. Resultados **22**
3.1 Propuestas de TUI 23
3.2 Resumen de artículos 59

Capítulo 4. Discusión **66**
4.1 Artículos por países 67
4.2 Artículos por años 68
4.3 TUI vs GUI 69
4.4 Grupo etario destinatario 69
4.5 TUI para la inclusión 70
4.6 Recursos físicos y digitales implementados 70
4.7 Objetivos de las investigaciones 71

Capítulo 5. Conclusiones y trabajo futuro **74**
5.1 Conclusiones 75
5.2 Trabajo futuro 76

Referencias **78**

Introducción

En Argentina, a partir de la Resolución CFE N° 343/18[1] los saberes: educación digital, programación y robótica comenzarán a ser obligatorios en todo el país. En el año 2022 se han desarrollado, en lo que respecta a las provincias de La Pampa, Mendoza y Córdoba, mesas de trabajo[2] e implementación de trayectos[3][4][5] orientados sobre estos saberes para educación inicial y primaria. Teniendo en cuenta la resolución mencionada, se observan distintas situaciones de enseñanza que se esperan promover en las y los estudiantes.

Para Educación Inicial (EI) el entorno del individuo es un elemento esencial en el cual a través de la exploración y observación busca soluciones ante problemas del mundo real mediante materiales concretos y/o digitales. En cuanto a Educación Primaria (EP) - primer y segundo ciclo - la creatividad y el error son parte del proceso en la elaboración de estrategias de resolución. Así mismo se hace un fuerte hincapié en el uso de la robótica como parte de la construcción de conocimientos relacionados con la programación. Para Educación Secundaria (ES) - ciclo básico y orientado - aparece la utilización de entornos de programación textuales e icónicos como herramientas para la elaboración de resoluciones de problemas aplicando estrategias y fragmentación de las soluciones. Al igual que en EP se hace hincapié en la aplicación de la robótica como parte del proceso para la elaboración de soluciones vinculadas con la sociedad (entorno social, económico, ambiental y cultural) (Resolución CFE N. 343/18, 2018).

Se puede observar que recién en los niveles educativos de ES se comienzan a realizar menciones de entornos de programación tanto textuales como icónicos. En EI y EP los saberes sobre programación no se encuentran condicionados por la utilización de un entorno de programación en particular.

Los lenguajes de programación tienen un rol fundamental para abordar algunos de los saberes propuestos. Existen distintos tipos de lenguajes de programación de acuerdo a la expresión de sus instrucciones. Por un lado, los lenguajes de programación textuales que requieren que el programador escriba las instrucciones. En estos lenguajes, el código debe ser escrito siguiendo reglas de sintaxis que derivan muchas

[1] Los Núcleos de Aprendizajes Prioritarios para la Educación Inicial, Primaria y Secundaria fueron elaborados mediante un proceso que incluyó trabajo técnico, consultas regionales, discusiones y acuerdos federales. A partir de esta resolución la educación digital, la programación y la robótica comenzarán a ser obligatorios en todos los establecimientos del país. Disponible en: NAP de Educación Digital, Programación y Robótica.

[2] Mesa de trabajo con coordinadores y coordinadoras de Educación Primaria (La Pampa): https://sitio.lapampa.edu.ar/index.php/educacion-informa-medios/item/mesa-de-trabajo-con-coordinadores-y-coordinadoras-de-educacion-primaria

[3] Comenzó el «Taller de Educación Digital, Programación y Robótica» para directivos y docentes de jardines de infantes (Mendoza): https://www.mendoza.edu.ar/comenzo-taller-de-educacion-digital-programacion-y-robotica-para-directivos-y-docentes-de-jardines-de-infantes/

[4] Docentes se capacitan para enseñar educación digital, programación y robótica (Córdoba): https://www.cordoba.gob.ar/docentes-se-capacitan-para-ensenar-educacion-digital-programacion-y-robotica/

[5] Implementan trayecto innovador de actualización profesional para docentes de Inicial: https://sitio.lapampa.edu.ar/index.php/educacion-informa-medios/item/implementan-trayecto-innovador-de-actualizacion-profesional-para-docentes-de-inicial

veces en errores y requieren un esfuerzo extra por quien debe aprender a utilizarlo. En este punto es importante mencionar que la gran mayoría de los lenguajes textuales - Python[6], C++[7], Java[8], entre otros - no han sido pensados para enseñar conceptos sobre programación. Por otro lado, existen en menor medida, lenguajes de programación del tipo icónico que hacen uso de la biblioteca Blockly[9], como el conocido Scratch[10] o Pilas Bloques[11], en los cuales las instrucciones se expresan a través de representaciones gráficas del estilo de bloques. Independientemente de si es a través de escritura o mediante bloques, el o la estudiante debe poseer conocimientos mínimos sobre informática para poder hacer uso de estas herramientas, ya que para ello se requieren un conjunto de comandos aprendidos, como palabras claves que se deben ingresar o teclas de función que se deben presionar. En este escenario, la incorporación de Interfaces de Usuario Tangibles (TUI por sus siglas en Inglés: *Tangible User Interfaces*) proporcionan diversos beneficios al permitir una forma de interacción diferente (Bern & Horn, 2010).

En este documento se presenta la investigación sobre el estado del arte en el uso de interfaces de programación tangible en el contexto educativo. La motivación de esta investigación radica en describir un estado del arte de la utilización de la programación tangible en el contexto educativo. Para ello se definieron los siguientes objetivos de investigación:

- Identificar las bases teóricas y/o enfoques que sustentan la programación tangible.
- Identificar distintas experiencias que utilizan programación tangible en contextos educativos en los últimos diez años.
- Definir un conjunto de criterios que permitan el análisis de las experiencias seleccionadas.
- Reconocer ventajas y desventajas de la utilización de la programación tangible en contextos educativos.
- Analizar las experiencias educativas que hacen uso de programación tangible.

Esta investigación se organiza como sigue: en el Capítulo 1 se aborda el marco teórico donde se presentan aportes significativos que han dado sustento a las interfaces de usuarios tangibles y a la programación tangible. En el Capítulo 2, se presenta la metodología empleada para llevar adelante la investigación. A continuación, en el Capítulo 3, se exponen los resultados obtenidos. En el Capítulo 4 se expone la discusión

[6] Python es un lenguaje de programación que permite trabajar rápidamente e integrar sistemas de manera más efectiva. Disponible en: https://www.python.org/
[7] C++ es una variante del veterano C. Es un lenguaje orientado a objetos y multiplataforma.
[8] Java es un lenguaje de programación muy versátil utilizado para crear programas de computadoras, App de Android, etc. Disponible en: https://go.java/
[9] Blockly es una biblioteca que agrega un editor de código visual a aplicaciones web y móviles, utilizado en MIT App Inventor, Micro:bit, entre otros. Disponible en: https://developers.google.com/blockly
[10] Con Scratch es posible programar historias interactivas, juegos y animaciones. Disponible en: https://scratch.mit.edu/
[11] Pilas Bloques es un software que está pensado para aprender a programar. Disponible en: https://pilasbloques.program.ar/

en torno a los resultados obtenidos y, finalmente, en el capítulo 5 se presentan las conclusiones alcanzadas y trabajo futuro.

Capítulo 1. Programación, bits y átomos

En este capítulo se recupera la definición de programación y las características que definen a un lenguaje de programación textual o icónico. Se presentan los primeros postulados sobre interfaces de usuario tangibles y luego se exponen algunos desarrollos previos que han servido como sustento para la implementación de nuevas TUI.

Para el abordaje de esta investigación es necesario recuperar y presentar los postulados que definen a la programación y a las interfaces de usuarios tangibles respectivamente.

La Real Academia Española define a la programación como “Acción y efecto de programar.”. Dentro de las ciencias de la computación es posible definir programar como la acción de escribir instrucciones sencillas con el fin de que sean interpretadas y ejecutadas por una computadora u otro dispositivo electrónico. En palabras de Tanenbaum (2000) “Una secuencia de instrucciones que describe cómo realizar cierta tarea se llama programar” (p. 1). Las instrucciones que se utilizan pertenecen a un lenguaje de programación, éste debe ser independiente de la máquina. En este sentido, en la actualidad, existe un amplio abanico de lenguajes de programación. La gran mayoría de estos lenguajes son textuales, es decir, se deben escribir las instrucciones. Existen en menor medida los lenguajes icónicos, caracterizados por poder arrastrar, encastrar y soltar bloques que representan las instrucciones.

Dentro de los lenguajes textuales se pueden distinguir aquellos de alto nivel, estos surgen con la intención de “hacer la programación más cercana al lenguaje natural humano; o dicho de otro modo, aumentar el nivel de abstracción” (Martínez Lopez, Pablo E., p. 29, 2013). La curva de aprendizaje de su sintaxis es relativamente corta dado que se asemeja a un lenguaje natural. En cambio, los lenguajes de bajo nivel conllevan un mayor tiempo de asimilación porque sus instrucciones se acercan más al lenguaje de una computadora.

Otra distinción dentro de los lenguajes textuales son los interpretados y los que se deben compilar para poder ser ejecutados. Recuperando los ejemplos de Tanenbaum (2000), donde L0 es el lenguaje máquina y L1 el lenguaje definido para escribir las instrucciones, en los lenguaje interpretados “la técnica consiste en escribir un programa en L0 que tome programas de L1 como datos de entrada y los ejecute examinando sus instrucciones una por una y ejecutando directamente la sucesión de instrucciones en L0 que equivale a cada una.”. Para los lenguajes que utilizan compilación previa el método consiste en “sustituir primero cada instrucción escrita en L1 por una sucesión equivalente de instrucciones en L0. El programa resultante consiste exclusivamente en instrucciones de L0. Luego, la computadora ejecuta el nuevo programa en L0 en lugar del antiguo programa en L1” (p. 2). En los lenguajes interpretados se da inicio al programa ejecutando las instrucciones definidas, ejecutando instrucción por instrucción y realizando una traducción en el momento a lenguaje máquina (binario) (ver script 1). Al existir un error, el programa no puede continuar su ejecución, por lo que se debe corregir el error y volver a ejecutar. Un lenguaje compilado, por su parte, como su nombre lo indica se debe compilar. Este proceso, que se realiza luego de escribir el código (conjunto de instrucciones), corrobora que las instrucciones estén correctamente escritas, sintáctica y semánticamente, y realiza una traducción hacia el lenguaje máquina. Una vez obtenida una compilación exitosa se puede ejecutar el programa.

Los lenguajes icónicos por su parte, buscan simplificar la definición de instrucciones sencillas para que se realice una determinada tarea (ver figura 1). En este sentido no es necesario aprender un nuevo lenguaje (su sintaxis), como por ejemplo la definición de variables, funciones, etc., sino más bien identificar las acciones que representa cada bloque y poder utilizarlo en consecuencia. Uno de los software más conocidos dentro de esta categoría es Scratch, el cual mediante bloques que representan

acciones o eventos, los cuales se encuentran diferenciados por colores, permite crear programas como juegos o historias, capaces de convertir texto a voz, utilizar la cámara web como un sensor, realizar animaciones, etc.

```
from gpiozero import MotionSensor
import time
import os

pir = MotionSensor(4)
while True:
     if pir.motion.detected:
          os.system("fswebcam -i 0 -d /dev/video0 -r
          /home/pi/Escritorio/Capturas/%d%m%y_%H%M%S.jpg")
     time.sleep(1)
```

Script 1. Instrucciones utilizadas en Python sobre Raspberry Pi para detectar movimiento mediante un sensor y realizar una captura con la cámara. Producción propia.

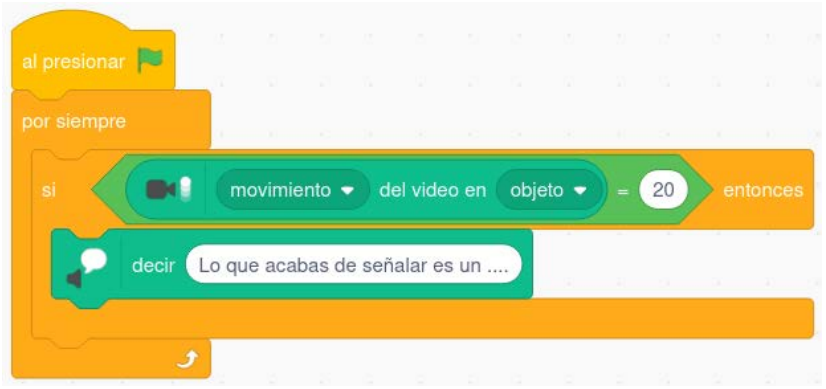

Figura 1. Instrucciones utilizadas en Scratch para identificar un objeto mediante el sensor de video y obtener un feedback a través de audio. Producción propia.

Existe un amplio abanico de posibilidades al momento de elegir un lenguaje de programación. Indistintamente si se trata de un lenguaje textual o icónico, cada uno posee características diferentes que hacen que resulten más adecuados que otros en distintos contextos. Entre estas características, se podría considerar su interfaz de usuario. En este sentido las TUI se presentan como una alternativa a las conocidas Interfaces de Usuario Gráficas (GUI por su siglas en inglés: *Graphical User Interface*).

Un primer acercamiento a las Interfaces de Usuario Tangibles está dado por los autores Ishii & Ullmer (1997). A partir de sus postulados las TUI se podrían pensar como un puente entre el ciberespacio[12] y el medio ambiente. Estas "aumentan el mundo físico real al acoplar información digital a objetos y entornos físicos cotidianos" (Ishii & Ullmer, 1997, p. 234 - 241). Proponen tres conceptos claves para pensar las TUI:

[12] El término ciberespacio fue acuñado por William Gibson en su novela Neuromancer y describe el mundo de los ordenadores y la sociedad creada en base a ellos. Otras definiciones y visiones surgen de los autores Michael Benedikt (desde la arquitectura), Javier Echeverría (desde la filosofía) y Manuel Castells (desde la sociología).

- **Superficies interactivas**, la transformación de superficies en interfaces activas, por ejemplo paredes, escritorios, puertas, etc.
- **Acoplamiento de bits y átomos**, la vinculación de objetos cotidianos manipulables como tarjetas, libros, etc., con información digital.
- **Medio ambiente**, la incorporación de recursos del ambiente como sonido, aire, luz y agua para la percepción del ciberespacio.

A lo largo de los años, con el avance en el estudio de las interfaces tangibles se han encontrado diversos beneficios al relacionarlas con los procesos de enseñanza y aprendizaje (Marshall et al., 2003; Price, 2008, Marshall, Price, y Rogers, 2003; Price, 2008).

La incorporación de las interfaces tangibles en lenguajes de programación ha dado lugar a la programación tangible. La programación tangible permite construir programas organizando y/o conectando objetos físicos que representan elementos y comandos del lenguaje. En palabras de Horn & Jacob (2007) "un lenguaje de programación tangible es similar a un lenguaje de programación visual o basado en texto. Sin embargo, en lugar de usar imágenes y palabras en una pantalla de computadora, los lenguajes tangibles usan objetos físicos para representar varios elementos de programación, comandos y estructuras de flujo de control" (p. 159). Estos lenguajes pueden aprovechar las características físicas de los objetos tangibles (forma, tamaño, color, textura, etc.) para expresar la sintaxis. En este sentido, se encuentran beneficiadas las actividades exploratorias, expresivas y experimentales (Julià et al., 2009). Cabe aclarar que, como mencionan Marshall et al. (2007), para poder realizar estas afirmaciones se necesitan pruebas empíricas y un mayor desarrollo teórico.

Basados en *What is the next generation of human-computer interaction?* (Jacob, 2006), Horn & Jacob (2007) expresan dos principios fundamentales de los lenguajes de programación tangibles:

1. **La interacción tiene lugar en el mundo real**. En los entornos de programación tangible no se programa a través de pantallas de computadoras donde encuentran distractores como juegos, chat en línea, etc. La programación se desarrolla en un entorno más natural, como pueden ser sus bancos de clase o el suelo. Así mismo para el equipo docente esta es una característica que suma en flexibilidad para determinar la estructura y el tiempo de las actividades de programación en clase.
2. **La interacción se comporta más como el mundo real**. Aquí se hace uso de los conocimientos cotidianos del/la estudiante (no informáticos). Por ejemplo, los bloques a utilizar tienen terminaciones similares a las de un rompecabezas que imposibilita construcciones no válidas del lenguaje.

Asimismo, dentro de las TUI, se pueden diferenciar los objetos tangibles activos que brindan una retroalimentación activa y los pasivos que no lo hacen. En palabras de Riedenklau et al. (2010) "la retroalimentación activa se refiere a la capacidad de influir activamente en la interacción, para cambiar la posición u orientación del objeto, o retroalimentación multimodal a través de la modalidad háptica, auditiva o visual, etc." (p. 2). En contrapartida, los objetos pasivos sirven como dispositivos de entrada pero no pueden realizar acciones por sí solos.

Los objetos pasivos "son aquellos objetos que actúan como dispositivos de entrada, pueden ser manipulados por los usuarios y detectados por un sistema, pero no son capaces de realizar ninguna acción por sí mismo" (Alvarado et al., 2020, p. 44). En cambio, los objetos activos permiten obtener un *feedback* que se permiten ser usados de manera independiente o de manera conjunta con otro dispositivo o *software* para enriquecer la interacción tangible a través de pantallas integradas, sensores, etc. (Alvarado et al., 2020).

A continuación se presentan desarrollos previos de lenguajes de programación tangibles.

1.1 Algunos Desarrollos previos

Existen varios desarrollos previos que han servido como sustento y han dado lugar a nuevas propuestas, a modo de ejemplo y con el fin de clarificar se describe aquí **AlgoBlocks**, **Tangible Programming Bricks** y **Quetzal.**

1.1.1 AlgoBlocks

Este entorno fue desarrollado por Suzuki y Kato (1993). El lenguaje está formado por una colección de cubos pasivos de aluminio que entrelazados representan un lenguaje similar a **Logo**[13]. Fue pensado para estudiantes de nivel primario y secundario con el objetivo de crear procedimientos (programas) a través de actividades que promuevan el aprendizaje colaborativo. Cada bloque (cubo) corresponde a un comando. El objetivo consiste en guiar a un submarino que está visible en la pantalla de una computadora. Luego de su implementación y posterior análisis, **AlgoBlocks** demostró estar calificado como un entorno conversacional fomentando el aprendizaje colaborativo entre estudiantes.

Imagen 1. Estudiantes utilizando AlgoBlocks
Imagen extraída de **Project Bloks** (Suzuki & Kato, 1993)

[13] "Logo es una variante del lenguaje de programación LISP, popular entre los investigadores de inteligencia artificial. Es un lenguaje poderoso pero simple para explorar la resolución de problemas lingüísticos y matemáticos" (McNerney, 2004)

1.1.2 Tangible Programming Bricks

Este lenguaje fue creado por Timothy S. McNerney (2004) luego de realizar una investigación sobre informática educativa en el MIT centrada en las interfaces físicas. Esta plataforma permite la creación de micromundos para que las y los estudiantes exploren computación y el pensamiento científico. El sistema hace uso de ladrillos pasivos Legos rediseñados. Los bloques se deben apilar para crear las instrucciones y al mismo tiempo se pueden insertar tarjetas en ellos, por ejemplo para especificar un parámetro numérico. La alimentación de energía la proporciona un bloque base sobre el cual se comienzan a apilar las instrucciones. Cada ladrillo cuenta con conectores en la parte superior e inferior que se utilizan para la alimentación y comunicación.

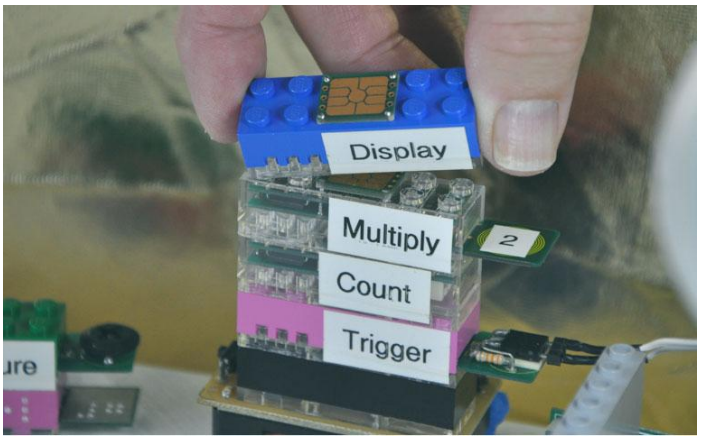

Imagen 2. Ladrillos de programación de Tangible Programming Bricks
Imagen extraída de **Project Bloks** (McNerney, 2004)

1.1.3 Quetzal

Horn y Jacob presentan este lenguaje (2007) con el objetivo de comprender mejor cómo los lenguajes tangibles pueden afectar el aprendizaje de las y los estudiantes en el aula. Se compone de piezas económicas y duraderas sin hacer uso de componentes electrónicos ni fuentes de alimentación. Su diseño está dado por "mosaicos de plástico entrelazados que representan estructuras, acciones y parámetros de flujo de control. Las declaraciones en el lenguaje están conectadas entre sí para formar cadenas de flujo de control. Los programas simples comienzan con una instrucción Begin y terminan con una sola instrucción End" (Horn & Jacob, 2007).

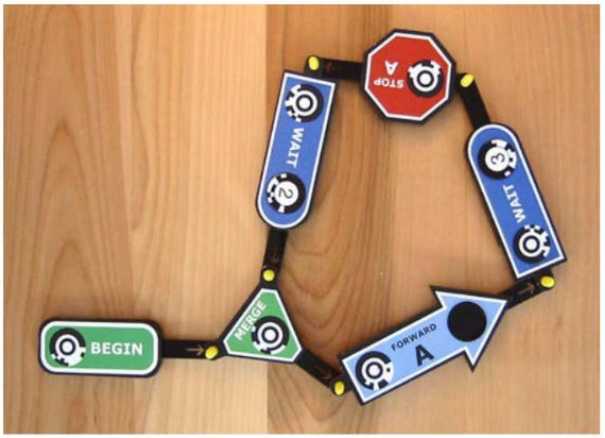

Imagen 3. Una sentencia merge crea un bucle en el programa (Horn & Jacob, 2007, fig. 2)

Capítulo 2. Metodología

En este capítulo se presentan las preguntas de investigación definidas, la estrategia de búsqueda utilizada, los resultados obtenidos y, los criterios de inclusión y exclusión para la selección de los artículos.

