E.T.S. de Ingeniería Industrial,
Informática y de Telecomunicación

Exploración sobre alfabetos de gestos en aplicaciones prearitméticas interactivas para educación infantil

Grado en Ingeniería Informática

Trabajo Fin de Grado

Autor: Alberto Ginesta Ruiz de Azagra

Directores: Mikel Aldaz Zaragüeta

Alfredo Pina Calafi

Pamplona, 1 de julio de 2016

upna Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Resumen

El propósito de este proyecto es adquirir conocimientos para desarrollar aplicaciones dirigidas al ámbito de la educación infantil. Como demostración de que se han asimilado e interiorizado dichos conocimientos se pretende desarrollar una aplicación educativa, para dispositivos táctiles Android.

La aplicación, llamada *Súmate*, pretende servir tanto como vehículo introductorio a las operaciones aritméticas más básicas como de herramienta de refuerzo al conteo.

Súmate permite al usuario trazar números y trata de reconocerlos. Cuando un número es reconocido se muestra un conjunto de círculos como representación del mismo. Una vez se dispone de varios conjuntos en pantalla se permite operar sobre ellos dando lugar a nuevos conjuntos.

Palabras clave:

Sumar, conteo, aritmética, números, educación infantil, *Súmate,* Android.

Agradecimientos

A Mikel Aldaz y Alfredo Pina, codirectores de este TFG, sin vosotros hubiese sido imposible llevar este proyecto a buen puerto.

A Benoît, vecino de laboratorio y apoyo fundamental.

A toda mi familia, por animarme y confiar en mí. En especial a mi madre, el pilar que me sostiene. A mi hermana, que siempre tuvo tiempo para darme su enfoque docente. A mi abuela, que no pudo ver este proyecto terminado.

A todos los compañeros de promoción que me hicieron crecer como persona. En especial gracias a Álvaro e Irene por estar en los buenos y en los malos momentos. Amigos y compañeros de viaje.

A mis amigos, que siempre tuvieron unas palabras de ánimo.

Al colegio Irabia-Izaga por permitirnos probar la aplicación en su centro.

Al Liceo Monjardín, antiguo colegio Ursulinas, por poner los cimientos que me han hecho llegar hasta aquí.

A ti, que has decidido dedicar unos segundos a leer estas líneas.

Exploración sobre alfabetos de gestos en prearitmética interactiva para educación infantil.

ÍNDICE

1. INTRODUCCIÓN

1.1 Propósito del proyecto

El propósito de este proyecto es adquirir conocimientos para desarrollar aplicaciones dirigidas al ámbito de la educación infantil. Como demostración de que se han asimilado e interiorizado dichos conocimientos se pretende desarrollar una aplicación educativa, para dispositivos táctiles Android, que sirva tanto de vehículo introductorio a las operaciones aritméticas más básicas como de herramienta de refuerzo al conteo.

Teniendo en cuenta los objetivos anteriores se puede considerar esta aplicación como un "juego serio" o "juego formativo" que busca entretener a la vez que se aprenden y asimilan conceptos de una forma interactiva.

1.2 Breve descripción

La aplicación, llamada "*Súmate"*, permite al usuario trazar números y trata de reconocerlos. Cuando un número es reconocido se muestra un conjunto de círculos como representación del mismo. Una vez se dispone de varios conjuntos en pantalla se permite operar sobre ellos dando lugar a nuevos conjuntos.

Actualmente se permiten las siguientes operaciones:

- Sumar/Juntar dos conjuntos.
- Visualizar que número representa un conjunto.
- Eliminar un conjunto de la pantalla.
- Eliminar todos los conjuntos de la pantalla.

Además se ha dotado a la aplicación de sonidos que permitan asociar fácilmente cada operación con la función que realiza.

1.3 Software utilizado.

1.3.1 Android Studio

Se ha escogido Android Studio[1] como entorno de desarrollo ya que, a pesar de consumir bastantes recursos, su interfaz resulta muy cómoda, integra una herramienta de control de versiones relativamente personalizable e incluye una *intelligence* muy avanzada.