Para el desarrollo de este trabajo se realiza una revisión sistemática de literatura. Dicha revisión se basa en la metodología definida por Kitchenham et al. (2009). En esta metodología, la autora plantea: (a) definir preguntas de investigación, (b) trazar una estrategia de búsqueda (dónde buscar, con qué palabras claves), (c) establecer criterios de inclusión y exclusión, que serán aplicados, tanto para la selección inicial, como para la selección final.

2.1 Preguntas de Investigación

P1. ¿Cuáles son las bases teóricas y/o enfoques que sustentan la Programación Tangible (PT)?

P2. ¿Qué experiencias existen de PT en contextos educativos en los últimos diez años?

P3. ¿Cuáles son las características de la PT en contextos educativos?

P4. ¿Qué conjunto de criterios permitirían un análisis de las experiencias con PT?

2.2 Estrategia de búsqueda

La búsqueda se ha realizado en **IEEE Xplore**[14], **ACM Digital Library**[15] y **ScienceDirect**[16]. Los filtros aplicados han sido por palabras claves y artículos publicados en los últimos diez años. Los detalles de la búsqueda realizada en las distintas librerías se presentan en la Tabla 1.

[14] **IEEE Xplore** ofrece acceso a textos completos de literatura técnica en ingeniería y tecnología. Disponible: https://ieeexplore.ieee.org/Xplore/home.jsp

[15] **ACM** proporciona la principal biblioteca digital del campo de la informática y ofrece a sus miembros y a la profesión informática publicaciones, conferencias y recursos profesionales de vanguardia. Disponible en: https://dl.acm.org/

[16] **ScienceDirect** se define así misma como la principal fuente mundial de investigación científica, técnica y médica. Permite la exploración de revistas, libros y artículos. Disponible en: https://www.sciencedirect.com/

Repositorio	Cadena de búsqueda
IEEE Xplore	("All Metadata":"tangible programming" and education) OR ("Author Keywords":tangible and programming and education) Filters Applied: 2011 - 2021
ACM Digital Library	[[All: "tangible programming"] AND [All: education]] OR [[Keywords: tangible] AND [Keywords: programming] AND [Keywords: education]] AND [Publication Date: (01/01/2011 TO 12/31/2021)]
ScienceDirect	(Find articles with these terms: ("tangible programming") AND (education)) AND Year(s): 2011-2021

Tabla 1. Repositorios consultados y cadenas de búsqueda utilizadas.

2.2.1 Resultados obtenidos

Los resultados que se obtuvieron fueron los siguientes:

Repositorio	Cantidad de resultados obtenidos
IEEE Xplore	19
ACM Digital Library	135
ScienceDirect	35
Total	**189**

Tabla 2. Cantidad de resultados obtenidos por repositorio

2.3 Criterios de inclusión y exclusión

A partir de estos resultados obtenidos en la etapa anterior se ha realizado una primera lectura de los resúmenes de los artículos. En base a ello se han descartado un total de 78 artículos. La supresión de estos radica en que su contenido no aporta información significativa a los objetivos propuestos (ver Tabla 3).

Criterios de exclusión	**Cantidad de artículos**
Otra temática	43
No se implementa lenguaje de programación tangible	25
No está pensado para enseñar programación	9
Doble publicación	1
Total	**78**

Tabla 3. Cantidad de artículos descartados luego de leer los resúmenes

Luego de realizar la primera lectura de los resúmenes se obtiene un total de 111 artículos para ser examinados. Para estos documentos se ha realizado una lectura y análisis que ha derivado en la exclusión de 65 de ellos. Los motivos fueron:

- Doble publicación. Artículos con contenido similar, mismos autores y autoras. Se seleccionó el más actual.
- Artículos que presentan una propuesta de interfaz de usuario tangible pero no está pensada para enseñar a programar.
- No se implementa un lenguaje de programación tangible.
- La temática abordada no está relacionada con la programación tangible.
- No aporta información significativa a la temática.

En base a la lectura y exclusión de artículos se consideran 46 documentos. Durante la lectura se ha considerado para el análisis:

- País donde se ha elaborado la publicación.
- Objetivos de la investigación.
- Características de los destinatarios: rango etario, si poseen alguna discapacidad y experiencia previa en programación.
- Experiencias con el grupo destinatario.

- TUI
    - Implementación, costos y comandos disponibles
    - Recursos físicos: bloques, pantalla, robot, parlantes, luces, etc.
    - Recursos digitales: animaciones, audios, etc.

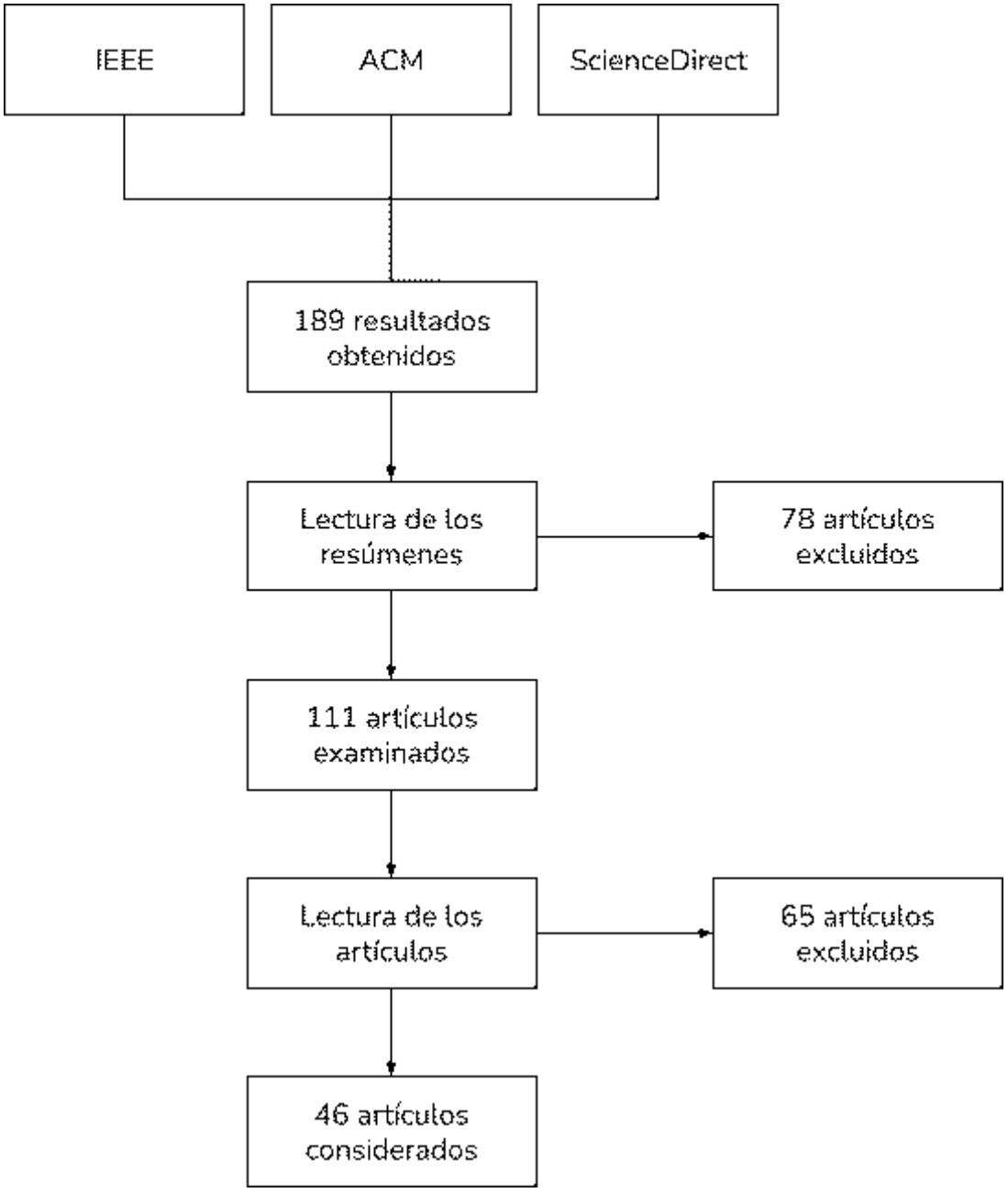

Figura 2. Flujo de recopilación, lectura y exclusión de los artículos

Capítulo 3. Resultados

En este capítulo se presentan los resultados obtenidos. Se describe cada uno de los artículos seleccionados y luego se presenta una tabla que resume las características principales de cada documento.

A partir de los artículos seleccionados, fue posible analizar un conjunto de propuestas que hacen un uso didáctico de la programación tangible. Algunos de los resultados obtenidos describen propuestas de interacción tangible, mientras que otros están centrados en la comparación de interfaces de programación tangibles con interfaces gráficas (TUI vs. GUI). Un grupo más pequeño de artículos presentan propuestas de programación tangible, pero con el agregado de que la interfaz está pensada para personas con disminución visual.

3.1 Propuestas de TUI

- **Sapounidis11** (Sapounidis & Demetriadis, 2011). Se presentan dos lenguajes de programación tangible: T_Butterfly y T_ProRob. Ambos fueron pensados para que personas novatas puedan asimilar -de una manera accesible- conceptos de programación. T_Butterfly está compuesto por cubos, una caja maestra y una computadora. El desafío consiste en guiar a una mariposa hacia una flor. La mariposa se muestra en pantalla y siempre parte desde su casa. Los cubos se conectan a la caja maestra y luego entre sí. Una vez realizado el programa se presiona un botón en la caja maestra la cual se encarga de enviar las instrucciones a la computadora para poder plasmarlas en pantalla (ver Imagen 4). Cabe aclarar que el diseño de los cubos permite ir visualizando el recorrido que realizará la mariposa y al mismo tiempo no permite conexiones erróneas entre cubos. Algunos de los comandos disponibles son avanzar, girar, avanzar dos veces, etc. Así mismo está permitido el uso de parámetros y en estos se aprovechan características físicas de los componentes dado que un parámetro con valor 4 tiene un peso mayor que un parámetro con valor 2. De manera similar, T_ProRob está compuesto por cubos, una caja maestra, una computadora y un robot. El desafío consiste en programar un robot Lego para que realice diferentes acciones. Luego mediante bluetooth el ordenador se encarga de enviar las instrucciones al robot. Para este lenguaje de programación se dispone de 28 comandos y 16 parámetros más pequeños: avanzar un paso, retroceder, girar a la derecha o a la izquierda. Además, los comandos `FOR` e `IF` están disponibles y admiten al mismo tiempo combinaciones más complejas, como bucles anidados. También existe un cubo en el cual es posible guardar un conjunto de instrucciones y utilizarlo más tarde como parte de la resolución. Quien se encarga de programar el robot tiene una retroalimentación instantánea en la utilización incorrecta de un parámetro sobre un comando a través de un led. Si el parámetro utilizado no corresponde al comando, el programa no se ejecuta y un led integrado en el propio parámetro se lo indica. El grupo de autores concluye que ambos lenguajes pueden ser utilizados por estudiantes de corta edad. Entre ellos pueden ser complementarios para conseguir un objetivo en común que facilite el acceso a los conceptos de programación.

Imagen 4. T_Butterfly: caja maestra y cubos utilizados para programar de manera tangible una mariposa virtual. Los comandos empleados: avanzar, girar a la izquierda, avanzar y avanzar dos veces (Sapounidis & Demetriadis, 2011, fig. 1)

- **Wang11** (Wang et al., 2011). En este artículo se presenta T-maze, un entorno de programación simple. El sistema se compone de sensores, una cámara, bloques de madera y un laberinto virtual. Los bloques de programación disponibles son inicio, fin, dirección, bucle y sensor. Todos cuentan con imanes en dos superficies opuestas y de esta manera generan una restricción semántica. Esta TUI permite la creación del laberinto que se debe resolver a través de los propios bloques. Las instrucciones son interpretadas por una cámara para posteriormente ejecutarlas en el personaje que se encuentra en un laberinto. Se realizó un estudio preliminar con diez estudiantes de cinco a nueve años de edad. Gran parte del grupo de estudiantes tenía experiencia previa en el uso de computadoras. El estudio se dividió en cuatro etapas: explicación y demostración, práctica, prueba y entrevista. T-maze demostró ser una herramienta eficiente, el grupo de estudiantes pudo utilizarla sin dificultades. Se destaca que puede fomentar el aprendizaje lógico y de resolución de problemas a través del juego.

Imagen 5. Estudiante utilizando T-maze (Wang et al., 2011, fig. 1)

- **Kwon12** (Kwon et al., 2012) Este trabajo presenta un estudio sobre una herramienta de programación tangible desarrollada por los autores llamada Algorithmic Bricks (A-Bricks). A-bricks está compuesto por un robot que tiene dos ruedas que funcionan con motores paso a paso y un sensor led que permite detectar una línea negra. El robot puede avanzar, girar hacia la izquierda/derecha, entre otras funciones. Su programación se realiza a través de ladrillos que pueden ser conectados horizontalmente o apilados entre sí. Cuenta con ladrillos que permiten indicar el inicio/final de una repetición. Se realizó un estudio que involucró a estudiantes de primaria de primer grado que fueron divididos en dos grupos: experimental y de control. El grupo experimental recibió A-bricks (tangible) y el grupo de control recibió Scratch (gráfica) como herramientas de programación. A partir de las experiencias realizadas se obtuvo que A-bricks resultó ser eficiente para permitir que las y los estudiantes puedan manifestar habilidades de pensamiento lógico y resolver problemas por su cuenta. Además, se observó que A-bricks es una herramienta eficaz para las primeras etapas del aprendizaje de la programación. Se destaca que la versión TUI fomenta en los alumnos la exploración de soluciones de manera independiente, utilizando la intuición.

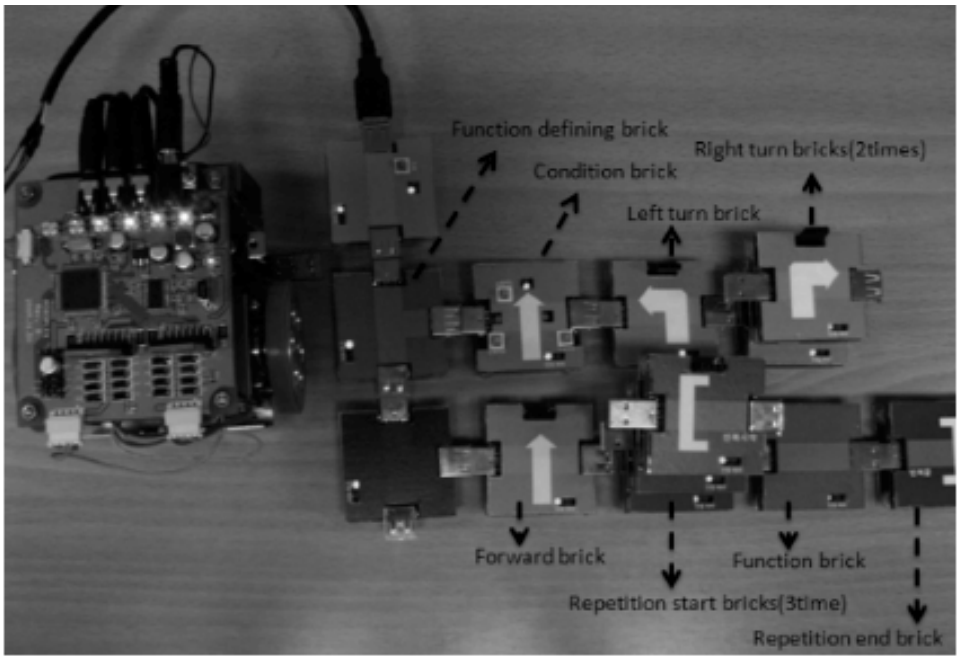

Imagen 6. Ejemplo de una secuencia de programación creada con A-Bricks. A la izquierda se observa el robot que ejecuta la secuencia (Kwon et al., 2012, fig. 8)

- **Scharf12** (Scharf et al., 2012). En este artículo se presenta Tangicoins 3.0, un juego que fomenta la construcción algorítmica y el razonamiento. El sistema está pensado para estudiantes de seis a nueve años de edad. El diseño hace un fuerte hincapié en generar un aprendizaje colaborativo entre el grupo de estudiantes. En este sentido se pregona por un juego no competitivo, donde se espera que se genere cooperación entre las y los estudiantes. El desafío del juego consiste en mover un personaje por un laberinto. Para definir las instrucciones se deben utilizar una serie de bloques, se distribuyen de manera uniforme de tal manera que si alguien no participa el desafío el mismo no se puede completar. Los cubos interactúan de manera inalámbrica entre sí y estos con la computadora que es donde se visualizan las acciones del avatar. Así mismo cada bloque cuenta con una pequeña pantalla gráfica que muestra animaciones e imágenes representativas del propio bloque y, dentro de ellos se encuentran acelerómetros y transceptores que permiten saber si han sido movidos o identificar qué bloque se encuentra a su lado. Se realizaron evaluaciones con 20 estudiantes de seis a nueve años de edad. Se observó que durante el juego se naturaliza el uso de funciones, parámetros y secuencias. Así mismo, para estudiantes de nueve años de edad el juego resultaba en corto plazo aburrido por no presentar problemas que los desafíen.

Imagen 7. Ejecutando la secuencia con Tangicoins 3.0 (Scharf et al., 2012, fig. 9)

- **Sipitakiat12** (Sipitakiat & Nusen, 2012). En este artículo se presenta Robo-Blocks, un sistema pensado para realizar un análisis de las capacidades que puede obtener un grupo de estudiantes a través de la depuración. Robo-Blocks está compuesto por un robot, bloques de programación y un bloque maestro. A través de los bloques de programación se pueden definir los movimientos del robot, los cuales son avanzar, retroceder y girar a la izquierda o derecha. Cada bloque tiene una perilla que permite definir la intensidad de cada giro o los pasos que debe avanzar, el valor se visualiza en una pantalla incorporada en cada bloque. Para realizar la programación se deben anexar al bloque maestro las instrucciones deseadas en forma de cadena. Luego las instrucciones son interpretadas y posteriormente enviadas al robot para que las ejecute. Se realizó un estudio con 52 estudiantes de ocho a nueve años. Se presentaron actividades donde el robot debía moverse por un laberinto y dibujar figuras geométricas. Se observó que, durante la ejecución de las instrucciones propuestas, las y los estudiantes identificaban los errores y se construía críticamente sobre ellos, no era necesario esperar al final de la ejecución para identificar los errores.

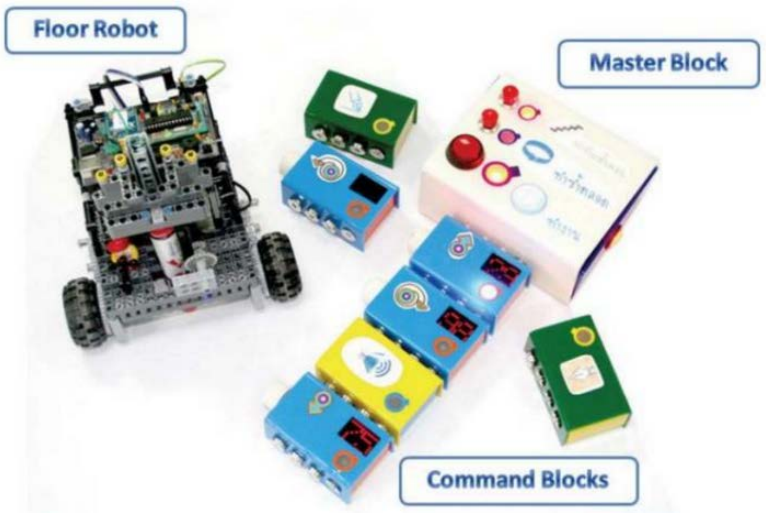

Imagen 8. El sistema Robo-Blocks (Sipitakiat & Nusen, 2012, fig. 1)

- **Oh13** (Oh et al., 2013). En este documento se presenta Digital Dream Lab, un prototipo de bloques de rompecabezas que expone a las y los estudiantes a conceptos de programación simples para desarrollar habilidades básicas de

lógica y pensamiento computacional. El sistema está pensado para estudiantes de cuatro a cinco años de edad, con habilidades limitadas en escritura y lectura. Los bloques de rompecabezas se colocan sobre una mesa, la cual en la parte inferior tiene una cámara que se encarga de identificar los bloques. Se utiliza un proyector para mostrar en una pared toda la información relacionada con la herramienta. El sistema se compone por distintos tipos de bloques: personajes (pez, rana, tortuga, etc.), animación (caminar, saltar y saludar), color, tamaño, variable y fondos. El grupo de investigación concluye que Digital Dream Lab sirve como soporte para futuras investigaciones y diseños de TUI.

Imagen 9. Del lado izquierdo estudiantes utilizando Digital Dream Lab y del lado derecho los componentes de la herramienta (Oh et al., 2013, fig. 1)

- **Chawla13** (Chawla et al., 2013). En este artículo se presenta Dr. Wagon, una herramienta pensada para la comprensión de conceptos formales y abstractos de programación. El sistema está pensado para estudiantes de seis a doce años de edad y se inspira en Robo-Blocks (**Sipitakiat12**) haciendo uso de un robot, una caja maestra y bloques de programación. Se incorporan algunas mejoras como el uso de bucle o condicionales en su programación. Además, hace uso de sensores de color y distancia. Se realizaron pruebas informales sobre el sistema. Luego de las evaluaciones se cree que el sistema puede ser un juguete atractivo que tiene la capacidad de alentar a estudiantes a explorar conceptos introductorios de programación.

Imagen 10. El sistema Dr. Wagon

- **Strawhacker13** (Strawhacker et al., 2013). En este documento se presenta CHERP, un sistema gráfico y tangible pensado para programar un robot. Se compara el aprendizaje de las y los estudiantes utilizando tres interfaces de programación: tangible, virtual y tangible-virtual. El robot se puede programar mediante bloques de madera o por una interfaz virtual mediante íconos representativos. Si se realiza la programación mediante bloques una cámara captura las instrucciones y las convierte en código para que el robot las pueda ejecutar. La investigación se realizó con 36 estudiantes de cinco a seis años de edad durante 13 sesiones de una hora. Se establecieron grupos de dos a tres integrantes para completar las diferentes actividades de programación asignadas a lo largo de las sesiones. Luego de la instancia de evaluación se observó que los grupos que utilizaron la interfaz tangible superaron a la gráfica y la híbrida en la resolución mediante secuenciación. Por otro lado, el grupo de la interfaz gráfica superó a las otras interfaces en el uso de bucles. Los grupos que utilizaron una interfaz híbrida se encontraron muy por debajo en comparación con el resto. Queda como trabajo futuro seguir recopilando más evidencias que permitan identificar qué interfaz es mejor en determinada ocasión o para un grupo determinado.

Imagen 11. Un robot de muestra con interfaces gráficas y tangibles (Strawhacker et al., 2013, fig. 1)

- **Zuckerman13** (Zuckerman & Gal-Oz, 2013). En este artículo se realiza una comparación entre una interfaz tangible y gráfica de un sistema llamado FlowBlocks. Este sistema hace uso de bloques de maderas que se conectan entre sí permitiendo generar un flujo de datos. Con el fin de generar versiones similares crearon una GUI de FlowBlocks para asegurar que ambas interfaces sean lo más similares posibles. Para dicha investigación participaron 58 estudiantes universitarios con una edad de 19 a 31 años. Fueron asignados aleatoriamente a cada interfaz y se les solicitó que utilizaran libremente el sistema durante 30 minutos. La decisión de una tarea no estructurada está dada por un lado en el propósito del propio sistema, el cual es apoyar el aprendizaje no estructurado y, por el otro, realizar una investigación distinta a las previas, es decir, en estudios preliminares que se han realizado comparaciones siempre a partir de actividades estructuradas, donde, en algunos casos, no se encuentran

diferencias entre las interfaces. Luego de la investigación se obtuvo que gran parte del grupo de estudiantes prefieren la TUI por sobre la GUI, independientemente del género o con qué interfaz interactúan en primera instancia. Los motivos estuvieron dados porque permite una interacción física, por proporcionar una retroalimentación agradable y producir un alto nivel de realismo.

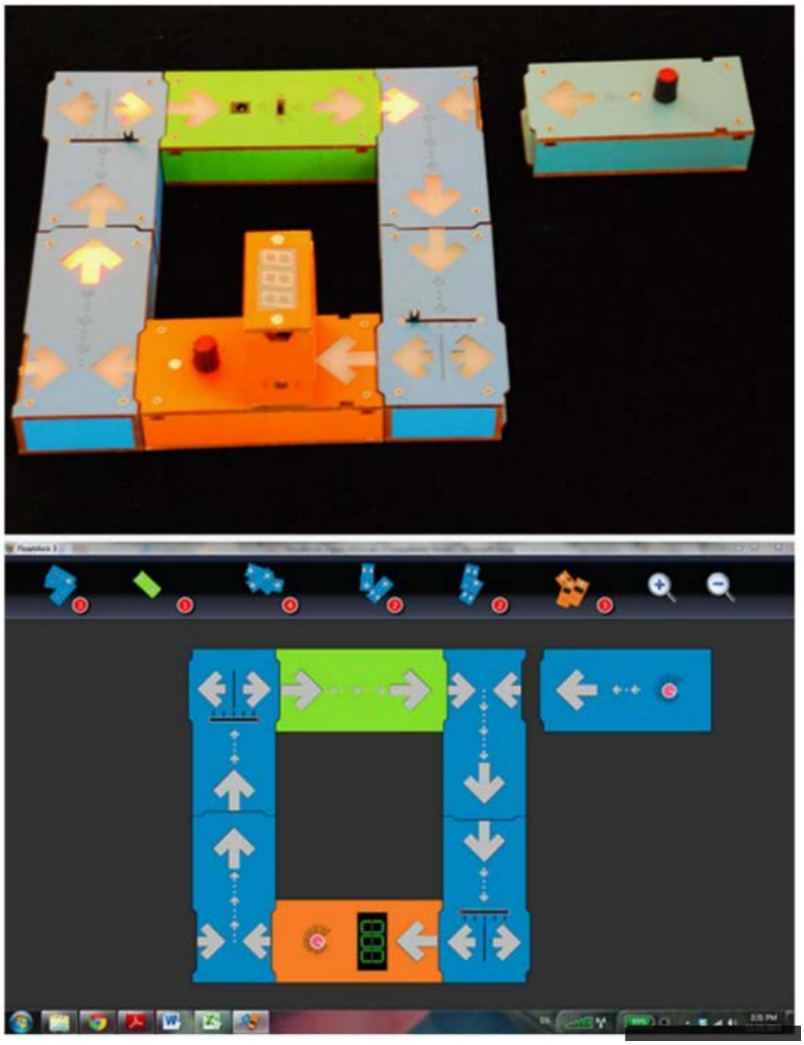

Imagen 12. Versión TUI de FlowBlocks (figura superior) y versión GUI (figura inferior). El modelo es un modelo de ciclo, que simula el flujo y la acumulación en un modelo de "bucle cerrado" (Zuckerman & Gal-Oz, 2013, fig. 2)

- **Kakehashi14** (Kakehashi et al., 2014). Se presenta P-CUBE que está pensado para que personas con discapacidad visual puedan contar con una herramienta en el inicio de la programación. P-CUBE está compuesto por un robot móvil, una caja de programación, bloques de programación y una computadora. Los bloques que se deben colocar dentro de la caja se identifican por etiquetas de radiofrecuencia, están pensados para ser reconocidos a través de información táctil. Hay cuatro tipos de bloques: movimiento, temporizador, condicional y repetición. Así mismo el robot puede avanzar o retroceder y emite sonidos diferentes cuando gira hacia la derecha o izquierda. El programa es transmitido hacia la computadora y posteriormente se transmite al robot. P-CUBE se puso a prueba en un taller para personas con discapacidad visual de nivel secundario y

bachillerato. En el grupo, tres de los integrantes tenían disminución visual grave y uno de ellos era ciego. Además, ninguno tenía conocimientos previos sobre programación. Quienes participaron del taller pudieron completar las tareas propuestas: realizar un programa secuencial y un rastreador de línea. El tiempo implementado para desarrollar las tareas fue de 90 min. A partir de esta experiencia se han realizado mejoras en los cubos dado que algunos eran difíciles de identificar y además se modificó la transmisión del programa para permitir que se ejecute en el robot directamente, ya que quienes participaron no querían depender de una persona para completar la ejecución de los programas. El entorno permite una mayor atención al momento de desarrollar los algoritmos al no tener que realizar operaciones sobre una computadora que les pueda ocasionar problemas. Así mismo se cree que será fácil de usar por personas principiantes en programación.

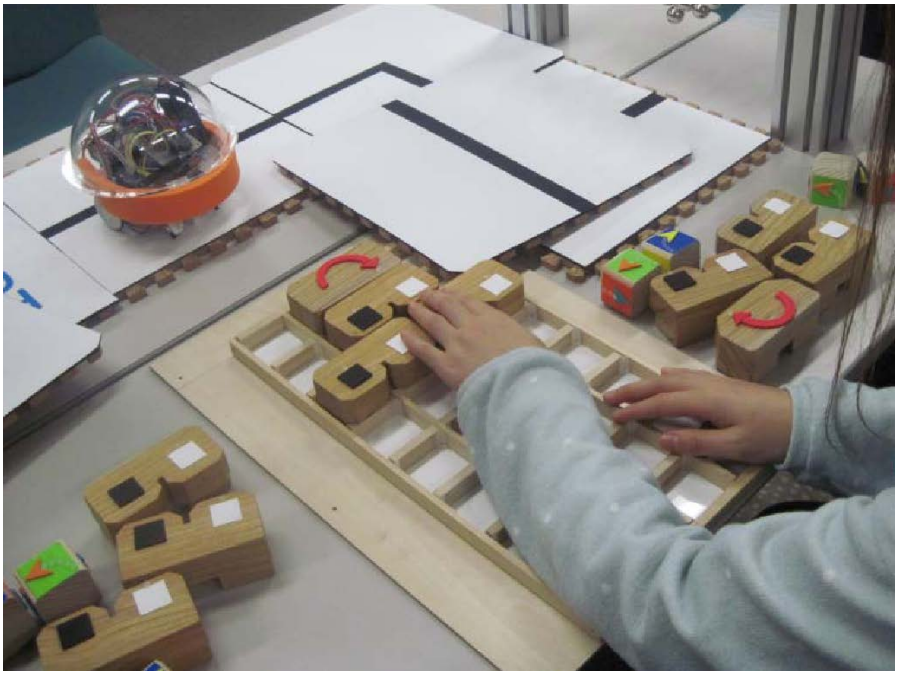

Imagen 13. Robot móvil, caja de programación y bloques de programación (Kakehashi et al., 2014, fig. 7)

- **Wang15** (Wang et al., 2015). En este artículo se presenta TanPro-Kit 2.0, un sistema pensado para estudiantes de seis a ocho años de edad. La herramienta está compuesta por un pad led y bloques de programación. Los bloques se utilizan para definir las instrucciones que serán utilizadas para navegar un laberinto. Dichas instrucciones son reflejadas en el pad. A medida que se van definiendo los bloques los mismos se ven reflejado en tiempo real en el pad, una vez definida las instrucciones se procede a la ejecución. Se realizó una prueba piloto con 15 estudiantes de seis a ocho años de edad. Se realizaron cuatro tareas enfocadas en el uso de parámetros y lógica. Luego de la evaluación se concluye que la herramienta puede ser utilizada sin dificultades por el grupo de estudiantes, donde además propicia el interés por nuevos conceptos, por ejemplo, condicionales.

Imagen 14. Estudiante utilizando TanPro-Kit 2.0 (Wang et al., 2015, fig. 1)

- **Qi15** (Qi et al., 2015). En este artículo se presenta TanProStory, una TUI pensada para transmitir conceptos de programación orientada a objetos. El sistema se utiliza en tres etapas. La primera consiste en la creación del personaje, la segunda en la programación (acciones del personaje) y la tercera se compone por la ejecución de las instrucciones definidas en la etapa anterior. Así mismo el programa está compuesto por tres partes, las cuales son una computadora donde se ejecutan las acciones, los bloques de programación tangible donde se establecen las instrucciones y un panel de sensores. Los bloques de programación sirven para crear y definir las acciones del personaje. Se realizó un estudio preliminar con 11 estudiantes de seis a nueve años de edad. Se les presentó la TUI indicando cómo utilizarla para la creación de un personaje, establecer las acciones y posteriormente realizar la ejecución. Luego se les solicitó que contaran una historia haciendo uso de la herramienta. Se observó que pudieron crear e inicializar personajes y acciones sin dificultades. Se espera realizar como trabajo futuro mejoras en la interfaz como así también comparar su uso frente a una GUI.

Imagen 15. Estudiante utilizando TranProStory (Qi et al., 2015, fig. 1)

- **Hu15** (Hu et al., 2015). En este artículo se presenta Strawbies, un juego de programación tangible pensado para estudiantes de cinco a diez años de edad. El objetivo del juego consiste en guiar a un personaje virtual en la recolección de fresas. El sistema hace uso de un iPad, visión por computadora y fichas de programación de madera. Las y los estudiantes deben definir la secuencia de eventos que realizará el personaje mediante las fichas para ser detectadas mediante la cámara y así ejecutar las instrucciones en el entorno virtual. El sistema se probó en dos colegios con estudiantes de ocho a doce años de edad. Observaron colaboración y negociaciones al momento de definir las acciones que debía realizar el personaje. Las y los estudiantes más pequeños no utilizaban gran cantidad de fichas pero se estima que en su mente sí tenían armada una estructura secuencial de las acciones que debía realizar el personaje.

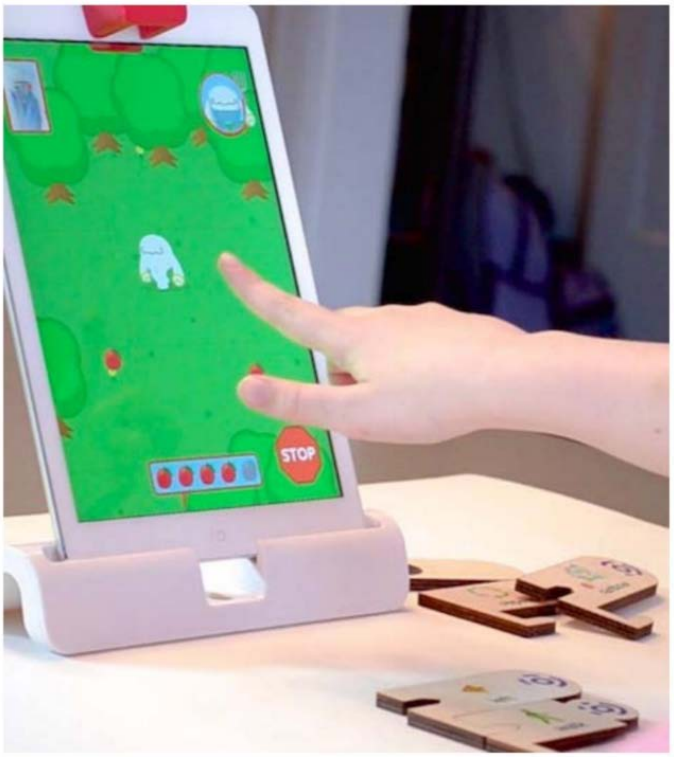

Imagen 16. Un estudiante crea un programa con Strawbies utilizando mosaicos tangibles y un iPad (Hu et al., 2015, fig. 1)

- **Sullivan15** (Sullivan et al., 2015). En este artículo se presenta KIBO, un sistema pensado para adquirir nociones básicas de ingeniería y programación. Se espera que con esta herramienta las y los estudiantes (de cuatro a siete años de edad) puedan experimentar con motores y actuadores y, al mismo tiempo, aprender conceptos de programación a partir del uso de secuencias, bucles y variables. Este kit está compuesto por un robot y bloques de programación tangibles. Los bloques son de madera y no contienen ningún componente electrónico. Se diferencian por pegatinas de distintos colores y distintos tipos de símbolos. También hacen uso de códigos de barras que son escaneados por el robot para recibir las instrucciones. El robot ha ido mutando de versión en versión gracias a la retroalimentación recibida en distintas pruebas piloto que se han realizado en diferentes eventos (colegios, campamentos, laboratorios, etc.). La investigación expone que las y los estudiantes pueden aprender conceptos básicos de

programación e ingeniería a una edad temprana. Este tipo de kit colabora en el aprendizaje de una manera lúdica.

Imagen 17. Robot KIBO, plataforma de arte y tres bloques de programación (Sullivan et al., 2015, fig. 1)

- **Tada15** (Tada & Tanaka, 2015). En este documento se presenta Hojas, un entorno de programación tangible que utiliza tarjetas de papel para aprender a programar. El sistema está pensado para ser utilizado por estudiantes de secundario o en los primeros años de universidad. Para poder hacer uso de este sistema se necesita de una cámara web que interprete las instrucciones (tarjetas) para convertirlas en código y ejecutar los comandos. Las tarjetas deben ser colocadas en el orden deseado de manera horizontal, luego de interpretar las tarjetas en pantalla se visualiza el dibujo resultante con sus movimientos. Las tarjetas permiten dibujar y mover formas, utilizar sonidos, como así también utilizar bucles. Se realizó una evaluación preliminar del sistema con ocho estudiantes de entre 18 y 22 años de edad, donde había integrantes que ya sabían programar o con nociones básicas de programación, solo una persona no tenía conocimientos previos sobre cómo programar. La herramienta mostró buenos niveles de usabilidad entre los usuarios que la utilizaron.

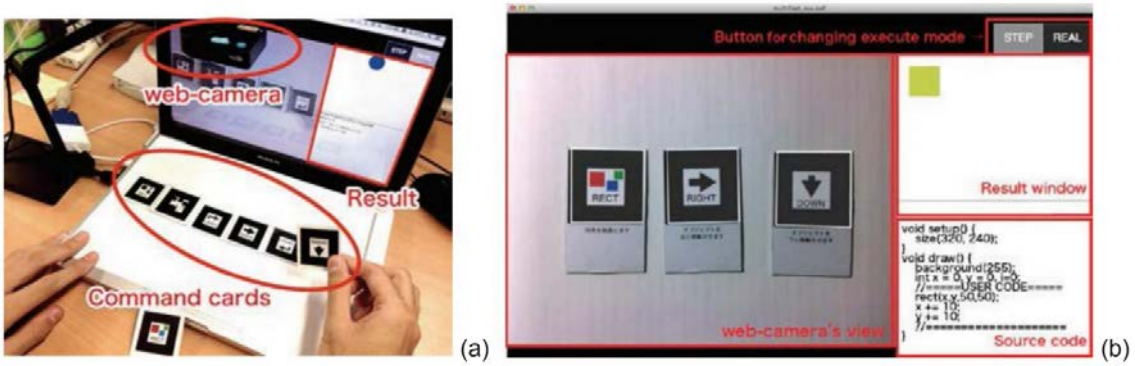

Imagen 18. (a) Estructura del sistema; (b) Ilustraciń general del entorno (Tada & Tanaka, 2015, fig. 2)

- **Zhu16** (Zhu et al., 2016). En este documento se realiza una comparación en el uso de una TUI y una GUI con el fin de promover el pensamiento

computacional. Se crearon a través de bloques legos comandos similares a los virtuales. Luego mediante una cámara web se realiza el mapeo para que posteriormente un software lo convierta en código ejecutable. Tanto la GUI como la TUI tenían dos tipos de salidas, virtual (animación en pantalla) y tangible (robot). El robot recibía la señal mediante bluetooth. El estudio se realizó con 12 estudiantes de siete a diez años de edad en un periodo de tres sesiones de 45 minutos aproximadamente. El estudio realizado sugiere que GUI distrae menos que la TUI, aunque generan menos argumentos en sus soluciones.

Imagen 19. De izquierda a derecha: entrada gráfica, salida gráfica, entrada tangible, salida tangible (Zhu et al., 2016, fig. 1)

- **Motoyoshi16** (Motoyoshi et al., 2016). En este artículo se presentan dos lenguajes de programación tangibles. Uno de ellos es P-CUBE el cual ya fue abordado en **Kakehashi14**. De aquí en adelante se describe Pro-Tan. Este lenguaje guarda algunas similitudes con P-CUBE dado que también hace uso de tarjetas RFID, aunque este es un sistema más liviano y pequeño al utilizar poliestireno tanto para la base como para las tarjetas que son utilizadas para realizar la programación. Esta misma característica le hace perder la capacidad de información táctil, por lo que no es utilizable por personas con disminución visual como si lo es P-CUBE. Para cada tarjeta de acción se debe configurar una tarjeta de temporización indicando así la duración de la instrucción. Luego de colocar las tarjetas sobre la base se debe presionar un botón para enviar las instrucciones al robot mediante Wifi. Seguidamente se debe presionar otro botón en el robot para ejecutar las instrucciones. Queda como trabajo futuro probar esta herramienta con estudiantes.

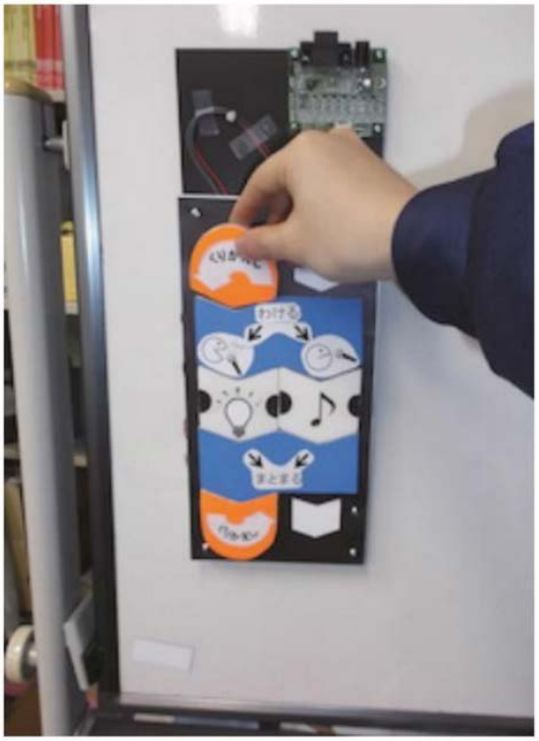

Imagen 20. Panel de programación de Pro-Tan (Motoyoshi et al., 2016, fig. 9)

- **Wang16** (Wang et al., 2016). En este documento se presenta el rediseño de TanProRobot. La versión 2.0 se compone de bloques de programación, un robot (auto) y manipuladores (elementos que se utilizan para decorar la escena). Los bloques se dividen en tres categorías: sensor, acción y potencia. Se realizó un estudio exploratorio con 11 estudiantes de 6 a 8 años de edad. Se les presentó el sistema al grupo de estudiantes con una demostración de uso. Luego se les asignaron tareas para realizar. TanProRobot 2.0 resultó ser un sistema de programación interesante para el grupo de estudiantes, fácil de aprender y utilizar. Queda planteado como trabajo futuro realizar mejoras a los bloques de programación, como así también implementar comparaciones con un lenguaje que haga uso de una interfaz gráfica.