Cuenta con un servicio de virtualización que permite ejecutar un proyecto casi en cualquier tipo de dispositivo (móvil, tablet, televisión) además de numerosos paquetes que permiten convertir AndroidStudio en un entorno totalmente personalizado.

1.3.2 Libgdx

Se ha utilizado Libgdx[2] (framework para desarrollo de juegos multiplataforma) porque permite utilizar openGL sin tener que programar a muy bajo nivel. Además presenta una gran compatibilidad con Android Studio y es bastante potente, por lo que a la hora de seguir desarrollando la aplicación, la elección de este framework no supone ningún tipo de limitación. Otro punto fuerte es que es multiplataforma, de forma que esta aplicación podría correr no solo en un dispositivo Android sino también sobre un ordenador (con ciertas limitaciones al no ser un dispositivo multitáctil) o sobre un dispositivo IOS.

1.3.3 Reconocedor $1

Se realizó un trabajo de indagación previo a la elección del reconocedor a utilizar. Se consideraron una serie de herramientas y se eligió un reconocedor $1 teniendo en cuenta las necesidades de la aplicación a desarrollar.

Se necesitaba un reconocedor liviano puesto que en una aplicación de este tipo resulta vital la fluidez de la misma. Además se necesitaba un reconocedor adaptable y flexible en el que se pudiesen introducir modificaciones de forma sencilla para aproximarlo a la lógica de la aplicación. También se valoró que permitiera cierta escalabilidad (hay muchas formas de dibujar un mismo número) y por supuesto, que presentase un alto grado de fiabilidad.

Fue complicado encontrar un reconocedor que cumpliese todos los requisitos anteriores. *Kivy* fue una de las herramientas descartadas ya que únicamente permitía definir gestos a nivel de código. Se convertía así esta tarea en una labor demasiado tediosa y que no garantizaba la flexibilidad necesaria para el correcto funcionamiento de la aplicación.

El que mejor se ajustaba a las necesidades era *Touch*, herramienta que permitía definir gestos realizándolos directamente sobre un dispositivo táctil y que operaba tanto con uno como con varios trazos. Lamentablemente esta herramienta no estaba disponible para descarga.

Tras descartar algunas opciones más se decidió optar por reconocedor de la familia $. Dentro de esta familia se establece una diferenciación entre los reconocedores que operan sobre un único trazo ($1) y los que permiten reconocer gestos a partir de varios ($N y $P). Se descartó utilizar uno de los multitrazo debido a su complejidad y a la dificultad de encontrar una implementación adaptable a la idea de la aplicación. En su lugar se optó por partir de un reconocedor $1 y realizar las modificaciones necesarias que más adelante se detallan.

2. PLANTEAMIENTO DIDÁCTICO DE LA APLICACIÓN

2.1 Importancia de las nuevas tecnologías en la educación

Las nuevas tecnologías están cada vez más presentes en la sociedad y nos permiten realizar numerosas tareas de manera cómoda y accesible. Aunque hace algunos años era impensable, son incontables las veces que, a lo largo del día, accedemos a distintas aplicaciones a través de nuestro teléfono móvil, ordenador o *tablet*.

Lo mismo ocurre en el ámbito educativo. Desde la llegada de la era digital, los maestros tienen a su alcance un sinfín de recursos disponibles que han revolucionado la labor docente. Los dispositivos que se están implantando en las aulas (pizarra digital, libros electrónicos, etc) ofrecen a los escolares nuevas posibilidades de aprendizaje que eran inaccesibles con los recursos convencionales. Por todo ello, el objetivo es aprovechar las posibilidades de los medios digitales para crear herramientas que permitan aprender interactivamente, más fácilmente y más profundamente.

Además, desde bien pequeños, los niños tienen acceso a estos dispositivos y están acostumbrados a manipularlos (en ocasiones mejor que los adultos), tanto en el hogar como en la escuela. Asimismo, al estar tan presentes en su vida diaria, se han convertido en una herramienta cercana y muy atractiva para ellos. No obstante, debemos asegurarnos de que desarrollen la capacidad de utilizarlos de manera responsable y crítica, de forma que puedan sacar el máximo partido de ellos.