Imagen 21. Vista general de TanProRobot 2.0 (Wang et al., 2016, fig. 2)

- **Goyal16** (Goyal et al., 2016). En este artículo se presenta Bits de código, un prototipo pensado para que estudiantes aprendan habilidades básicas del pensamiento computacional. Este sistema se puede escalar y personalizar para diferentes grupos por edades. El lenguaje hace uso de tarjetas de papel, un tablero de actividades y una app móvil. Las tarjetas son bloques de papel que el grupo de estudiantes utilizan para crear la secuencia del programa. Se conectan entre sí en una superficie plana y se diferencian por colores según los comandos o funciones. Existen diferentes tableros de actividades, los cuales van variando para incrementar la dificultad de los desafíos. El objetivo consiste en movilizar a una tortuga por un laberinto para alcanzar la comida. La secuencia de comando se escanea mediante una aplicación la cuál va registrando cada uno de los comandos y luego visualiza las acciones de la tortuga (utilizando realidad aumentada) cuando se enfoca la cámara del smartphone al tablero. Queda como trabajo futuro seguir agregando más funcionalidades al sistema, como por ejemplo la utilización de bucles y, probar el sistema con un grupo de estudiantes.

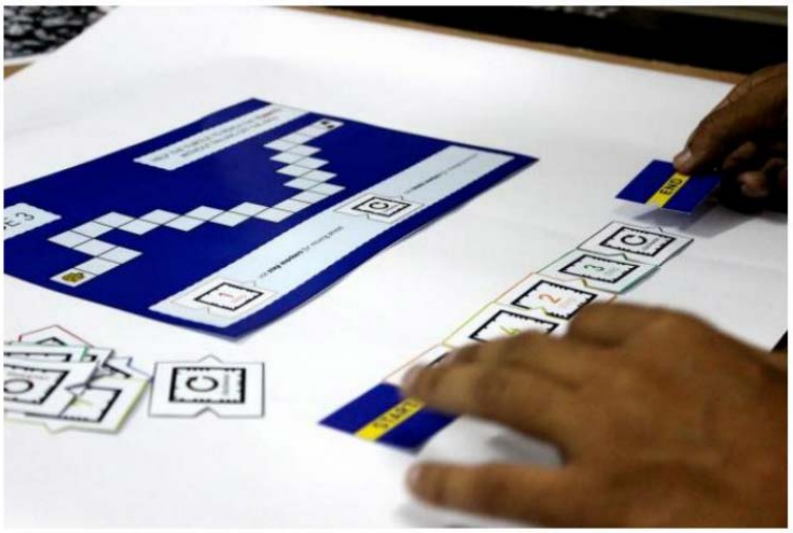

Imagen 22. Una persona que crea un programa usando bits de papel (Goyal et al., 2016, fig. 1)

- **Soleimani16** (Soleimani et al., 2016). En este artículo se presenta CyberPLAYce, un sistema pensado para experimentar y mejorar el pensamiento computacional-espacial a través de la narración de historias. Dicho sistema está compuesto por una placa Arduino y módulos magnéticos de luz, temperatura, sonido, distancia y led. Los módulos deben ser colocados en paneles triangulares que permiten una conexión libre entre sí. El sistema se puso a prueba con 12 estudiantes de 8 a 12 años de edad. Se concluye que puede proporcionar a las y los estudiantes una herramienta para resolver problemas, tomar decisiones y analizar tareas a través de la interacción con problemas del mundo real de una manera abstracta.

Imagen 23. Niños y niñas participando en actividades ciberfísicas proporcionadas por CyberPLAYce (Soleimani et al., 2016, fig. 14)

- **Melcer17** (Melcer & Isbister, 2017). En este documento se presenta un juego de programación llamado Bots & (Main)Frames. El objetivo consiste en programar un robot (virtual) para que llegue a un determinado punto a través de un laberinto. El robot puede moverse hacia adelante y girar 90 grados a la izquierda-derecha. También es posible hacer uso de bucle y funciones, aunque estos se van poniendo a disposición a medida que avanzan de nivel. Los bloques son de madera y cuentan con un gancho de un lado y un ojo en el lado opuesto para encadenarlos y armar la secuencia. Para identificar los bloques se utiliza ReacTIVision. Se probó el sistema con un total de 36 participantes universitarios con un conocimiento no superior a seis meses en programación. Se crearon parejas para observar las interacciones colaborativas entre las y los participantes. Se observó que necesitaron un tiempo sustancial para aplicar las soluciones a los desafíos planteados. Además, se descubrió que los elementos tangibles pueden proporcionar más interacción y comunicación. En el artículo se plantea que para poder realizar conclusiones más precisas es necesario probar el sistemas con un mayor número de participantes.

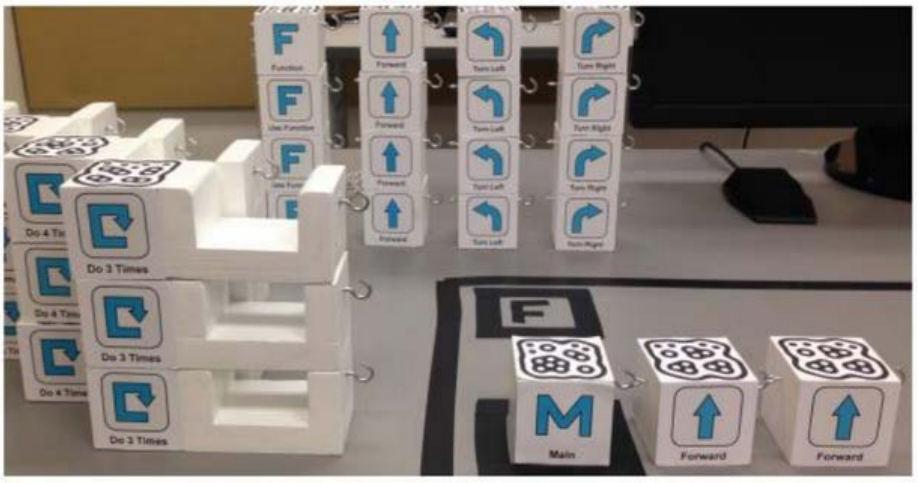

Imagen 24. La versión de bloques de programación tangibles de Bots & (Main) Frames (Melcer & Isbister, 2017, fig. 3)

- **Caceres18** (Caceres et al., 2018). Se presenta una interfaz de programación tangible denominada FYO (*Follow Your Objective* - Siga su objetivo) que está compuesto por un tablero programable, bloques basados en rompecabezas y un robot móvil como intérprete. Las y los estudiantes pueden experimentar la programación básica, sintaxis y depuración al mismo tiempo que juegan durante el proceso. La sintaxis se encuentra definida por bloques de comando (avanzar, girar izquierda/derecha e ir a la función), bloques de parámetros (uno, dos, tres, F1, entre otros) y bloques de definición. El tablero cuenta con un botón "Play" que al pulsarlo comienza a interpretar el código y lo envía al robot mediante una comunicación inalámbrica. FYO fue implementado con estudiantes de entre cinco y ocho años de edad para realizar un estudio exploratorio de su usabilidad como una plataforma de programación tangible. Los resultados obtenidos demostraron que los juegos conocidos como rompecabezas son importantes porque pueden interactuar intuitivamente con la plataforma y jugar mientras aprenden. Los autores mencionan que, si bien algunos participantes necesitaron más tiempo que el resto para programar el comportamiento del robot pudo notarse que con la práctica lograron mejorar dicho tiempo.

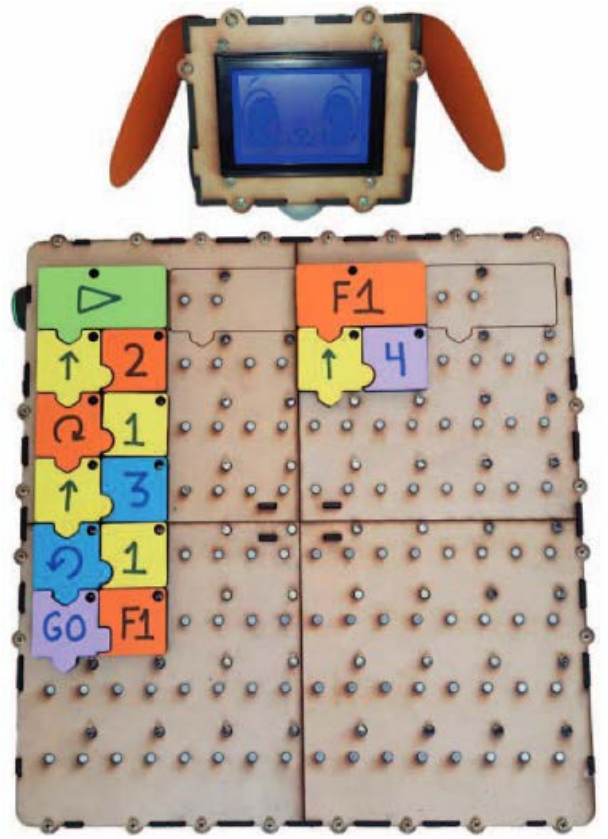

Imagen 25. Ejemplo de una secuencia de programación utilizando una función (Caceres et al., 2018, fig. 1)

- **Jin18** (Jin et al., 2018). En este documento se presenta AR-Maze, una herramienta de programación tangible pensada para el aprendizaje de programación. Este sistema hace uso de bloques que son reconocidos mediante realidad aumentada, permitiendo una retroalimentación durante la programación o la ejecución. El sistema está compuesto por AR-Maze Game una tableta inteligente que proporciona imágenes, texto y audio superpuesto a la escena real,

un mapa de ubicación y tres tipos de bloques. El objetivo consiste en mover a un personaje por distintos laberintos. Los bloques son de madera y contienen imanes en lados opuestos para ayudar a las y los estudiantes a seguir una construcción correcta. Se realizó un estudio preliminar con ocho estudiantes de cinco a nueve años de edad. Luego de la evaluación se concluye que el grupo de estudiantes fue ganando confianza e interés a medida que utilizaban la herramienta y adquirían nuevos conceptos informáticos. Se plantean como trabajos futuros realizar mejoras en el sistema presentado.

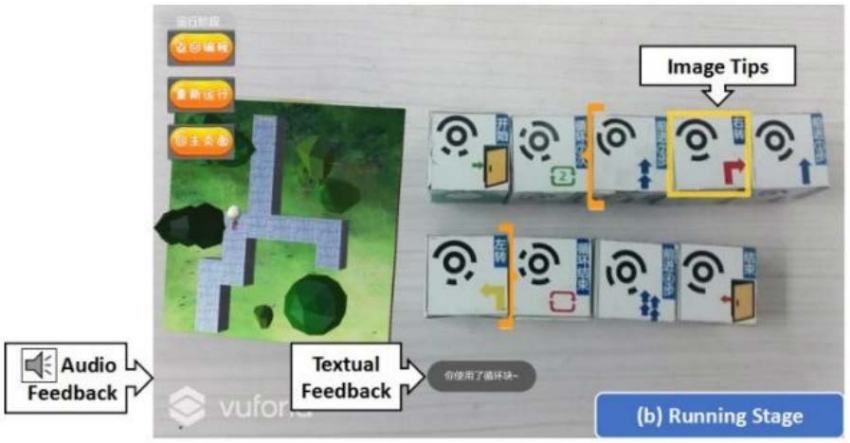

Imagen 26. Retroalimentación en la etapa de ejecución (Jin et al., 2018, fig. 4)

- **Bodén18** (Bodén et al., 2018). En este artículo se presenta DBugs, una solución tangible pensada para estudiantes de primaria (6 a 9 años) con el objetivo de generar aprendizaje colaborativo en ciencias de la computación, puntualmente descomposición de problemas, razonamiento lógico y depuración. El sistema está compuesto por cuatro bloques de colores, un bloque más grande con un botón y una cámara web y, un proyector apuntando hacia el piso. Cada bloque es entregado a un estudiante por lo que es necesario la participación de cada uno para cumplir el objetivo. El desafío consiste en hacer mover una hormiga por un laberinto. Se realizaron pruebas del sistema en dos escuelas primarias donde algunos presentaban experiencia previa en programación con Lego. Se descubrió que el gran tamaño promueve la colaboración entre las y los estudiantes, como así también la participación generalizada en la resolución de problemas.

Imagen 27. Grupo de estudiantes haciendo uso de DBugs (Bodén et al., 2018, fig. 2).

- **Koushik19** (Koushik et al., 2019). En este documento se describe y presenta StoryBlocks, un juego de programación pensado para personas con disminución visual que hace uso de bloques tangibles para representar el código. Los programas que se crean son para programar juegos o historias de audio. El sistema se compone por bloques tangibles, un espacio de trabajo y un software para interpretar y reproducir los programas. Cada bloque tiene un identificador táctil y una etiqueta visual. Los mismos deben ser ubicados en un área designada para permitir capturar mediante una cámara web las instrucciones. Para comenzar a escanear el programa, el o la estudiante debe presionar una tecla en la notebook, se procede a realizar el escaneo y se produce una salida. El sistema se puso a prueba con 16 participantes de 11 a 65 años de edad (estudiantes, docentes y personal del colegio), donde la gran mayoría no tenía conocimiento previo sobre informática. Esta evaluación demostró ser un entorno de aprendizaje accesible y atractivo para estudiantes novatos. Así mismo este sistema brinda un espacio de discusión y reflexión sobre conceptos de programación.

Imagen 28. Un estudiante de secundaria con discapacidad visual crea una historia con la ayuda de los maestros (Koushik et al., 2019, fig. 1)

- **Sabuncuoğlu19** (Sabuncuoğlu et al., 2018). En este documento se presenta Code Notes una herramienta de programación tangible de código abierto pensada para crear habilidades de pensamiento computacional. Este sistema se compone de una app Android y bloques de cartón. Los bloques son imprimibles lo que lo convierte en un recurso de bajo costo y accesible. Las instrucciones deben ser colocadas una debajo de otra para establecer la secuencia, luego mediante la app se escanea el algoritmo y este desencadena una animación en el smartphone, por ejemplo: dibujar un árbol, animar una figura geométrica, entre otras. Se concluye que su accesibilidad puede facilitar su aplicación en el aula y a otras asignaturas (matemáticas, historia, etc.). Así mismo se estima que es un buen sistema para realizar una introducción a Scratch y posteriormente a Python.

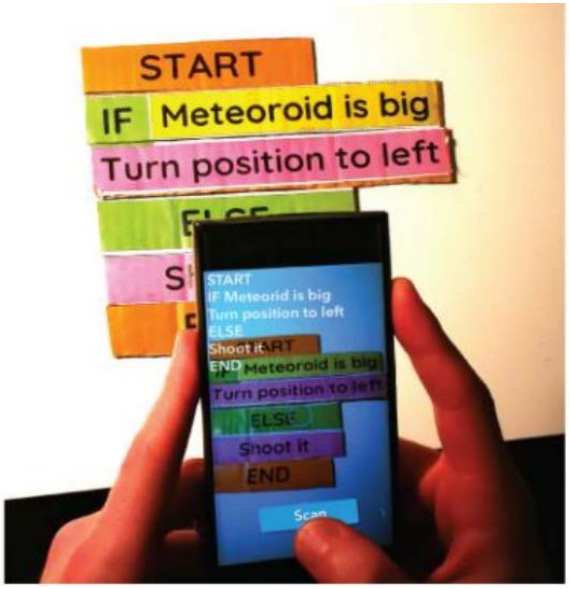

Imagen 29. Prototipo de un juego espacial para enseñar conceptos básicos de programación con Code Notes. (Sabuncuoğlu et al., 2018, fig. 2)

- **Sapounidis19** (Sapounidis et al., 2019). En este artículo se retoma el lenguaje T-ProRob presentado en Sapounidis11. El estudio involucró estudiantes de seis a

trece años que utilizaron una interfaz gráfica y otra tangible para programar el comportamiento de un robot Lego NXT. La cantidad de estudiantes que participaron fue de 148 durante un periodo de 24 meses. Se crearon parejas para las actividades propuestas y luego se les permitió interactuar libremente. El objetivo de este estudio consistió en explorar los perfiles de preferencias de los niños en GUI y TUI. Como experiencia previa las y los estudiantes estaban familiarizados con el uso de la computadora. El desafío que se proponía consistía en programar un robot Lego NXT para crear secuencias de programación simples y avanzadas. Las aplicaciones utilizadas fueron V-ProRob (gráfica) y T-ProRob (tangible). T-ProBob está compuesto de cubos de plexiglás de distintos tamaños dependiendo si es una instrucción, condicionante, repetición o parámetro. Los cubos se pueden conectar entre sí para generar secuencias de programación. Algunos de los comandos disponibles son girar a la izquierda/derecha, mover un paso hacia atrás/adelante, encender/apagar la luz, hacer un sonido, etc. Los parámetros del sensor se utilizan para los comandos de condición y pueden activar los sensores ultrasónicos, táctiles, de luz y de sonido del robot. Los cubos de parámetros son más pequeños y existen dos tipos: números y sensores. Además, es posible realizar condiciones, repeticiones y bucles anidados. Todos estos cubos se deben colocar en la "caja maestra" que al presionar el botón de ejecución lee los comandos y realiza la comunicación con una computadora en la cual se almacena el programa y posteriormente esta realiza la comunicación con el robot Lego NXT. V-ProRob se desarrolló de tal manera que cumpliera con las mismas prestaciones, pero a través de una pantalla en la cual se deben arrastrar y soltar los bloques de instrucciones para crear y ejecutar secuencias de programación. A partir de este experimento se obtuvo que la mayoría de las y los participantes expresaron su preferencia por la herramienta con TUI. Además, se observó que los estudiantes más pequeños prefirieron las TUI sobre las gráficas en todos los casos, mientras que en estudiantes mayores las preferencias se dividieron y pudo notarse una relación con el género.

Imagen 30. Cubos de plexiglás utilizados para programar de manera tangible un robot Lego NXT. (Sapounidis et al., 2019, fig. 1)

- **Mussati19** (Mussati et al., 2019). Para el desarrollo de este estudio participaron 77 estudiantes universitarios. La experiencia consistió en comparar interfaces de programación para el robot educativo Thymio. Ningún estudiante tenía conocimiento previo sobre las interfaces de programación gráfica y tangible utilizadas. Para la programación tangible se utilizaron bloques de papel que se pueden encastrar entre sí como un rompecabezas para formar la instrucción deseada. Las instrucciones elaboradas son capturadas por una cámara que se encuentra conectada a una computadora. Los datos obtenidos son procesados por un software de visión de computadora para luego ser enviados de manera inalámbrica al robot. Los comandos disponibles son los mismos que se encuentran disponibles para la GUI. Las instrucciones permitidas son eventos y acciones. Los eventos representan diferentes tipos de entradas que el robot puede recibir (por ejemplo, datos de sensores, o botones presionados) y, los bloques de acción representan acciones que el robot puede realizar (reproducir sonido, avanzar, cambiar color de LED, entre otros). También existen bloques de tipo paramétricos que permiten seleccionar diferentes sensores. Luego de utilizar ambas interfaces de programación, el lenguaje tangible propuesto parece ser más atractivo, fomentando la exploración y el descubrimiento de nuevas funcionalidades, aumentando la interacción entre estudiantes y sumando la característica de que puede ser implementado por docentes y estudiantes debido a su bajo costo de implementación.

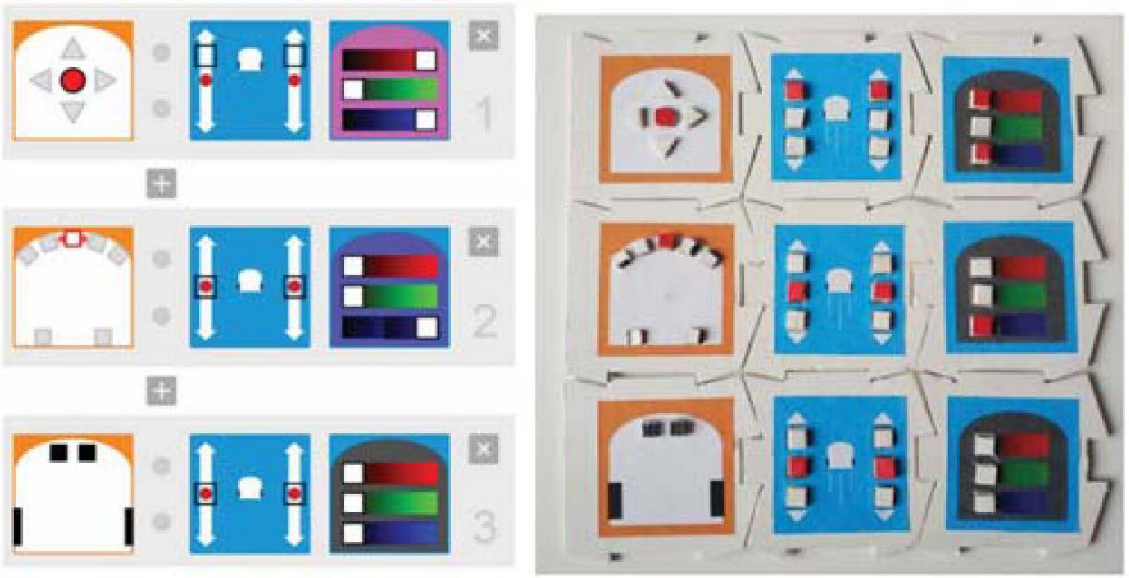

Imagen 31. A la izquierda los bloques utilizados en la GUI para programar el robot Thymio. A la derecha los mismos bloques pero tangibles para programar el mismo robot (Mussati et al., 2019, fig. 1)

- **Meadthaisong19** (Meadthaisong & Meadthaisong, 2019). En este artículo se presenta una TUI que busca desarrollar el pensamiento computacional en estudiantes de primaria. La propuesta está pensada para estudiantes que no tienen experiencia previa en programación. El objetivo consiste en programar un robot a través de piezas impresas en 3D. Las formas de las piezas no se especifican y no se alcanzan a apreciar en las imágenes proporcionadas. La programación se realiza a través de los comandos adelante/atrás, girar a la derecha/izquierda, ángulo, tiempo, sensor y motor. Los comandos tienen tarjetas RFID, los cuales mediante un lector de RFID se registran (las instrucciones) que

luego son enviadas de manera inalámbrica al robot. Los autores concluyen que diseñar una programación tangible para controlar un robot móvil es fácil para implementar con niños y niñas en las escuelas primarias. En este caso, el robot móvil de control de programación tangible se opera sin el uso de una computadora o tableta.

Imagen 32. Robot móvil, control de comandos e instrucciones (Meadthaisong & Meadthaisong, 2019, fig. 7)

- **Deng19** (Deng et al., 2019). En este artículo se presenta ARCat, una herramienta de programación tangible pensada para la enseñanza de programación a través de algoritmos de búsqueda. El sistema hace uso de realidad aumentada para mostrar el escenario por el cual se debe mover el personaje para llegar a un determinado punto. Los movimientos se programan a través de tarjetas las cuales están diferenciadas por colores según sus funcionalidades. Dichas tarjetas están hechas de papel e imán y se colocan sobre una placa de metal para definir las instrucciones. Se realizó un estudio preliminar de ARCat con seis estudiantes (8 a 9 años) donde dos tenían experiencia en programación. Antes de comenzar a utilizar el lenguaje presentaron cada una de las partes que lo componen y cómo se debe utilizar. Luego del estudio preliminar concluyen que es posible enseñar algoritmos informáticos a través de los algoritmos de búsqueda.

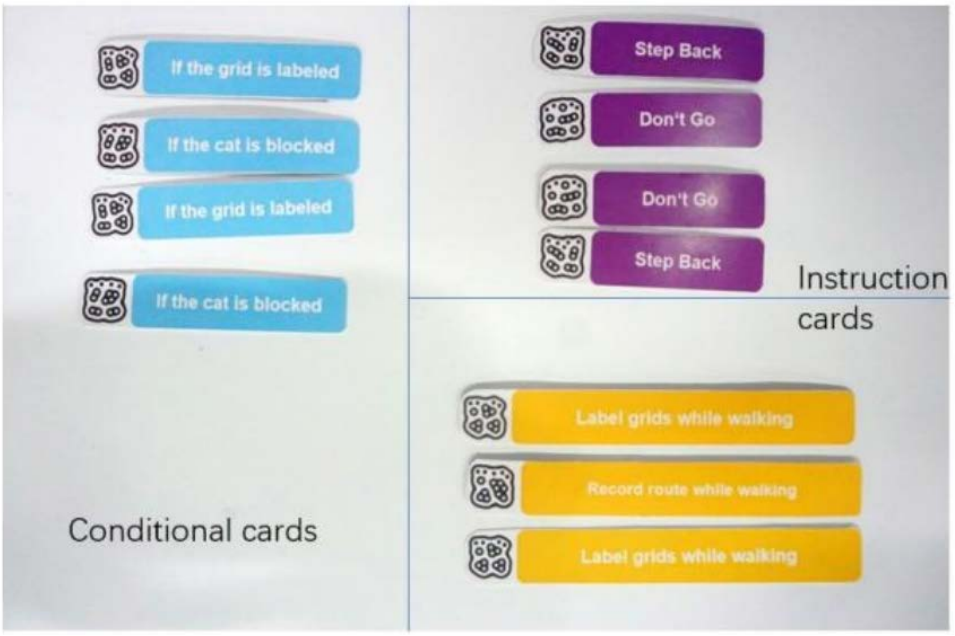

Imagen 33. Tarjetas de programación utilizadas en ARCat (Deng et al., 2019, fig. 3)

- **India19** (India et al., 2019). En este artículo se presenta una investigación centrada en el uso del lenguaje tangible Torino (Microsoft) de manera lúdica. El interrogante principal es indagar si se puede utilizar Torino de manera creativa y como un juguete para la creación de historia y música y, al mismo tiempo, introducir conceptos de programación en personas no videntes. Se realizaron varias sesiones de juego donde las y los participantes trabajaron en parejas en sesiones de 45 y 60 minutos. El grupo de estudiantes estaba constituido por 12 personas de 7 a 13 años de edad. Luego de las sesiones rescatan la importancia de mantener un ambiente de libertad creativa y exploración para el aprendizaje lúdico. Como trabajo futuro se proponen crear estrategias para establecer la comprensión y retención real de los conceptos.

Imagen 34. Estudiantes jugando con Torino en la escuela durante la sesión de juego (India et al., 2019, fig. 1)

- **Sakamoto19** (Sakamoto & Ohshima, 2019). En este artículo se presenta una herramienta educativa que hace uso de programación tangible para que las y los estudiantes de primaria adquieran conocimientos básicos de programación. Se busca que a través de pruebas y errores puedan adquirir ideas y conceptos de programación. Los conceptos básicos que se buscan incorporar son secuenciación, condicional e iteración. El objetivo consiste en programar una araña a través de tarjetas físicas para que esta se mueva y realice diferentes formas. Los tipos de comandos disponibles son inicio, fin, condicional, iterativo,

avanzar y girar derecha-izquierda. El espacio de trabajo es una mesa dividida en dos. Por un lado se colocan las tarjetas y por el otro extremo se proyecta la ejecución del programa, es decir, la araña realiza las figuras programadas. Debajo de la mesa hay una cámara conectada a una PC encargada de reconocer los comandos y del otro lado un proyector. Al momento de ejecutarse el programa los comandos son resaltados mientras se ejecutan y la araña realiza la instrucción establecida. Este sistema se puso a prueba en diferentes eventos donde observaron que no es apropiado para estudiantes de entre cinco y seis años de edad dada la dificultad de comprender las figuras geométricas. Observaron un mejor desempeño en estudiantes de diez años de edad o superior, aunque consideran que con un acompañamiento adecuado estudiantes de seis años de edad pueden hacer uso de la herramienta. Así mismo a través de distintas experiencias concluyen que la herramienta promueve la comprensión de programación mediante la cooperación.

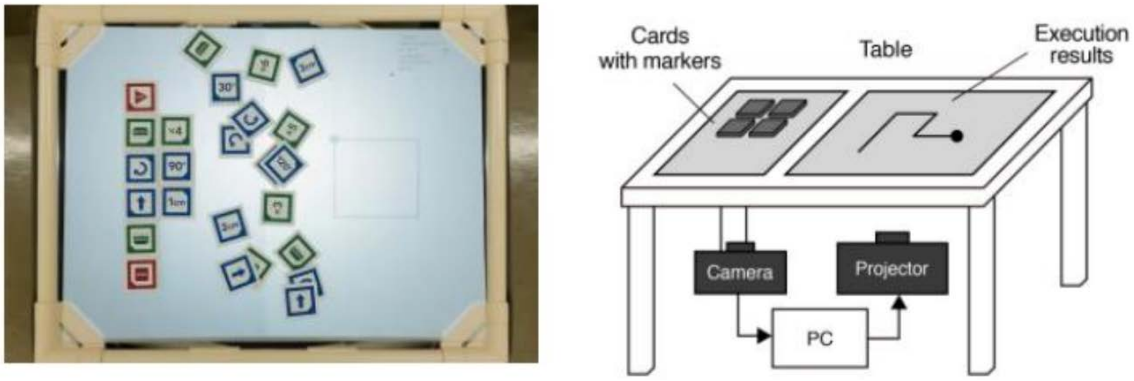

Imagen 35. Del lado izquierdo la vista superior de la mesa. Del lado derecho configuración del sistema (Sakamoto & Ohshima, 2019, fig. 2)

- **Sabuncuoğlu20** (Sabuncuoğlu & Sezgin, 2020). En este documento se expone Kart-ON, un sistema que se presenta como accesible para una introducción a la programación en personas principiantes en la temática. Se busca hacer uso de objetos cotidianos para realizar una programación tangible y un smartphone que interprete esos comandos para generar algún tipo de animación (dibujar un muñeco de nieve, mover un avión, etc.). El sistema permite utilizar tarjetas predefinidas que luego son interpretadas para realizar las acciones establecidas o, asignar, por ejemplo, a un bloque Lego una primitiva, que luego será interpretado en base a la instrucción establecida con anterioridad. La interpretación de los comandos se realiza a través de visión artificial.

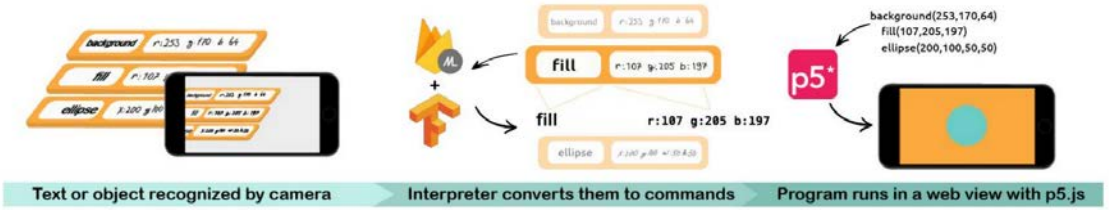

Imagen 36. Interpretación y ejecución de los comandos en Kart-ON (Sabuncuoğlu & Sezgin, 2020, fig. 1)

- **Rong20** (Rong et al., 2020). En este artículo se presenta CodeRhythm, un sistema de programación tangible pensado para que personas con disminución

visual puedan aprender conceptos básicos de programación mediante la creación de melodías sencillas. Este sistema está compuesto por bloques de sílabas (DO, RE, MI, FA, SOL, entre otras) y bloques de función (inicio, cambio, bucle y pausa). Los bloques se conectan entre sí mediante imanes, los cuales al mismo tiempo ayudan a una conexión válida entre ellos por la fuerza magnética que ejercen. Cada bloque contiene una placa Arduino Mini junto con un parlante, permitiendo que sean independientes entre sí y obteniendo una retroalimentación inmediata de cada uno sin necesidad de utilizar otros bloques. Al mismo tiempo cuentan con un relieve que permiten diferenciarlos e identificarlos para su utilización a través del tacto. Este sistema se puso a prueba con dos personas adultas con disminución visual. Luego de la ambientación e introducción y posterior utilización, quienes hicieron uso del lenguaje destacaron la independencia de los bloques para poder reconocerlos independientemente y la facilidad de conexión entre los bloques entre sí. Al mismo tiempo sugieren la incorporación de más bloques de sílabas para generar una mayor cantidad de melodías.

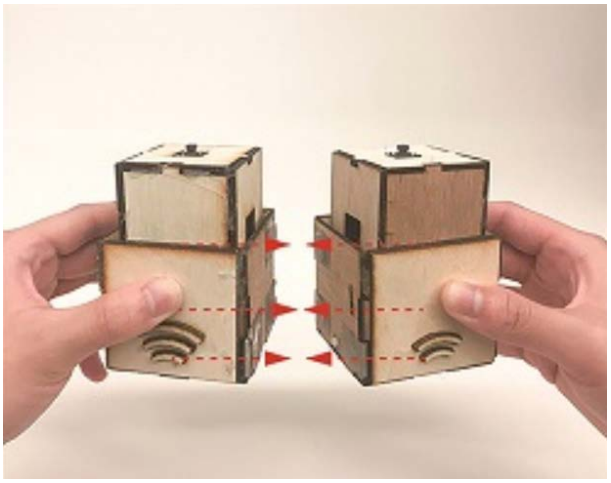

Imagen 37. Ensamble de bloques CodeRhythm (Rong et al., 2020, fig. 1)

- **Lin20** (Lin et al., 2020). En este artículo se busca crear un marco de enseñanza para mejorar las habilidades del pensamiento computacional. Para ello se basan en el método de aprendizaje lúdico utilizando una TUI. Realizaron un experimento con siete estudiantes de cinco y seis años de edad. El desafío consistía en programar un robot (auto) para que se dirija de un punto hacia el otro. La programación se realizaba mediante tarjetas de papel de colores aumentando la dificultad con diferentes obstáculos en el trayecto. El robot identificaba los colores y ejecutaba las acciones asignadas. Luego del experimento concluyen que las y los estudiantes pueden transferir sus pensamientos cognitivos fácilmente por la interacción con la TUI, pudiendo de esta manera realizar el enfoque en el pensamiento computacional. Así mismo sugieren que el aprendizaje basado en juegos junto con la TUI mejoran el rendimiento.

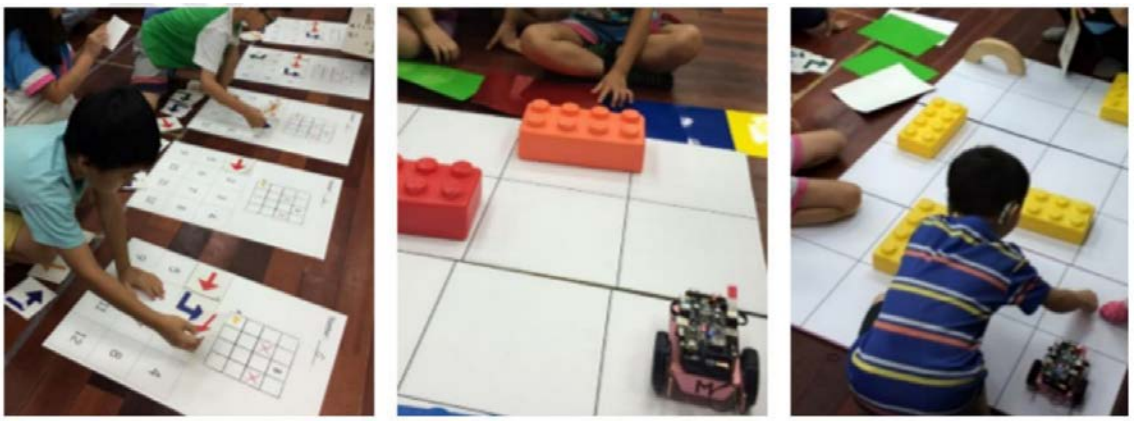

Imagen 38. TUI en todas las fases experimentales (Lin et al., 2020, fig. 9)

- **Sabuncuoglu20** (Sabuncuoglu, 2020). En este documento se presenta una plataforma de música asequible y accesible con el objetivo de enseñar conceptos básicos de programación a personas con disminución visual. Se realizó un estudio preliminar con 14 estudiantes de secundaria con dificultades visuales. Para el diseño de la TUI se consideró evitar el uso extensivo del alfabeto braille dado que las y los estudiantes pasan mucho tiempo tratando de leer y recordar los códigos. También se tuvo en cuenta en no utilizar varias modalidades de retroalimentación para el reconocimiento de los bloques porque esto puede abrumar y confundir al grupo de estudiantes. Así mismo se decidió utilizar formas abstractas y básicas en lugar de íconos detallados. Cada uno de los bloques tiene una pegatina NFC que permite ser identificado por una aplicación Android sin necesidad de hacer uso de una cámara. Luego de escanear los bloques musicales se escanea el bloque de reproducción y se comienza a ejecutar el código. Luego del estudio realizado se estima que el sistema propuesto tiene potencial en apoyar la participación y colaboración entre las y los estudiantes. Se observó además que los bloques son fáciles de diferenciar y utilizar. Así mismo el grupo de estudio manifestó estar interesado en continuar aprendiendo sobre programación.

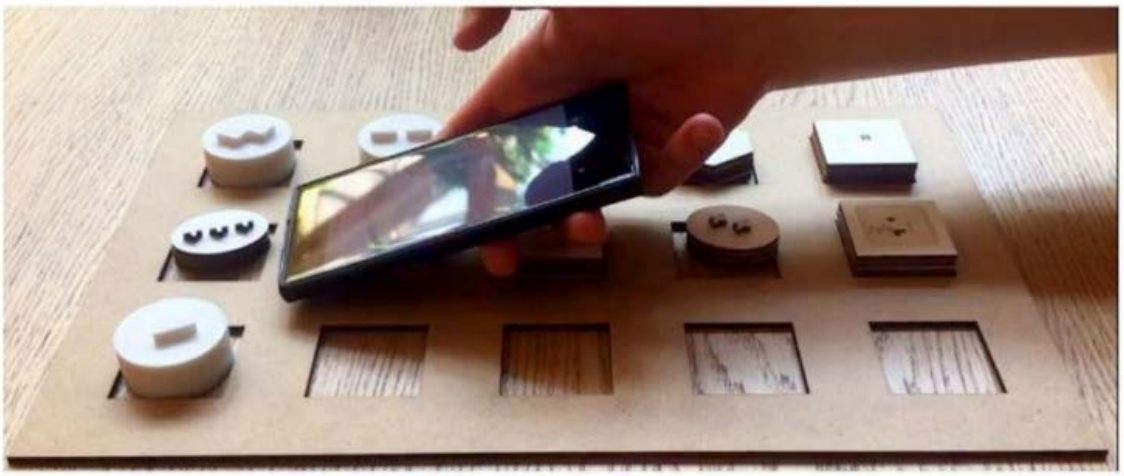

Imagen 39. Lectura NFC de un bloque de música (Sabuncuoglu, 2020, fig. 4)

- **Pires20** (Pires et al., 2020). En este documento se amplía el trabajo previo sobre entornos de programación accesibles para personas con disminución visual.

Sobre la información recabada adoptaron una solución para poner a prueba con un grupo de estudiantes (siete). La solución adoptada contaba con bloques tangibles y audio, un mapa táctil reconocible y un robot (DASH). Se utilizaron tres tipos de bloques: Play Block para ejecutar el programa, de acción (moverse, bailar, etc.) y de dirección que indica hacia dónde se debe dirigir el robot. Los bloques de acción cuentan con un botón que al ser presionado envía una señal de bluetooth a otro dispositivo (secundario) el cuál emite un sonido informativo. El mapa táctil está constituido por losas de gomaespuma con colores diferentes. Cada losa representa una unidad y los colores se intercalan para que los puedan diferenciar. Los bloques son colocados sobre una mesa para que se indique la secuencia deseada y los mismos son transcritos (por otra persona) al programa gráfico que permite una programación del robot. Se concluye que varios son los entornos tangibles surgidos en el último tiempo pero pocos pensados para ser accesibles. Se prefiere preservar las cualidades de los entornos existentes afianzados por investigaciones y prácticas para personas videntes y dar pasos cuidadosos hacia la inclusión con educadores de personas con necesidades especiales.

Imagen 40. Dos niños con discapacidad visual realizando un ejercicio dirigido a un objetivo (Pires et al., 2020, fig. 1)

- **Almjally20** (Almjally et al., 2020). En este artículo se compara el uso de una TUI y una GUI destinadas al aprendizaje de habilidades de programación. El estudio se realizó con estudiantes de una escuela primaria de Arabia Saudita. Se crearon cuatro grupos: TUI-F (estudiantes niñas), TUI-M (estudiantes niños), GUI-F (estudiantes niñas) y GUI-M (estudiantes niños). La interfaz tangible y la gráfica tienen el mismo objetivo, controlar un robot (mBot) que se encuentra sobre un tapete. Ambos sistemas fueron diseñados para ser lo más similares posible. Los bloques disponibles son acción (adelante, girar izquierda-derecha), sonido e iteración. Se realizaron sesiones de 30 a 45 minutos, donde se les solicitó completar cuestionarios para evaluar actitudes previas y posteriormente realizar las actividades propuestas. El estudio realizado sugiere que la GUI puede ser más efectiva para enseñar a estudiantes de temprana edad, mientras que la TUI puede tener un efecto positivo en actitudes hacia la informática.

Imagen 41. La configuración de TUI (izquierda) y la GUI (derecha) (Almjally et al., 2020, fig. 1)

- **Mehrptra20** (Mehrotra et al., 2020). En este artículo se presenta PaPL, un lenguaje de programación basado en papel y visión por computadora. PaPL se pensó como una versión tangible de Thymio (Mussati19). Se realizaron dos estudios para probar PaPL. El primero de ellos se llevó a cabo con 100 estudiantes de 18 y 19 años de edad, ninguno con conocimientos previos en programación. Se realizaron sesiones de 20 minutos con grupos conformados por 8 a 12 estudiantes. El segundo de los estudios se realizó en un ambiente más auténtico, con 32 estudiantes y 2 docentes. El grupo de estudiantes tenía una edad de 10 y 11 años. Ningún participante tenía conocimiento previo sobre programación o el sistema Thymio. Luego de los estudios, para el primer grupo de investigación se concluye que el tamaño del grupo puede influir en la forma de interacción con el sistema. Un grupo más reducido sería más pertinente el uso de PaPL. Así mismo este sistema puede alentar a las y los estudiantes a comparar soluciones y corregir conceptos erróneos en los demás. Para el segundo grupo de estudio el grupo de docente cree que PaPL es un buen punto de entrada antes de utilizar un sistema con pantalla. Destacan el potencial que tiene para promover la colaboración y la comunicación. Les permitió realizar un apoyo más equilibrado sobre el grupo. Así mismo creen que docentes con poca experiencia en informática pueden hacer uso del sistema sin dificultades.

Imagen 42. Estudiantes interactuando con PaPL (Mehrotra et al., 2020, fig. 3)

- **Carbajal20** (Carbajal & Baranauskas, 2020). En esta publicación se presenta TaPrEC+mBot, un sistema de programación tangible pensado para que niños y niñas aprendan conceptos básicos de programación. El sistema está compuesto por cuatro partes: Notebook+Sistema de identificación por radiofrecuencia, bloques de programación, un robot y software de control. Los bloques de programación son de madera, con forma de rompecabezas, diferenciados por colores, símbolos y un identificador RFID. En primera instancia se debe realizar la programación mediante los bloques de madera. Luego mediante el lector RFID se escanea cada uno de los bloques para que el software de control lo identifique y lo envíe a la cola de procesamiento. Seguidamente se envía mediante bluetooth las instrucciones al robot y este las ejecuta. A través de un taller de 90 minutos se puso a prueba este sistema con nueve estudiantes de cinco a seis años de edad (cuatro niñas y cinco niños). Previo a esto se realizó un encuentro con el grupo de docentes encargados del curso para planificar en conjunto una actividad significativa para el grupo de estudiantes. Luego del taller se destaca la importancia de contar con un espacio amplio para que las y los estudiantes trabajen y dispersen el material y, que de esta manera puedan jugar y experimentar. Así mismo se destaca la exploración creativa y participación activa por parte de las y los estudiantes.