2.2 Perfil de usuario

Las funcionalidades desarrolladas, así como el interfaz visual, fueron diseñados orientados a usuarios de entre 4 y 6 años que estén introduciéndose en el mundo de la aritmética más básica.

Según las etapas evolutivas de Piaget[3], los niños de esta edad se encuentran en el estadio de pensamiento intuitivo, que abarca de los 4 a los 7 años. En esta fase, los niños empiezan a ser capaces de mostrar su pensamiento lógico, pasan de la manipulación a la representación y surgen las operaciones matemáticas.

Sin embargo, es recomendable que el uso de esta aplicación sea tutelado por un usuario de mayor edad que fije objetivos, incentive y guíe al niño ayudándole a explorar todas las posibilidades que ofrece la aplicación.

2.3 Características pedagógicas de la aplicación

Teniendo en cuenta la gran incidencia que tiene la utilización de las nuevas tecnologías en la enseñanza y las características evolutivas de los destinatarios, a continuación se detallan las particularidades de la aplicación en cuanto al proceso de aprendizaje se refiere.

Tomando como referencia la Teoría de las Inteligencias Múltiples de Howard Gardner[4], se puede afirmar que el ser humano no posee una única inteligencia, sino que nace con un total de ocho inteligencias diferentes, que va desarrollando a lo largo de su vida. En la medida en que vaya ejercitando cada una de ellas, le resultará más o menos sencillo desempeñar unas tareas u otras.

A su vez, esto implica que distintos alumnos pueden interiorizar un mismo concepto por vías diferentes y, por lo tanto, se deben ofrecer diversos caminos para garantizar que todos ellos llegan a asimilar el contenido propuesto en función de la inteligencia que tienen más desarrollada.

Una de las vías de entrada más efectivas en el aprendizaje, y más aún en edades tan tempranas, es la visual (inteligencia viso-espacial). Por ello, la aplicación trata de presentar la información de forma clara y con un interfaz minimalista, evitando distracciones para tratar de mantener al usuario centrado y concentrado únicamente en lo que se quiere realizar. Es por esto por lo que no se utilizan diferentes formas de presentar la información y se trata de mantener un aspecto uniforme en todo momento.

De la misma forma, tomando como referencia las Regletas Cuisenaire (material empleado en matemáticas en educación infantil), se ha asignado a cada conjunto un color, en función del número de elementos que lo componen. Así, a simple vista el usuario puede identificar el conjunto tanto por el color como por la cantidad.

Además de los estímulos visuales, muchos niños (y también adultos) tienen más facilidad para recordar o aprender algo cuando lo relacionan con un sonido concreto (inteligencia auditiva-musical). Por este motivo, parece acertado también completar este aprendizaje con estímulos sonoros, asociando un sonido representativo a cada movimiento que permite realizar la aplicación.

Otro aspecto importante en el aprendizaje en educación infantil es el movimiento (inteligencia corporal-cinestésica). Por esta razón, la asimilación de los contenidos se apoya en la utilización de las manos para realizar cualquier operación. Por ejemplo, la acción de sumar dos elementos queda perfectamente representada por el gesto necesario para llevarla a cabo, puesto que el niño debe desplazar ambos conjuntos hasta juntarlos y obtener el resultante, haciendo real la acepción de sumar como reunir o agrupar. De este modo, es previsible que el usuario pueda interiorizar la información de forma mucho más rápida que si utilizase una tiza y una pizarra o incluso un ordenador.

Además de su interfaz amigable, al ser una aplicación que opera sobre un dispositivo táctil, puede resultar una forma más atractiva de aprender o trabajar aspectos que con los recursos tradicionales, que pueden parecer más monótonos o aburridos.