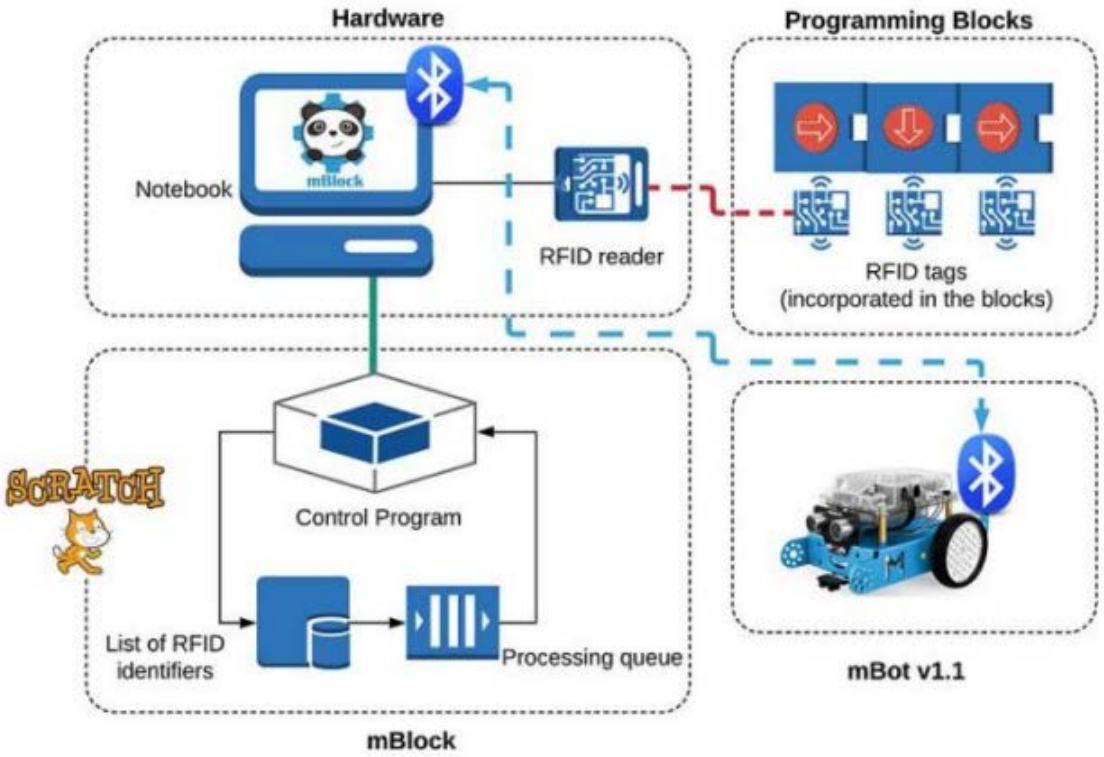

Imagen 43. Arquitectura de TaPrEC+mBot(Carbajal & Baranauskas, 2020, fig. 2)

- **Rocha21** (Rocha et al., 2021). En este documento se presenta Accembly, un lenguaje de programación tangible desarrollado para fomentar el pensamiento computacional y pensado para personas con disminución visual. El sistema está compuesto por bloques tangibles, un robot, mosaicos hechos de espuma (superficie donde se trasladará el robot) y una app diseñada para dispositivos Android. La app utiliza la cámara del dispositivo para interpretar las instrucciones (secuencia de bloques) y luego envía los comandos al robot mediante bluetooth. Los bloques tienen partes salientes e imanes que facilitan el encastre entre ellos, esto lo que busca es reducir la carga cognitiva al momento de realizar el ensamblado. Así mismo los comandos se diferencian por colores y símbolos que ayudan a identificar la acción de cada uno. Los tipos de bloques son acción, dirección y bucle. Existe un bloque play donde en la parte inferior se encuentra la secuencia de comandos y en la parte superior un botón con relieve que da inicio a la interpretación de comandos. Cuando una persona lo presiona se activa una señal auditiva alertando su inicio y la aplicación interpreta la secuencia en el espacio de trabajo. Se prepararon kits de trabajo para posteriormente realizar un estudio con siete familias en donde pudieran probar de manera autónoma y en sus hogares los kits desarrollados. Los mismos contaban con: bloques de programación, el robot programable, un smartphone con un trípode para interpretar la secuencia de comandos y una guía de actividades. Cada familia contaba con un integrante de entre 7 y 14 años de edad con alguna dificultad visual. Se realizaron registros de manera visual, es decir, grabaciones de sesiones donde se utilizaban en familia los kits, un relevamiento previo de datos que pudieran aportar información, por ejemplo, nivel educativo y vinculación con la tecnología por parte de madres y padres. Posteriormente se realizó una entrevista con el fin de que los adultos pudieran realizar sugerencias

u opiniones sobre Accembly. Destacaron la luz y el sonido ya que permite comprender a las y los estudiantes donde se encuentra el robot. Las autoras y autores concluyen que Accembly fue eficaz para el desarrollo de actividades compartidas y la utilización de conceptos de pensamiento computacional. El sistema realiza un aporte fundamental para achicar la brecha entre personas videntes y no videntes.

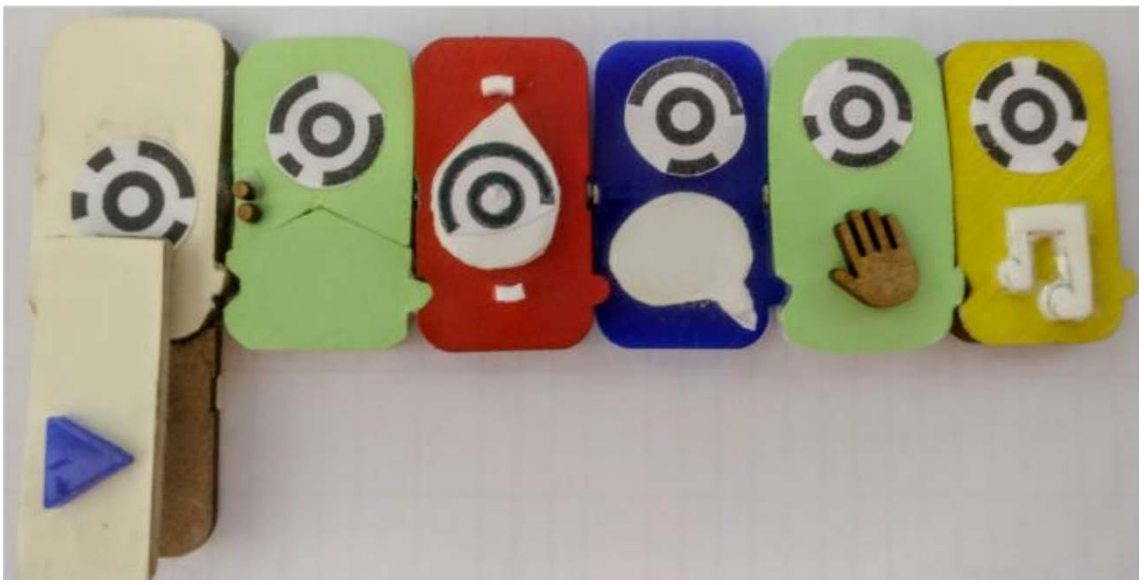

Imagen 44. Bloques de programación del sistema Accembly. (Rocha et al., 2021, fig. 2)

- **Pedersen21** (Pedersen et al., 2021). En este artículo se presentan dos interfaces de programación, una tangible (ER Kubo) y la otra virtual (ER Dash) (ver Imagen 45). Ambas interfaces presentan la misma salida, un robot que recibe las instrucciones y realiza las acciones definidas. Para este estudio primero se presentó ER Kubo y luego ER Dash con el fin de indagar si existe una transferencia de conocimiento. Dichas interfaces se pusieron a prueba con un grupo de 26 estudiantes (6 a 9 años) durante 3 semanas, donde se realizaba un encuentro semanal con una duración aproximada de dos horas. El primero de los encuentros se centró en conocer los comandos, es decir, en generar movimientos básicos en los robots. El segundo de los encuentros estuvo centrado en generar pequeños programas que resolvieran pequeños desafíos a través de la secuenciación de bloques, por ejemplo que el robot fuera de un punto a otro. El último de los encuentros estuvo centrado en la incorporación de bucles y funciones, con el objetivo de optimizar las soluciones previas. Durante los encuentros se observó que ER Kubo permitía un trabajo colaborativo entre las y los participantes dado que los bloques de programación se encontraban sobre una mesa donde el grupo de estudiantes podía participar de la solución y aportar soluciones. Esto no ocurrió con ER Dash dada la limitación de estar todo encapsulado en una pantalla. Así mismo se detectó una transferencia de conocimiento de la programación tangible a la virtual, aunque las y los autores sugieren una prueba con un mayor grupo de estudiantes. Por otro lado, sugieren implementar este mismo estudio con un grupo de estudiantes que no estén interesados con la tecnología.

Imagen 45. A la izquierda ER Kubo utilizado para realizar programación tangible, del lado derecho ER Dash utilizado para realizar programación virtual (Pedersen et al., 2021, fig. 1)

- **Henrry21** (Henrry & Dumas, 2021)**.** En este documento se presenta un sistema pensado para resolver una serie de problemas de programación a través de bloques tangibles y realidad aumentada. Cada bloque tiene un identificador que ordenados entre sí representan una respuesta ante una situación problemática propuesta. La realidad aumentada proporciona un feedback ante la solución propuesta señalando los errores si los hubiera. Las situaciones problemáticas propuestas son del tipo asignación de valores a variable, condicionales y bucles. Dichas situaciones problemáticas se presentan a través de un enunciado junto a sus bloques. El sistema descrito pretende ayudar a las y los docentes a identificar las dificultades que tiene el grupo de estudiantes en la introducción a la programación. Se puso a prueba con docentes y estudiantes por separado para evaluar su desempeño y poder considerar a este sistema como una herramienta para ser utilizada en un contexto de enseñanza-aprendizaje.

Imagen 46. Una estudiante utilizando el prototipo (Henrry & Dumas, 2021, fig. 6)

- **Cardoso21** (Cardoso et al., 2021)**.** En este artículo se presenta LEGOWorld, un sistema multimodal pensado para que niños y niñas con disminución visual puedan enfocarse en el aprendizaje de conceptos de programación y el desarrollo del pensamiento computacional a través de programación tangible o por voz[17]. Las piezas de programación están compuestas por bloques lego y tapas impresas en 3D que se colocan sobre los bloques. El objetivo consiste en programar un robot que ejecutará las instrucciones definidas. El sistema está compuesto por un pequeño robot, un espacio físico (LEGO) destinado para las acciones a realizar por el robot, bloques tangibles (LEGO) para diseñar las instrucciones y una app móvil (ver imagen 47). La utilización de los bloques LEGO permite hacer uso de las propiedades que estos traen aparejadas, algunas de ellas son la modularidad, resistencia del material, alto contraste de colores, entre otras. Los bloques deben ser colocados de manera horizontal de arriba hacia abajo sobre una superficie blanca de LEGO. Las tapas 3D permiten a través del relieve y los colores ser identificadas por personas con disminución visual. Algunas de las funciones disponibles son inicio/fin del ciclo, izquierda/derecha, play, entre otras. Así mismo se realizó una encuesta a un grupo de profesionales (docentes) sobre la accesibilidad, sus componentes y sobre cómo puede favorecer la inclusión a personas con disminución visual. A partir de ella se destaca como una solución para personas con disminución visual y del mismo modo se destacó la comprensión intuitiva, facilidad de uso y accesibilidad. Creen que puede estimular las habilidades cognitivas como la memoria, el razonamiento, entre otras. Por otro lado, sugieren que es un sistema que puede ser utilizado tanto en el aula como en el hogar.

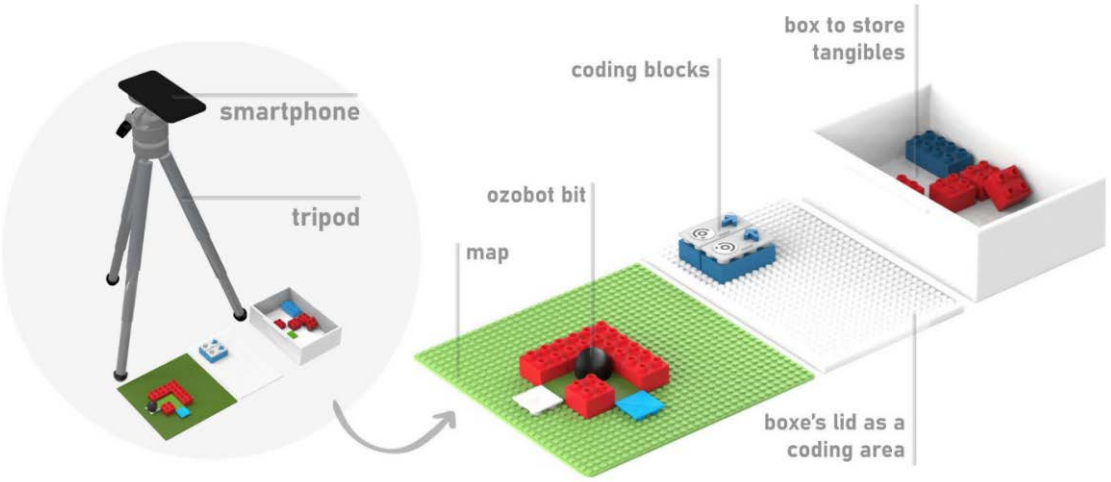

Imagen 47. Elementos que conforman al sistema LEGOWorld. (Cardoso et al., 2021, fig. 2)

- **Goolsby21** (Goolsby et al., 2021). En esta publicación se presenta un prototipo de entorno de programación tangible alternativo para que personas con disminución visual puedan hacer uso del software Scratch. El grupo objetivo son estudiantes de 6 a 12 años de edad con o sin disminución visual. El editor cuenta con los comandos de movimiento, control, operadores, variables y eventos. Los

[17] Dada la naturaleza propia de esta investigación se consideran las características relacionadas con la programación tangible.

bloques se ensamblan en una superficie estructurada donde son mapeados mediante una webcam para generar el mismo código en Scratch (ver imagen 48). Estos fueron elaborados mediante impresión 3D y para poder ser identificados (por categoría) utilizan un relieve en la parte inferior de cada bloque y, del mismo modo los comandos son representados por símbolos con el agregado de un alto contraste de colores. Así mismo los bloques cuentan con imanes que refuerzan la sensación de un encastre correcto al considerar la polarización de los mismos para un agarre más efectivo entre las piezas. Se plantea como trabajo a futuro, para lograr una mayor accesibilidad, la necesidad de utilizar un robot con el cual se pueda generar una interacción y que el espacio de trabajo no genere restricciones en el armado del programa, por ejemplo en la cantidad de bloques a utilizar.

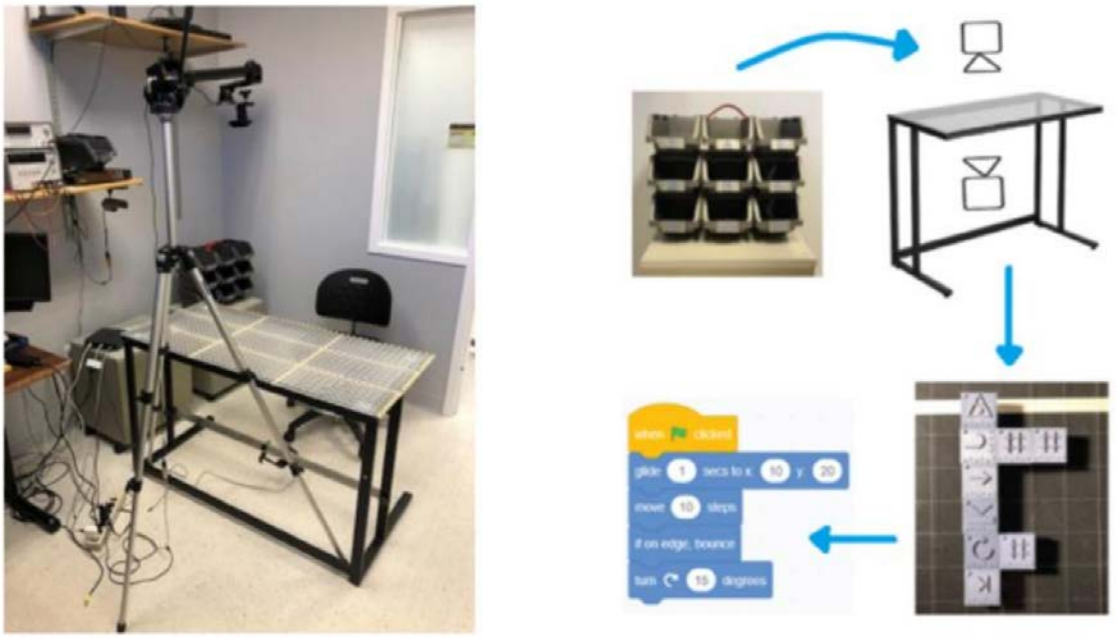

Imagen 48. Del lado izquierdo espacio de trabajo. Del lado derecho secuencia de pasos para el armado del programa. (Goolsby et al., 2021, fig. 1)

- **Im21** (Im & Rogers, 2021). En este artículo se presenta Draw2Code, un kit de computación diseñado en papel para que personas de corta edad (cinco años en adelante) puedan aprender conceptos básicos de programación mediante la creación de animaciones de realidad aumentada interactivas. Draw2Code está compuesto por bloques (hechos en papel) y una aplicación móvil. Los tipos de bloque disponibles son Sprite, Action y Event con formas y colores que ayudan a distinguirlos. El bloque Sprite permite la creación del personaje que será utilizado en la animación, es decir, en este bloque se puede dibujar el personaje o utilizar un objeto pequeño que cumpla esa función. El bloque Action permite definir el tamaño y posición del personaje (define cuándo y cómo se anima el personaje). El bloque Event permite la interacción mediante gestos con los cuáles se le indica cuándo comenzar a la animación. De manera similar a un rompecabezas los bloques tienen conectores que impiden el error de sintaxis al crear un programa. El escaneo de los comandos se realiza con la aplicación móvil y posteriormente es posible ejecutar la secuencia programada pudiendo ser visualizada la animación en la pantalla del dispositivo móvil. Se realizó una

evaluación del sistema propuesto. En dicha evaluación participaron un total de 9 niños y niñas de entre 5 y 12 años de edad acompañados por sus padres y madres. Los autores concluyeron que la instancia donde deben escanear la secuencia de comandos se torna más desafiante para aquellas personas que tienen motricidad fina débil. Por otro lado, cada vez que se debía modificar solo una parte del código era acción necesaria escanear el mismo en su totalidad, se propone como una futura mejora permitir la modificación desde la pantalla del dispositivo móvil en pos de reducir la carga de trabajo y permitir centrarse en la interacción con los bloques. El estudio sugiere que la interfaz de programación propuesta permite una colaboración natural entre un jóven con su padre y/o madres en un mismo espacio de trabajo. Draw2Code mostró ser accesible para personas jóvenes y novatas en programación, siendo este al mismo tiempo un buen primer puntapié para aprender sobre tecnologías más avanzadas. Así mismo despertó un interés e indagación (en mayores de 7 años) sobre el funcionamiento de las tecnologías implementadas en este sistema, por ejemplo el uso de realidad aumentada.

Imagen 49. Del lado izquierdo los bloques elaborados para la creación de una animación. Del lado derecho la animación en acción (Im & Rogers, 2021, fig. 1)

3.2 Resumen de artículos

A continuación se presenta la tabla 4 que resume los artículos identificados. Dicha tabla se compone por un identificador (ID), año de publicación, país donde se elaboró la publicación, el grupo destinatario, los recursos físicos, recursos digitales utilizados y la experiencia realizada.

ID	Año	País	Grupo destinatario	Recursos físicos	Recursos digitales	Objetivo/s de la investigación
Sapounidis11	2011	Grecia	Personas novatas	• Cubos activos • Caja maestra • T_Butterfly: Pantalla • T_ProRob: Robot	T_Butterfly: Animación que muestra la ejecución del programa	Abordar conceptos básicos de programación
Wang11	2012	China	Niñas y niños	• Bloques pasivos de madera • Cámara web • Sensores • Pantalla	Animación que muestra la ejecución del programa	Fomentar aprendizaje lógico y de resolución de problemas a través del juego
Kwon12	2012	Corea del Sur	Niñas y niños	• Ladrillos pasivos • Sensores led • Robot	-	Desarrollar habilidades de pensamiento lógico y resolver problemas de manera independiente
Scharf12	2012	Alemania	Niñas y niños	• Bloques activos • Computadora • Transceptores • Acelerómetro • Pantalla	Animaciones en cada bloques Animación que muestra la ejecución del programa.	Abordar la construcción algorítmica y el razonamiento. Fomentar un aprendizaje colaborativo
Sipitakiat12	2012	Tailandia	Niñas y niños	• Bloques activos • Robot	-	Ejercitar capacidades depuración
Oh13	2013	EE. UU.	Niñas y niños	• Bloques pasivos de rompecabezas • Proyector • Mesa interactiva	Animación que muestra la ejecución del programa	Desarrollar habilidades básicas de lógica y pensamiento computacional.

Chawla13	2013	EE. UU.	Niñas y niños	• Bloques pasivos de madera • Sensores • Robot	-	Comprender conceptos formales y abstractos de programación
Strawhacker13	2013	EE. UU.	Niñas y niños	• Bloques pasivos de madera • Cámara • Computadora • Robot	-	Abordar conceptos básicos de programación
Zuckerman13	2013	Israel	Estudiantes universitarios	• Bloques activos de madera	-	Abordar conceptos básicos de programación de una manera no estructurada
Kakehashi14	2014	Japón	Estudiantes de secundaria y bachillerato	• Bloques pasivos • Caja de programación • Computadora • RFID + Arduino • Robot	-	Abordar conceptos básicos de programación
Wang15	2015	China	Niñas y niños	• Arduino • Bloques activos • Pad led	-	Abordar conceptos básicos de programación
Qi15	2015	China	Niñas y niños	• Bloques pasivos • Notebook • Sensores	Animación que muestra la ejecución del programa	Abordar conceptos de programación orientada a objetos a través de la creación de historias
Hu15	2015	EE. UU.	Niñas y niños	• Bloques pasivos de madera • iPad	Animación que muestra la ejecución del programa	Abordar conceptos básicos de programación
Sullivan15	2015	EE. UU.	Niñas y niños	• Bloques pasivos de madera	-	Adquirir nociones básicas de ingeniería y programación

				• Robot		
Tada15	2015	Japón	Estudiantes de secundaria o primeros años de universidad	• Tarjetas pasivas de papel • Cámara web	Animación que muestra la ejecución del programa	Abordar conceptos básicos de programación
Zhu16	2016	Hong Kong	Niñas y niños	• Bloques pasivos de lego • Camara web • Robot	Animación que muestra la ejecución del programa	Promover el pensamiento computacional
Motoyoshi16	2016	Japón	-	• Tarjetas pasivas de poliestireno • Robot	-	Abordar conceptos básicos de programación
Wang16	2016	China	Niñas y niños	• Bloques pasivos • Robot	-	Abordar conceptos básicos de programación
Goyal16	2016	India	Niñas y niños	• Tarjetas pasivas de papel • Tablero de actividades	Animación que muestra la ejecución del programa	Aprender habilidades básicas del pensamiento computacional
Soleimani16	2016	EE. UU.	Niñas y niños	• Módulos activos	-	Experimentar y mejorar el pensamiento computacional-espacial a través de la narración de historias
Melcer17	2017	EE. UU.	Estudiantes universitarios	• Bloques pasivos de madera	Animación que muestra la ejecución del programa	Fomentar aprendizaje lógico y de resolución de problemas a través del juego
Caceres18	2018	Perú	Niñas y niños	• Tapete de programación • Bloques pasivos de rompecabezas • Robot	-	Experimentar la programación básica, sintaxis y depuración al mismo tiempo que juegan.

Jin18	2018	China	Niñas y niños	• Bloques activos de madera • Tablet	Animación que muestra la ejecución del programa	Abordar conceptos básicos de programación
Bodén18	2018	Australia	Estudiantes de primaria	• Cubos pasivos • Proyector	Animación que muestra la ejecución del programa	Promover aprendizaje colaborativo en ciencias de la computación, puntualmente descomposición de problemas, razonamiento lógico y depuración
Koushik19	2019	EE. UU.	Personas con disminución visual	• Bloques pasivos • WebCam • Notebook	Audios que reflejan la ejecución del programa	Abordar conceptos básicos de programación a través de historias.
Sabuncuoğlu19	2019	Turquía	Niñas y niños	• Bloques pasivos de papel • Smartphone	Animación que muestra la ejecución del programa	Desarrollar habilidades de pensamiento computacional
Sapounidis19	2019	Grecia	Niñas y niños	• Bloques pasivos • PC • Caja maestra • Robot	-	Abordar conceptos básicos de programación
Mussati19	2019	Suiza	Universitarios	• Bloques pasivos de papel • Sensores • PC • Robot	-	Abordar conceptos básicos de programación
Meadthaisong19	2019	Tailandia	Personas con disminución visual	• Piezas pasivas 3D • Tarjetas y lector RFID • Robot	-	Desarrollar el pensamiento computacional
Deng19	2019	China	Niñas y niños	• Tarjetas pasivas de	Animación que	Abordar la programación a

				papel	muestra la ejecución del programa	través de algoritmos de búsqueda
India19	2019	India	Personas con disminución visual	• Bloques pasivos	-	Introducir conceptos de programación mediante la creación de historias y música
Sakamoto19	2019	Japón	Niñas y niños	• Tarjetas activas • Pc • Proyector • Mesa	Animación que muestra la ejecución del programa	Adquirir conocimientos básicos de programación
Sabuncuoğlu20	2020	Turquía	Personas novatas	• Tarjetas pasivas predefinidas • Smartphone	Animación que muestra la ejecución del programa	Introducir conceptos de programación
Rong20	2020	Hong kong	Personas con disminución visual	• Bloques activos • Arduino • Parlantes	-	Aprender conceptos básicos de programación mediante la creación de melodías sencillas
Lin20	2020	Taiwan	Niñas y niños	• Tarjetas pasivas de papel • Placa Arduino • Robot	-	Mejorar las habilidades del pensamiento computacional
Sabuncuoglu20	2020	Turquía	Niños y niñas no videntes	• Bloques pasivos de madera • Smartphone	-	Adquirir conceptos básicos de programación mediante la creación de melodías
Pires20	2020	Portugal	Niños y niñas no videntes	• Bloques activos • Robot • Gomaespuma • Parlante	-	Abordar conceptos básicos de programación
Almjally20	2020	Arabia Saudita	Primaria	• Bloques pasivos 3D • Tableta • Robot	-	Aprendizaje de habilidades de programación
Mehrptra20	2020	Suiza	Niñas y niños	• Tarjetas pasivas de papel	-	Abordar conceptos básicos de programación

				• Robot		
Carbajal20	2020	Brasil	Niñas y niños	• Bloques pasivos de madera • Robot • Lector RFID	-	Abordar conceptos básicos de programación
Rocha21	2021	Portugal	Niñas y niños	• Bloques pasivos • Smartphone • Mosaicos de espuma • Robot	-	Fomentar el pensamiento computacional
Pedersen21	2021	Dinamarca	Niñas y niños	• Bloques pasivos • Robot	-	Evaluar la transferencia de conocimiento de la programación tangible a la virtual
Henrry21	2021	Bélgica	-	• Bloques pasivos • Smartphone	Animación que muestra la ejecución del programa	Resolver problemas utilizando programación
Cardoso21	2021	Portugal	Niñas y niños	• Bloques pasivos • Smartphone • Robot	-	Aprender conceptos de programación y el desarrollo del pensamiento computacional
Goolsby21	2021	EE. UU.	Niñas y niños	• Bloques pasivos • Webcam	Animación que muestra la ejecución del programa	Abordar conceptos básicos de programación
Im21	2021	EE. UU.	Niñas y niños	• Bloques pasivos de papel • Smartphone	Animación que muestra la ejecución del programa	Abordar conceptos básicos de programación
ID	**Año**	**País**	**Grupo destinatario**	**Recursos físicos**	**Recursos digitales**	**Objetivo/s de la investigación**

Tabla 4. Caracterización de los artículos seleccionados para el análisis.

Capítulo 4. Discusión

En este capítulo se presenta un resumen de los resultados obtenidos. Se expone la cantidad de artículos por país (Sección 4.1), la cantidad de publicaciones realizadas por año (Sección 4.2), la comparación de TUI y GUI (Sección 4.3), el grupo etario destinatario (Sección 4.4), propuestas que se piensan para la inclusión de personas no videntes (Sección 4.5), los recursos lógicos y físicos implementados (Sección 4.6) y por último los objetivos didácticos (Sección 4.7).

4.1 Artículos por países

A partir de la información recabada se obtiene que el mayor flujo de artículos relacionados con la temática proviene de EE. UU, seguido por China y un poco más por debajo le sigue Japón. A continuación, se resumen los datos mediante una tabla y un gráfico ilustrativo (ver Tabla 5 e Gráfico 1).

País	Cantidad de artículos por país
EE. UU.	10
China	6
Japón	4
Turquía	3
Portugal	3
Grecia	2
Tailandia	2
Hong Kong	2
India	2
Suiza	2
Corea del Sur	1
Alemania	1
Israel	1
Perú	1
Australia	1
Taiwán	1
Arabia Saudita	1
Dinamarca	1
Brasil	1
Bélgica	1

Tabla 5. Resumen de cantidad de artículos por país.

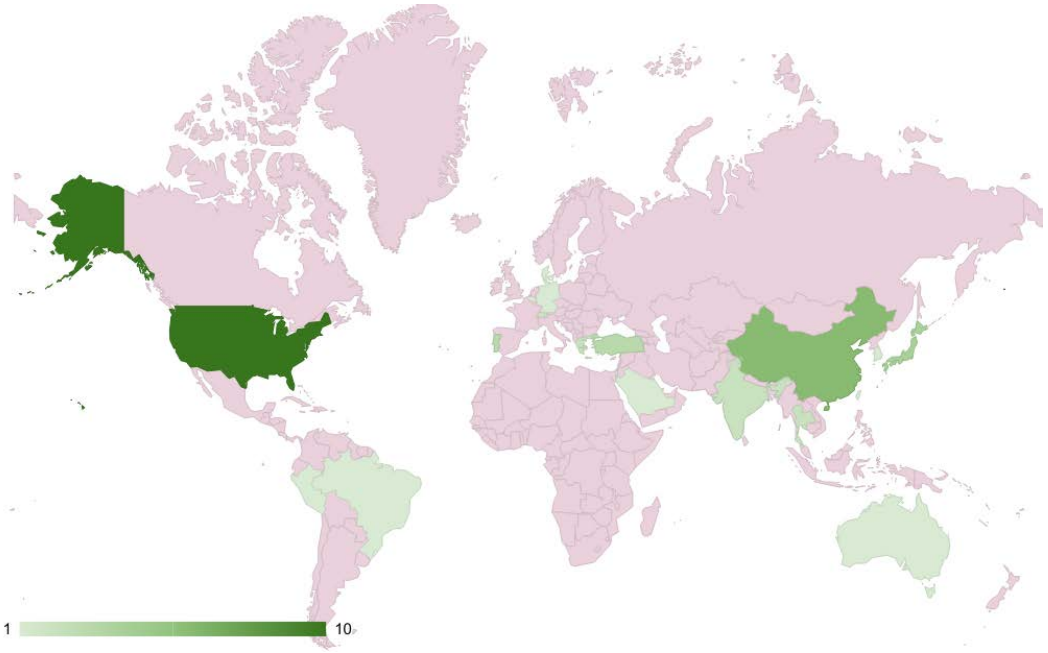

Gráfico 1. Cantidad de artículos por país.

4.2 Artículos por años

A partir del análisis de los años en el que se publicaron los artículos localizados, se observa que la cantidad de artículos por años ha ido aumentando considerablemente, aunque existen algunos descensos notables. Para el año 2011 se identificó un único artículo, los años siguientes, 2012 y 2013, presentan cuatro artículos para cada uno de ellos, lo que denota en un incremento, pero, en el 2014 se observa un nuevo descenso a un único artículo. En los años 2015 y 2016 se observa un nuevo ascenso, teniendo cinco artículos por cada uno de los años, aunque nuevamente se genera un nuevo descenso con un único artículo para el año 2017 y tres para el año 2018. Para los años siguientes, 2019 y 2020, se observa la mayor frecuencia de documentos seleccionados, siendo ocho en total para cada uno de los años, finalizando la búsqueda con seis artículos para el año 2021. A continuación se presenta un gráfico que resume lo expuesto (ver Gráfico 2).

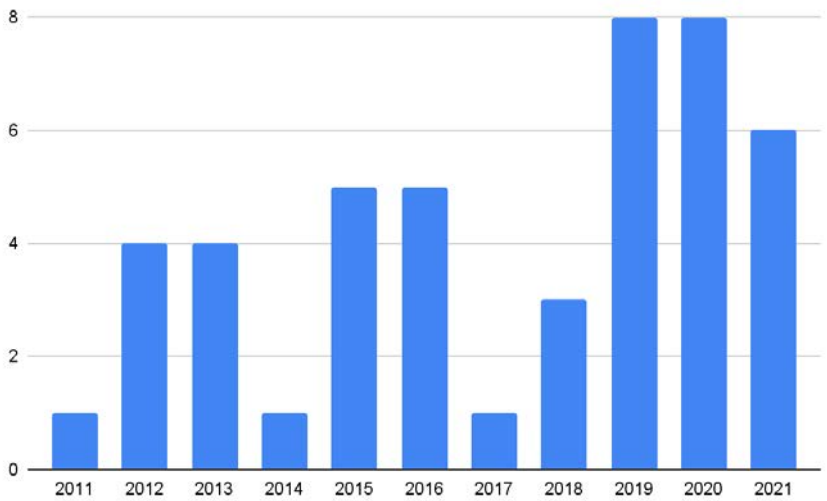

Gráfico 2. Cantidad de artículos seleccionados por año

4.3 TUI vs GUI

Entre los artículos analizados en esta investigación, existe un grupo (5/46) que se ha centrado en la comparación de las TUI con las GUI. El objetivo principal está enfocado en identificar en qué aspectos resulta más apropiado un tipo de interfaz que otro en situaciones de enseñanza y aprendizaje de programación. **Kwon12** concluye que el entorno tangible resultó ser una herramienta eficaz para que las y los estudiantes puedan manifestar habilidades de pensamiento lógico y resuelvan problemas por su cuenta. Destaca que la versión TUI fomenta la exploración de soluciones utilizando la intuición. Así mismo, **Zuckerman13** obtuvo que gran parte del grupo de estudiantes prefieren la TUI por sobre la GUI, independientemente del género o con qué interfaz interactuaron en primera instancia. Los motivos estuvieron dados porque permite una interacción física, por proporcionar una retroalimentación agradable y producir un alto nivel de realismo. En **Zhu16,** el estudio realizado sugiere que si bien se observó que con la versión GUI las y los estudiantes se distrajeron menos, la versión TUI posibilitó que tuvieran más argumentos para discutir las soluciones. En **Sapounidis19** se obtuvo que la mayoría de las y los participantes expresaron su preferencia por la TUI. Además, se observó que los estudiantes más pequeños prefirieron las TUI sobre las gráficas en todos los casos, mientras que en estudiantes mayores las preferencias se dividieron y pudo notarse una relación con el género. Por último, en **Mussati19**, luego de utilizar ambas interfaces de programación, se concluye que el lenguaje tangible propuesto resultó ser más atractivo. Al igual que en **Kwon12**, se encuentra un beneficio en cuanto a la exploración y el descubrimiento de nuevas funcionalidades. Se destaca que en la versión TUI se nota un aumento en la interacción entre estudiantes y puede ser implementado por su bajo costo de elaboración.

Para todos los casos abordados el grupo de estudio no llega a ser significativo como para realizar afirmaciones que puedan sustentar a una interfaz por sobre la otra.

4.4 Grupo etario destinatario

En los artículos analizados se identifica que las propuestas de TUI se piensan para estudiantes de cuatro años de edad en adelante. Se destaca una mayor cantidad de artículos que piensan las TUI para un grupo etario comprendido entre los seis y nueve años de edad. Luego, se nota que la cantidad de artículos orientados a grupos etarios de mayor edad comienza a disminuir (ver Gráfico 3).

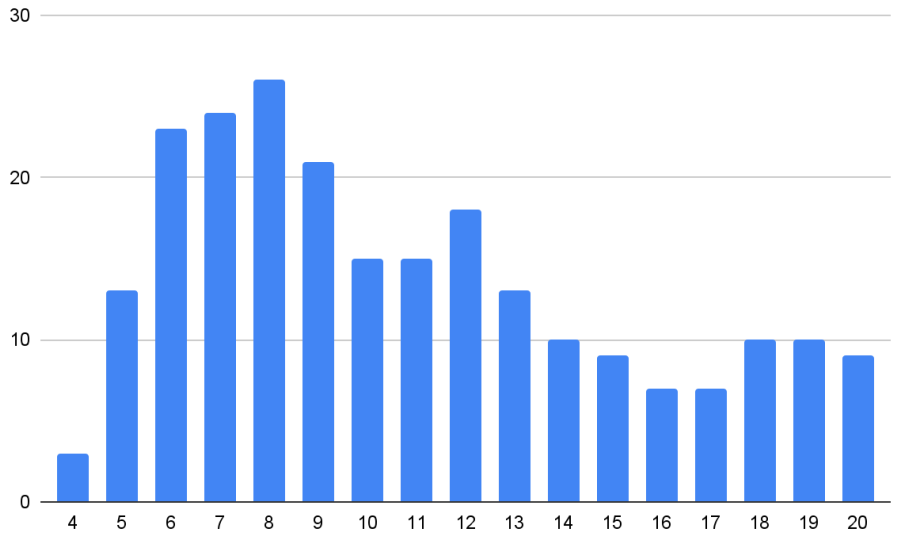

Gráfico 3. Frecuencia absoluta del grupo etario destinatario más frecuente identificado

4.5 TUI para la inclusión

Un subconjunto de los artículos analizados (8/46) (**Koushik19**, **Meadthaisong19**, **India19**, **Rong20**, **Sabuncuoglu20**, **Pires20**, **Cardoso21** y **Goolsby21**) proponen las TUI como una herramienta de inclusión, en particular, el grupo destinatario son personas ciegas o con disminución visual. En gran parte de las TUI identificadas, el o la estudiante necesita hacer uso de su visión para poder utilizar la herramienta, por ejemplo, identificar un bloque por su color o dibujo. En cambio, los 8 trabajos mencionados hacen uso de las TUI con un valor agregado, generando piezas que puedan ser identificadas a través del tacto por personas con disminución visual y, en algunos casos por personas sin dificultades visuales (**Rocha21**). Estas propuestas se identifican a partir del año 2019 en adelante.

4.6 Recursos físicos y digitales implementados

En el análisis realizado se pudo identificar que, una gran variedad de recursos físicos son utilizados en las propuestas, siendo los bloques de madera recurrentes en varios artículos como elementos implementados para crear los programas (10/46). Luego le sigue el uso de tarjetas o papel para definir las instrucciones (8/46). El resto de los bloques utilizados se distribuyen por estar elaborados en metal, impresión 3D, bloques Lego u otro componente. Así mismo, gran parte de los objetos utilizados para definir las instrucciones son pasivos en su gran mayoría (36/46), siendo muy pocos los que son activos (10/46). Otro factor interesante es la salida de la secuencia programada, es decir, donde se ejecutan las instrucciones. Aquí se identifica otro elemento recurrente que es un robot (22/46), que varía de artículo en artículo, es decir, algunos son robots comerciales (por ejemplo Lego) (10/22) y en otros casos son robots que se diseñan *ad-hoc* (12/22). A continuación, se ilustra una nube de palabras con los elementos tangibles utilizados (ver Figura 3).

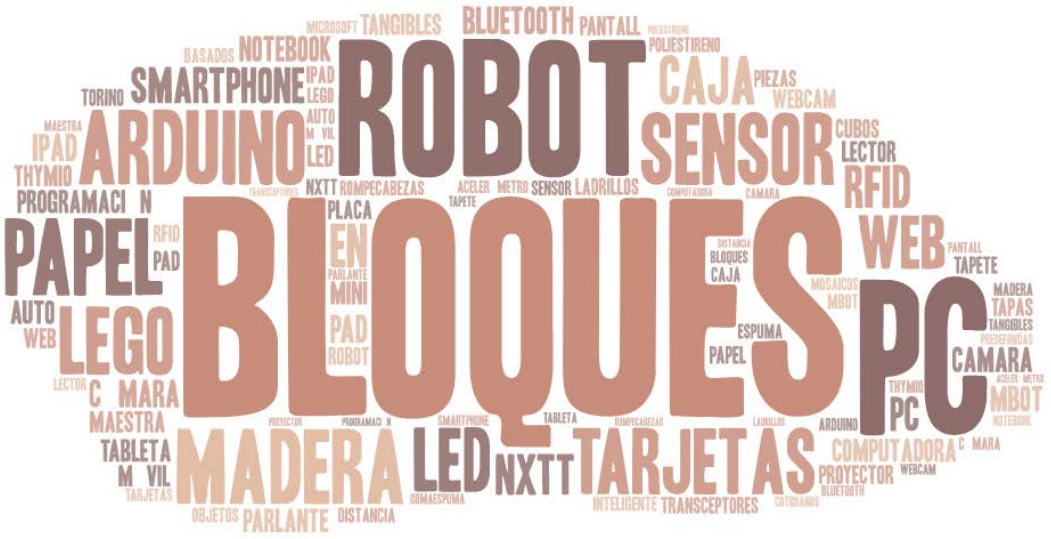

Figura 3. Nube de palabras con los recursos físicos identificados en los artículos (producción propia).

Por el lado de los recursos digitales, se identifica que gran parte de las TUI implementan una animación que muestra la ejecución del programa (19/46), por lo general un avatar que debe recorrer un laberinto. En menor medida se utiliza la creación de melodías como resultado de las instrucciones definidas (5/46).

4.7 Objetivos de las investigaciones

Al realizar un análisis de las experiencias descritas en los artículos identificados se observa uno o varios objetivos en cada una de ellas. Si bien el objetivo con mayor preponderancia es el de Abordar nociones básicas de programación (35/46) se encuentran otros objetivos como: Promover el pensamiento computacional (9/46). Luego aparecen el Fomentar el razonamiento lógico (6/46) y Ejercitar la resolución de problemas (5/46). Solo en unos pocos casos se incorpora Ejercitar capacidades de depuración (3/46), Fomentar el aprendizaje lúdico (3/46), Fomentar el aprendizaje colaborativo (2/46) y Abordar aspectos de la construcción algorítmica (2/46) (ver tabla 6).

Objetivo	Cantidad de artículos que lo menciona
Abordar nociones básicas de programación	35
Promover el pensamiento computacional	9
Fomentar el razonamiento lógico	6
Ejercitar la resolución de problemas	5
Ejercitar capacidades de depuración	3
Fomentar el aprendizaje lúdico	3

Fomentar el aprendizaje colaborativo	2
Abordar aspectos de la construcción algorítmica	2

Tabla 6. Resumen de los objetivos más mencionados en las experiencias.