Cuando los niños empiezan a interactuar con su entorno y con las personas que les rodean, aumentan exponencialmente sus opciones de aprendizaje. La observación,

la imitación y la relación con sus iguales hacen que el alumno se plantee nuevas cuestiones o retos que no habrían aflorado trabajando de forma individual (inteligencia interpersonal). No son pocos los alumnos que entienden un concepto gracias a la explicación de un compañero. Por eso, la aplicación está pensada tanto para un único usuario como para parejas o grupos pequeños, permitiendo que los niños se retroalimenten entre ellos.

También resulta interesante tener en cuenta que, tras un primer contacto, las ganas de continuar utilizando una aplicación disminuyen a la par que lo hace la sensación de novedad. Por esto resulta fundamental que la aplicación sea interactiva, de forma que, en función de lo que en usuario realice, aparezcan distintos resultados. En caso contrario la aplicación está abocada al fracaso.

De la misma manera, es importante recordar que el objetivo al utilizar la aplicación nunca debe ser sustituir otras metodologías de aprendizaje. El enriquecimiento será mucho mayor si la empleamos como refuerzo o complemento a lo que el usuario esté aprendiendo en la escuela.

3. ARQUITECTURA

El proyecto consta de tres partes diferenciadas (reconocedor, lógica del juego y generación de salida gráfica). Por ello se ha decidido agrupar todas las clases necesarias en tres paquetes distintos.

3.1 Distribución

3.1.1 Paquete dollar1Recognizer

Este paquete encierra toda la funcionalidad correspondiente al reconocimiento. Analiza el trazo realizado y busca coincidencias con alguna figura conocida.

Incluye una versión adaptada del reconocedor $1 ideado por Jacob O. Wobbrock, Andrew D. Wilson y Yang Li [5]. Este reconocedor ha sido el utilizado para tratar de identificar el número que el usuario está tratando de dibujar.

Se ha partido de una implementación en Java realizada por Alex Olwal. Ésta se encuentra alojada en GitHub [6]. Las clases de mayor interés dentro de este paquete son *TemplateData, Utils* y *Recognizer*. La primera contiene todas las plantillas entre las cuales se tratará de buscar coincidencias. La segunda realiza todas las operaciones necesarias para normalizar el trazo realizado por el usuario (rota, escala y traslada), encierra dicho trazo en una *BoundingBox*. Además implementa un método que devuelve el menor ángulo entre una plantilla y el trazo dibujado. Por último, la clase *Recognizer*, busca entre todas las plantillas de las que dispone la mejor coincidencia, devolviendo un identificador de la plantilla con mejor ajuste y un grado de certeza.

Fue necesario realizar una serie de modificaciones para adaptar el comportamiento del reconocedor a las necesidades del proyecto:

- Se incluyó una plantilla para cada uno de los números que se iba a tratar de reconocer a partir del trazo del usuario (números del 0 al 9).
- Se modificó el comportamiento del reconocedor ya que originalmente el reconocedor sólo operaba con un trazo.
- Se incluyeron algunas plantillas más para determinados números ya que algunos pueden ser dibujados tanto en uno como en dos trazos (números 4 y 5).
- Se añadieron algunos métodos de consulta y actualización para algunos atributos (*get*/*set*).
- Para el reconocimiento se definió un grado de certeza laxo debido al tipo de usuarios para el que se ha ideado la aplicación. Es decir, se ha priorizado tratar de reconocer qué número ha intentado dibujar el usuario, frente a forzarle a que dicho número esté perfectamente trazado.

3.1.2 Paquete game

En dicho paquete se encuentra toda la lógica del juego. Contiene las siguientes clases de interés:

3.1.2.1 Clase BoundingBox

Esta clase es la encargada de encerrar en un rectángulo los objetos sobre los que se va a operar. Incluye algunos métodos de interés como *calculateBoundingBox(…)* que calcula el rectángulo que encierra a un conjunto de puntos. O como *inScreen(…)* que nos permite saber si un objeto está dentro de los límites de la pantalla. Además métodos como *isInBox(…)* o *isOverBoundingBox(…)* nos indican si un punto u otra "caja" están sobre una *bounding box* respectivamente.