Capítulo 5. Conclusiones y trabajo futuro

A partir del análisis de los resultados obtenidos en este último capítulo se exponen las conclusiones obtenidas y el trabajo a futuro.

5.1 Conclusiones

Para el desarrollo del presente trabajo de investigación se empleó la revisión sistemática de literatura. Dicha revisión consistió en (a) definir preguntas de investigación, (b) trazar una estrategia de búsqueda (dónde buscar, con qué palabras claves), (c) establecer criterios de inclusión y exclusión, que fueron aplicados, tanto para la selección inicial, como para la selección final.

La motivación de esta investigación estuvo dada por describir un estado del arte de la utilización de la programación tangible en el contexto educativo. Para ello se definieron los siguientes objetivos de investigación:

- Identificar las bases teóricas y/o enfoques que sustentan la programación tangible.
- Identificar distintas experiencias que utilizan programación tangible en contextos educativos en los últimos diez años.
- Definir un conjunto de criterios que permitan el análisis de las experiencias seleccionadas.
- Reconocer ventajas y desventajas de la utilización de la programación tangible en contextos educativos.
- Analizar las experiencias educativas que hacen uso de programación tangible.

Se puede decir que Project Blocks, el trabajo de Suzuki & Kato (1993), es uno de los más significativos junto a los postulados de Ishii & Ullmer (1997). Para estos últimos, un lenguaje de programación tangible es similar a uno visual o de texto, aunque en lugar de usar una computadora como intermediaria se utilizan objetos físicos cotidianos que representan las instrucciones que se utilizaran. En estos primeros años se desarrollaron lenguajes de programación tangibles como Tangible Programming Bricks o Quetzal.

Para identificar las distintas experiencias que hacen uso de programación tangible en contextos educativos en los últimos diez años se recurrió a una revisión sistemática de literatura en los repositorios IEEE Xplore, ACM Digital Library y ScienceDirect.

El conjunto de criterios que permitieron un análisis de las experiencias fueron:

- País donde se ha elaborado la publicación.
- Objetivos de la investigación.
- Características de los destinatarios: rango etario, si poseen alguna discapacidad y experiencia previa en programación.
- Experiencias con el grupo destinatario.
- TUI

- Implementación, costos y comandos disponibles
- Recursos físicos: bloques, pantalla, robot, parlantes, luces, etc.
- Recursos digitales: animaciones, audios, etc.

A partir de la información obtenida, la programación tangible se presenta como una alternativa para situaciones de enseñanza y aprendizaje de la programación. En el estudio se observó que los destinatarios fueron estudiantes a partir de los cuatro años hasta los primeros años de universidad. En particular, se encontró una mayor frecuencia de trabajos para estudiantes de 6 a 9 años de edad. Además, se encontraron múltiples trabajos que realizan un aporte proponiendo herramientas de programación tangible destinadas a alumnos con discapacidad visual.

De los trabajos que realizaron comparaciones entre aplicaciones de programación gráfica y tangible, se notó que en ambos tipos de interfaces se evidenciaron preferencias y aspectos de interés y que para las TUI se destacaron aspectos como la retroalimentación y las posibilidades de exploración, entre otras.

En cuanto a las características de la programación tangible mencionadas en las experiencias analizadas se destacan: promover que el foco de atención se encuentre en la actividad a resolver, puesto que permite que el interés de las y los participantes no se encuentre afectada por el uso de la computadora; y posibilitar el desarrollo de propuestas de aspecto lúdico para jugar mientras se aprende. Además, se busca generar instancias de colaboración entre las y los estudiantes, tratando de no generar competencia entre los pares. Otro aspecto que se observa de las aplicaciones de programación tangible es que al utilizar objetos físicos conocidos, se interactúe de manera intuitiva con la aplicación, como en el caso de las piezas de rompecabezas en la plataforma de programación FYO. Otra característica recurrente en la gran mayoría de las TUI analizadas es la utilización de bloques pasivos para definir las instrucciones, solo unas pocas hacen uso de bloques activos.

Recuperando las experiencias mencionadas, si bien la mayoría de los trabajos tienen como objetivo abordar nociones básicas de programación y, son destinados a estudiantes (6 a 9 años de edad) o personas novatas en programación, también existen propuestas con objetivos más generales y que se aplican más allá de la programación en sí, como es el promover el pensamiento computacional, el razonamiento lógico y el aprendizaje colaborativo, entre otros.

Se puede notar que con la mayoría de los lenguajes analizados en este estudio solo se han realizado pruebas piloto para ver su desempeño y usabilidad. Se destacan los trabajos de Koushik19, Meadthaisong19, India19, Rong20, Sabuncuoglu20, Pires20, Cardoso21, Goolsby21 por proponer TUI que puedan ser utilizadas por personas con discapacidad visual y de esta manera puedan contar con herramientas en el inicio de la programación.

5.2 Trabajo futuro

En base a la investigación realizada se espera poder validar los criterios de evaluación definidos a través de la validación por juicio de expertos y su aplicación en

nuevas experiencias. Esto permitirá estandarizar el uso de los criterios definidos. Asimismo, se pretende diseñar y desarrollar una TUI que se presente como una alternativa para incorporar las nociones básicas sobre programación en alumnas y alumnos que no tienen conocimientos previos sobre esta temática. Se considera de interés tener presente que la herramienta sea accesible y usable por personas con problemas motrices o visuales. La implementación y prueba de la TUI permitirá, además, sumar resultados que den sustento a un enfoque de aprendizaje de la programación más intuitivo y usable a través del uso de programación tangible. Al mismo tiempo se pretende recabar información que sustente las ventajas y desventajas en el uso de una TUI en un primer acercamiento hacia la programación.

Referencias

Tanenbaum, S. Andrew. Organización de computadoras: un enfoque estructurado. PRENTICE HALL. México 2000. ISBN 970-17-0399-5

Martínez López, Pablo E. Las bases conceptuales de la Programaci´on: Una nueva forma de aprender a programar / - 1ra ed. - La Plata : el autor, 2013. EBook. http://www.gobstones.org/bibliografia/Libros/BasesConceptualesProg.pdf

Bers, M. U., & Horn, M. S. (2010). Tangible programming in early childhood. High-tech tots: Childhood in a digital world, 49, 49-70. https://doi.org/10.1109/TE.2012.2190071

Marshall, P., Price, S., y Rogers, Y. (2003). Conceptualising tangibles to support learning. En Proceedings of the 2003 conference on interaction design and children (pp. 101–109). New York, NY, USA: ACM.Descargado de http://doi.acm.org/10.1145/953536.953551 doi: 10.1145/953536.953551 https://doi.org/10.1109/EDUCON.2017.7943096

Price, S. (2008). A representation approach to conceptualizing tangible learning environments. En Proceedings of the 2nd international conference on tangible and embedded interaction (pp. 151–158). New York, NY, USA: ACM. Descargado de http://doi.acm.org/10.1145/1347390.1347425 doi:10.1145/1347390.1347425 https://doi.org/10.1109/TE.2018.2876363 https://doi.org/10.1109/TE.2018.2876363

Almjally, A., Howland, K., & Good, J. (2020). Comparing TUIs and GUIs for Primary School Programming. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 521-527.

https://doi.org/10.1145/3328778.3366851

Alvarado, M. C., Sanz, C., & Baldassarri, S. (2020). Análisis de experiencias con objetos activos en actividades educativas basadas en interacción tangible. *Revista de la Asociación Interacción Persona Ordenador (AIPO)*, *1*(1), Art. 1.

Bodén, M., Pretorius, B., Matthews, B., & Viller, S. (2018). DBugs: Large-scale artefacts for collaborative computer programming. *Proceedings of the 17th ACM Conference on Interaction Design and Children*, 545-550. https://doi.org/10.1145/3202185.3210773

Caceres, P. C., Venero, R. P., & Cordova, F. C. (2018). Tangible programming mechatronic interface for basic induction in programming. *2018 IEEE Global Engineering Education Conference (EDUCON)*, 183-190. https://doi.org/10.1109/EDUCON.2018.8363226

Carbajal, M. L., & Baranauskas, M. C. C. (2020). Analyzing the socioenactive dimensions of creative learning environments with preschool children. *Proceedings of the 19th Brazilian Symposium on Human Factors in Computing Systems*, 1-10. https://doi.org/10.1145/3424953.3426510

Cardoso, G., Pires, A. C., Abreu, L. V., Rocha, F., & Guerreiro, T. (2021). LEGOWorld: Repurposing Commodity Tools & Technologies to Create an Accessible and Customizable Programming Environment. *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, 1-6. https://doi.org/10.1145/3411763.3451710

Chawla, K., Chiou, M., Sandes, A., & Blikstein, P. (2013). Dr. Wagon: A «stretchable» toolkit for tangible computer programming. *Proceedings of the 12th International Conference on Interaction Design and Children*, 561-564.

https://doi.org/10.1145/2485760.2485865

Resolución CFE N. 343/18, Pub. L. No. 343 (2018). https://www.argentina.gob.ar/sites/default/files/res_cfe_343_18_0.pdf

Deng, X., Wang, D., Jin, Q., & Sun, F. (2019). ARCat: A Tangible Programming Tool for DFS Algorithm Teaching. *Proceedings of the 18th ACM International Conference on Interaction Design and Children*, 533-537. https://doi.org/10.1145/3311927.3325308

Goolsby, B., Pawluk, D., Kim, H. W., & Fusco, G. (2021). A Tangible Block Editor for the Scratch Programming Language. *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, 1-6. https://doi.org/10.1145/3411763.3451833

Goyal, S., Vijay, R. S., Monga, C., & Kalita, P. (2016). Code Bits: An Inexpensive Tangible Computational Thinking Toolkit For K-12 Curriculum. *Proceedings of the TEI '16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction*, 441-447. https://doi.org/10.1145/2839462.2856541

Horn, M. S., & Jacob, R. J. K. (2007). Designing tangible programming languages for classroom use. *Proceedings of the 1st international conference on Tangible and embedded interaction*, 159-162. https://doi.org/10.1145/1226969.1227003

Hu, F., Zekelman, A., Horn, M., & Judd, F. (2015). Strawbies: Explorations in tangible programming. *Proceedings of the 14th International Conference on Interaction Design and Children*, 410-413. https://doi.org/10.1145/2771839.2771866

Im, H., & Rogers, C. (2021). Draw2Code: Low-Cost Tangible Programming for Creating AR Animations. *Interaction Design and Children*, 427-432. https://doi.org/10.1145/3459990.3465189

India, G., Ramakrishna, G., Bisht, J., & Swaminathan, M. (2019). Computational Thinking as Play: Experiences of Children who are Blind or Low Vision in India. *The 21st International ACM SIGACCESS Conference on Computers and Accessibility*, 519-522. https://doi.org/10.1145/3308561.3354608

Ishii, H., & Ullmer, B. (1997). Tangible bits: Towards seamless interfaces between people, bits and atoms. *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, 234-241. https://doi.org/10.1145/258549.258715

Jacob, R. J. K. (s. f.). *What is the next generation of human-computer interaction? | CHI '06 Extended Abstracts on Human Factors in Computing Systems* (world). Recuperado 17 de diciembre de 2020, de https://dl.acm.org/doi/abs/10.1145/1125451.1125768

Jin, Q., Wang, D., Deng, X., Zheng, N., & Chiu, S. (2018). AR-maze: A tangible programming tool for children based on AR technology. *Proceedings of the 17th ACM Conference on Interaction Design and Children*, 611-616. https://doi.org/10.1145/3202185.3210784

Julià, C. F., Gallardo, D., & Jordà, S. (2009). *TurTan: Un Lenguaje de Programación Tangible*. 10.

Kakehashi, S., Motoyoshi, T., Koyanagi, K., Oshima, T., Masuta, H., & Kawakami, H. (2014). Improvement of P-CUBE: Algorithm education tool for visually impaired persons. *2014 IEEE Symposium on Robotic Intelligence in Informationally Structured Space (RiiSS)*, 1-6. https://doi.org/10.1109/RIISS.2014.7009180

Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering – A systematic

literature review. *Information and Software Technology*, *51*(1), 7-15. https://doi.org/10.1016/j.infsof.2008.09.009

Koushik, V., Guinness, D., & Kane, S. K. (2019). StoryBlocks: A Tangible Programming Game To Create Accessible Audio Stories. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1-12. https://doi.org/10.1145/3290605.3300722

Kwon, D.-Y., Kim, H.-S., Shim, J.-K., & Lee, W.-G. (2012). Algorithmic Bricks: A Tangible Robot Programming Tool for Elementary School Students. *IEEE Transactions on Education*, *55*(4), 474-479. https://doi.org/10.1109/TE.2012.2190071

Lin, S.-Y., Chien, S.-Y., Hsiao, C.-L., Hsia, C.-H., & Chao, K.-M. (2020). Enhancing Computational Thinking Capability of Preschool Children by Game-based Smart Toys. *Electronic Commerce Research and Applications*, *44*, 101011. https://doi.org/10.1016/j.elerap.2020.101011

Marshall, P., Price, S., & Rogers, Y. (2003). Conceptualising tangibles to support learning. *Proceedings of the 2003 conference on Interaction design and children*, 101-109. https://doi.org/10.1145/953536.953551

Marshall, P., Rogers, Y., & Hornecker, E. (2007). *Are tangible interfaces really any better than other kinds of interfaces?* CHI'07 workshop on Tangible User Interfaces in Context & Theory, San Jose, California, USA. http://www.cl.cam.ac.uk/conference/tangibleinterfaces/

McNerney, T. S. (2004). From turtles to Tangible Programming Bricks: Explorations in physical language design. *Personal and Ubiquitous Computing*, *8*(5), 326-337. https://doi.org/10.1007/s00779-004-0295-6

Meadthaisong, S., & Meadthaisong, T. (2019). Mobile Robot Control by Tangible programming for Developmenting of Computer Scientist Thinking skill in Elementary School. *2019 16th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, 45-48. https://doi.org/10.1109/ECTI-CON47248.2019.8955308

Mehrotra, A., Giang, C., Duruz, N., Dedelley, J., Mussati, A., Skweres, M., & Mondada, F. (2020). Introducing a Paper-Based Programming Language for Computing Education in Classrooms. *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*, 180-186. https://doi.org/10.1145/3341525.3387402

Melcer, E., & Isbister, K. (2017). Embodiment, collaboration, and challenge in educational programming games: Exploring use of tangibles and mouse. *Proceedings of the 12th International Conference on the Foundations of Digital Games*, 1-6. https://doi.org/10.1145/3102071.3116222

Motoyoshi, T., Tetsumura, N., Masuta, H., Koyanagi, K., Oshima, T., & Kawakami, H. (2016). Tangible gimmick for programming education using RFID systems. *IFAC-PapersOnLine*, *49*(19), 514-518. https://doi.org/10.1016/j.ifacol.2016.10.608

Mussati, A., Giang, C., Piatti, A., & Mondada, F. (2019). A Tangible Programming Language for the Educational Robot Thymio. *2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA)*, 1-4. https://doi.org/10.1109/IISA.2019.8900743

Oh, H., Deshmane, A., Li, F., Han, J. Y., Stewart, M., Tsai, M., Xu, X., & Oakley, I.

(2013). The digital dream lab: Tabletop puzzle blocks for exploring programmatic concepts. *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction*, 51-56. https://doi.org/10.1145/2460625.2460633

Pedersen, B. K. M. K., Jacobsen, D. M., Teichert, L. J. L., & Nielsen, J. (2021). Educational Robotics and Mediated Transfer: Transitioning from Tangible Tile-based Programming, to Visual Block-based Programming. *Companion of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, 402-406. https://doi.org/10.1145/3434074.3447201

Pires, A. C., Rocha, F., de Barros Neto, A. J., Simão, H., Nicolau, H., & Guerreiro, T. (2020). Exploring accessible programming with educators and visually impaired children. *Proceedings of the Interaction Design and Children Conference*, 148-160. https://doi.org/10.1145/3392063.3394437

Price, S. (2008). A representation approach to conceptualizing tangible learning environments. *Proceedings of the 2nd international conference on Tangible and embedded interaction*, 151-158. https://doi.org/10.1145/1347390.1347425

Qi, Y., Wang, D., Zhang, L., & Shi, Y. (2015). TanProStory: A Tangible Programming System for Children's Storytelling. *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, 1001-1006. https://doi.org/10.1145/2702613.2732806

Réalité augmentée et interface tangible au service des novices en programmation | 32e Conférence Francophone sur l'Interaction Homme-Machine. (s. f.). Recuperado 5 de septiembre de 2022, de https://dl.acm.org/doi/10.1145/3451148.3458643

Riedenklau, E., Hermann, T., & Ritter, H. (2010). *Tangible Objects and Interactive*

Sonification as a Scatter Plot Alternative for the Visually Impaired. Georgia Institute of Technology. https://smartech.gatech.edu/handle/1853/50065

Rocha, F., Pires, A. C., Neto, I., Nicolau, H., & Guerreiro, T. (2021). Accembly at Home: Accessible Spatial Programming for Children with Visual Impairments and their Families. *Interaction Design and Children*, 100-111. https://doi.org/10.1145/3459990.3460699

Rong, Z., Chan, N. F., Chen, T., & Zhu, K. (2020). CodeRhythm: Designing Inclusive Tangible Programming Blocks. *Companion Publication of the 2020 ACM Designing Interactive Systems Conference*, 105-110. https://doi.org/10.1145/3393914.3395895

Sabuncuoglu, A. (2020). Tangible Music Programming Blocks for Visually Impaired Children. *Proceedings of the Fourteenth International Conference on Tangible, Embedded, and Embodied Interaction*, 423-429. https://doi.org/10.1145/3374920.3374939

Sabuncuoğlu, A., Erkaya, M., Buruk, O. T., & Göksun, T. (2018). Code notes: Designing a low-cost tangible coding tool for/with children. *Proceedings of the 17th ACM Conference on Interaction Design and Children*, 644-649. https://doi.org/10.1145/3202185.3210791

Sabuncuoğlu, A., & Sezgin, M. (2020). Kart-ON: Affordable Early Programming Education with Shared Smartphones and Easy-to-Find Materials. *Proceedings of the 25th International Conference on Intelligent User Interfaces Companion*, 116-117. https://doi.org/10.1145/3379336.3381472

Sakamoto, R., & Ohshima, T. (2019). Code Weaver: A Tangible Programming Learning Tool with Mixed Reality Interface. *SIGGRAPH Asia 2019 Posters*, 1-2.

https://doi.org/10.1145/3355056.3364561

Sapounidis, T., & Demetriadis, S. (2011). Touch Your Program with Hands: Qualities in Tangible Programming Tools for Novice. *2011 15th Panhellenic Conference on Informatics*, 363-367. https://doi.org/10.1109/PCI.2011.5

Sapounidis, T., Stamovlasis, D., & Demetriadis, S. (2019). Latent Class Modeling of Children's Preference Profiles on Tangible and Graphical Robot Programming. *IEEE Transactions on Education*, *62*(2), 127-133. https://doi.org/10.1109/TE.2018.2876363

Scharf, F., Winkler, T., Hahn, C., Wolters, C., & Herczeg, M. (2012). Tangicons 3.0: An educational non-competitive collaborative game. *Proceedings of the 11th International Conference on Interaction Design and Children*, 144-151. https://doi.org/10.1145/2307096.2307113

Sipitakiat, A., & Nusen, N. (2012). Robo-Blocks: Designing debugging abilities in a tangible programming system for early primary school children. *Proceedings of the 11th International Conference on Interaction Design and Children*, 98-105. https://doi.org/10.1145/2307096.2307108

Soleimani, A., Green, K. E., Herro, D., & Walker, I. D. (2016). A Tangible, Story-Construction Process Employing Spatial, Computational-Thinking. *Proceedings of the The 15th International Conference on Interaction Design and Children*, 157-166. https://doi.org/10.1145/2930674.2930703

Strawhacker, A., Sullivan, A., & Bers, M. U. (2013). TUI, GUI, HUI: Is a bimodal interface truly worth the sum of its parts? *Proceedings of the 12th International Conference on Interaction Design and Children*, 309-312. https://doi.org/10.1145/2485760.2485825

Sullivan, A., Elkin, M., & Bers, M. U. (2015). KIBO robot demo: Engaging young children in programming and engineering. *Proceedings of the 14th International Conference on Interaction Design and Children*, 418-421. https://doi.org/10.1145/2771839.2771868

Tada, K., & Tanaka, J. (2015). Tangible Programming Environment Using Paper Cards as Command Objects. *Procedia Manufacturing*, *3*, 5482-5489. https://doi.org/10.1016/j.promfg.2015.07.693

Wang, D., Zhang, C., & Wang, H. (2011). T-Maze: A tangible programming tool for children. *Proceedings of the 10th International Conference on Interaction Design and Children*, 127-135. https://doi.org/10.1145/1999030.1999045

Wang, D., Zhang, L., Qi, Y., & Sun, F. (2015). A TUI-based Programming Tool for Children. *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education*, 219-224. https://doi.org/10.1145/2729094.2742630

Wang, D., Zhang, L., Xu, C., Hu, H., & Qi, Y. (2016). A Tangible Embedded Programming System to Convey Event-Handling Concept. *Proceedings of the TEI '16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction*, 133-140. https://doi.org/10.1145/2839462.2839491

Zhu, K., Ma, X., Wong, G. K. W., & Huen, J. M. H. (2016). How Different Input and Output Modalities Support Coding as a Problem-Solving Process for Children. *Proceedings of the The 15th International Conference on Interaction Design and Children*, 238-245. https://doi.org/10.1145/2930674.2930697

Zuckerman, O., & Gal-Oz, A. (2013). To TUI or not to TUI: Evaluating performance and preference in tangible vs. graphical user interfaces. *International Journal of*

Human-Computer Studies, *71*(7), 803-820.

https://doi.org/10.1016/j.ijhcs.2013.04.003

Suzuki, H., & Kato, H. (1993, August). AlgoBlock: a tangible programming language, a tool for collaborative learning. In Proceedings of 4th European Logo Conference (pp. 297-303).