3.1.2.2 Clase MainGame

Es la clase principal de la aplicación, crea una instancia del procesador de gestos (clase *GestureEventManager*) y muestra la pantalla inicial (lienzo en blanco).

3.1.2.3 Clase NumberRendering

Es la clase que representa los números. Tiene asociada la siguiente información: el valor del número que representa y de los dos sumandos que lo han formado (si el número proviene de una operación suma), un color, una bounding box y una imagen del trazo del número.

El método *calculateCentrePoints*() es el encargado de calcular los centros de las círculos que representan el número. Por otro lado, los métodos *move*() y *moveTo*() se encargan de mover el objeto donde corresponda, siempre dentro de los límites de la pantalla.

3.1.2.4 Clase GestureEventManager

Es el procesador de gestos de la aplicación. Contiene una serie de métodos que son llamados automáticamente cada vez que un evento táctil se produce. El método *touchDown*(…) es llamado cada vez que el usuario toca la pantalla , *touchDragged*(…) cuando el usuario desplaza un dedo por la pantalla, y finalmente cuando lo levanta de invoca al método *touchUp*(…). Esta clase además es la encargada de reproducir el sonido que corresponda en función de la operación que se realice.

3.1.3 Paquete screens

En este último paquete se aloja todo lo relativo a la visualización. Se ha creado una clase abstracta que engloba a cada tipo de pantalla empleado en el juego, consiguiendo así una distribución ordenada que permita delimitar las funcionalidades de cada método en cada clase y realizar así modificaciones con facilidad y logrando también cierto grado de escalabilidad.

3.1.3.1 Clase BaseScreen

Es la clase abstracta, de ella extiende *DrawScreen.*

3.1.3.2 Clase TextureToDraw

Esta clase contiene una textura (imagen) a la que se le asocia un grado de transparencia.

3.1.3.3 Clase DrawScreen

Extiende de *BaseScreen*, es la encargada de dibujar en la pantalla todo lo que sea necesario. Incluye el método *render(…)* que se ejecuta automáticamente unas 60 veces por segundo. Es en él en el que se dibuja en pantalla.

3.2 Flujo del programa

A continuación se explica el flujo del programa desde que se recibe un evento hasta que se dibuja en pantalla el resultado de este evento.

Los números que ya han sido reconocidos de almacenan en un *ArrayList*, ocurre lo mismo con las coordenadas del trazo que pueda estar dibujando un usuario.

Solo se permite mover dos objetos de forma simultánea.

3.2.1 Se toca la pantalla: touchDown()

En primer lugar, cuando el usuario toca la pantalla se llama al método *touchdown* que recibe las coordenadas X e Y donde se ha realizado el toque y el número de dedo con el que se ha realizado el mismo (cada toque simultáneo en la pantalla se va numerando sucesivamente desde 0).

En este método se evalúan las siguientes condiciones en el orden en el que aparecen.

3.2.1.1 Las coordenadas del toque corresponden con la *BoundingBox* de algún número ya mostrado.

Se evalúa si es el segundo toque que sobre este objeto en un corto periodo de tiempo. En este caso se asigna una opacidad total al dígito asociado a dicho número para que éste aparezca de nuevo.

Si se trata del primer toque sobre este número, se incrementa un contador y se inicia un temporizador que espera durante un breve periodo de tiempo a un segundo toque. Cuando la espera finaliza sin que se haya producido este segundo evento, el contador se restablece, ya que se interpreta que el usuario quiere mover un objeto a

otra zona de la pantalla (ya sea para sumarlo con un segundo elemento o para ubicarlo en otro lugar).

Si no hay ya dos objetos seleccionados, se recupera el número tocado del *ArrayList* y se almacenan las coordenadas del toque para posteriormente calcular un vector de desplazamiento en caso de que se deslice el dedo por la pantalla. Además se modifica a *true* una variable booleana *moving* que indica que se quiere desplazar un objeto por la pantalla.

3.2.1.2 El toque se ha producido sobre la papelera

Se procede de la misma forma que en el caso anterior con la salvedad de que en caso de que se haya producido un doble toque se pone a *true* una variable booleana *clearNumbers* que indica que se quiere limpiar la pantalla y se eliminan todos los objetos que hubiesen sido reconocidos.

3.2.1.3 El toque no se produce sobre ninguno de los objetos anteriores

El usuario está trazando un número, por tanto almacenamos la coordenada del toque en el *ArrayList* destinado para ello y se crea un temporizador que de momento no realiza ninguna acción.

3.2.2 Se arrastra el dedo por la pantalla: touchDragged()

Cuando el usuario desplaza el dedo por la pantalla se llama al método touchDragged repetidamente. Este método recibe las coordenadas X y Y donde se ha realizado el toque y el número de dedo con el que se ha realizado el mismo (cada toque simultáneo en la pantalla se va numerando sucesivamente desde 0).

3.2.2.1 Se está desplazando un objeto: *moving == true*

El usuario desea reubicar un objeto. Se analiza con qué dedo ha realizado el toque el usuario. Si la variable pointer (número del dedo que realiza el toque) nos indica que el desplazamiento se está realizando con el primer dedo que se colocó sobre la pantalla, se mueve primer objeto que se seleccionó. Para ello utilizando las coordenadas del toque se construye un vector de desplazamiento y se mueve este objeto a la posición resultante.

En el caso de que el dedo corresponda al del segundo toque, se procede de forma análoga pero desplazando el segundo objeto seleccionado.

3.2.2.2 No se está desplazando ningún objeto y se está arrastrando un único dedo por la pantalla

El usuario sigue trazando un número se añade la coordenada al *arraylist* correspondiente, si es necesario se llama a la función *midPoints*() que interpola dos puntos para así obtener un trazo bien definido y de aspecto más natural.

3.2.3 Se levanta el dedo de la pantalla: touchUp

Cuando el usuario levanta el dedo de la pantalla se llama al método *touchUp* que recibe las coordenadas X e Y donde se ha realizado el toque y el número de dedo con el que se ha realizado el mismo (cada toque simultáneo en la pantalla se va numerando sucesivamente desde 0).

3.2.3.1 Se desea limpiar el lienzo *clearNumbers == true*

Se eliminan todos los objetos que pudiese haber en pantalla (con la excepción del icono de la papelera).

3.2.3.2 Se ha producido un toque sobre la papelera o sobre un número

Se incrementa el contador de trazo.

3.2.3.3 No se ha movido ningún objeto: *moving == false*

Si no se ha movido ningún objeto es porque el usuario estaba trazando un número y ha terminado dicho trazo. Se incrementa el contador de trazo, se recuperan los puntos dibujados por el usuario y se envían al reconocedor.

- Si el reconocedor identifica un número con una certeza inferior al umbral definido:
    - Si se trata del primer trazo realizado se permanece a la espera un tiempo por si fuese un número que requiere de dos trazos para ser dibujado.
    - Si ya se trata del segundo trazo, esto significa que el número no ha podido ser reconocido por lo que se limpia el *arraylist* de coordenadas y se espera a que el usuario realice un nuevo toque en la pantalla.

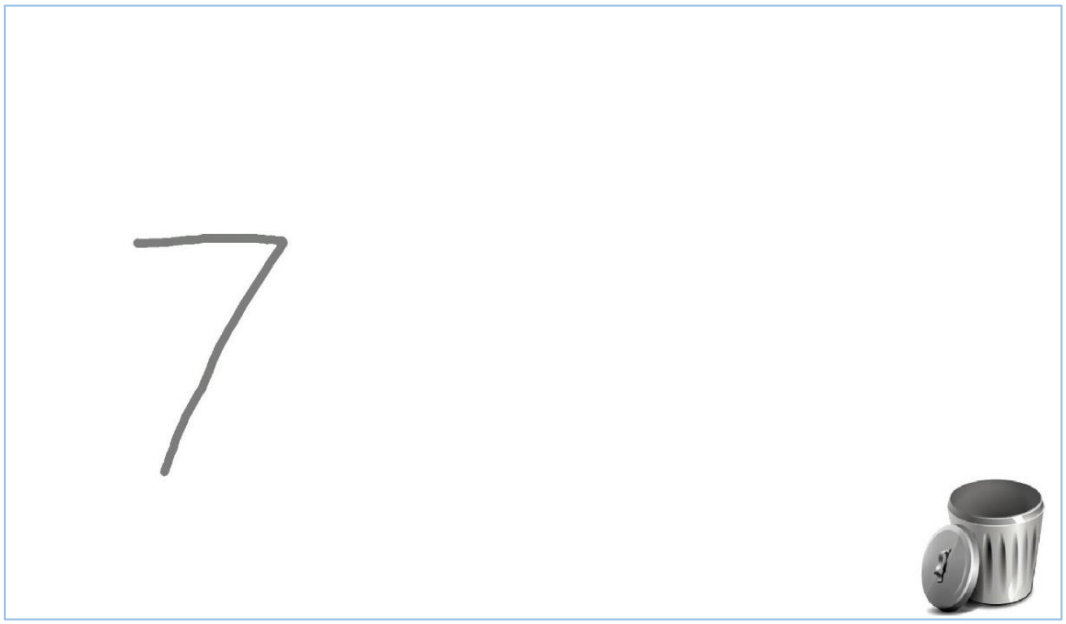

Ilustración 1: Se espera un segundo trazo

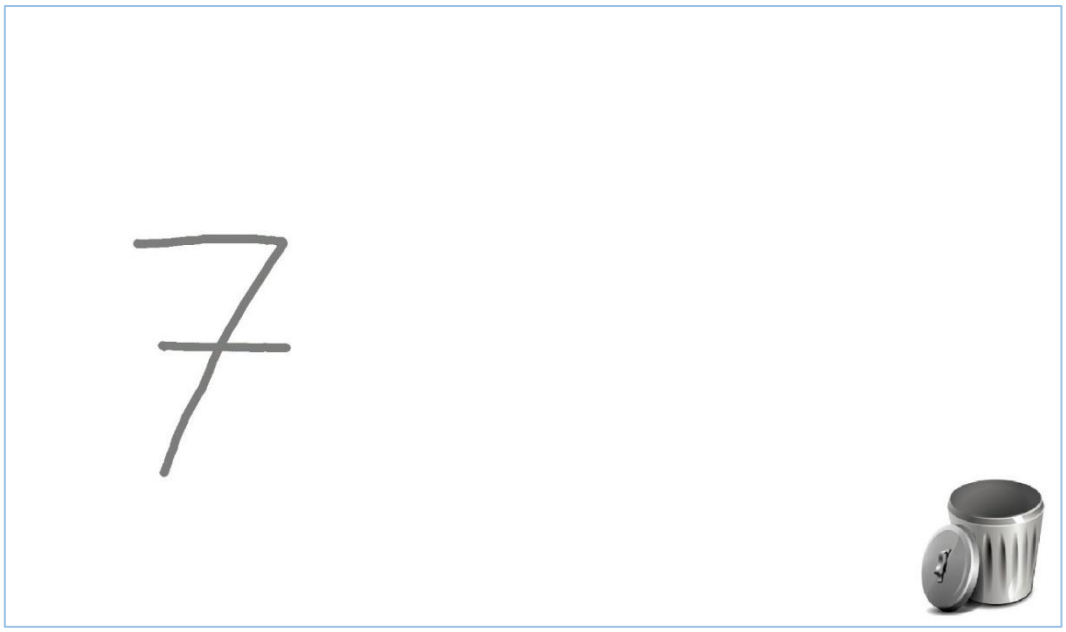

Ilustración 2: Se realiza el segundo trazo

- Si se reconoce un número con un grado de certeza suficiente se añade este nuevo número al *arraylist*. Al ser un número creado mediante un trazo, y no mediante una suma el color de los círculos de la representación, será el color por defecto asignado a dicho número. Como es lógico el número cero no tiene asociada ninguna representación en forma de círculos.

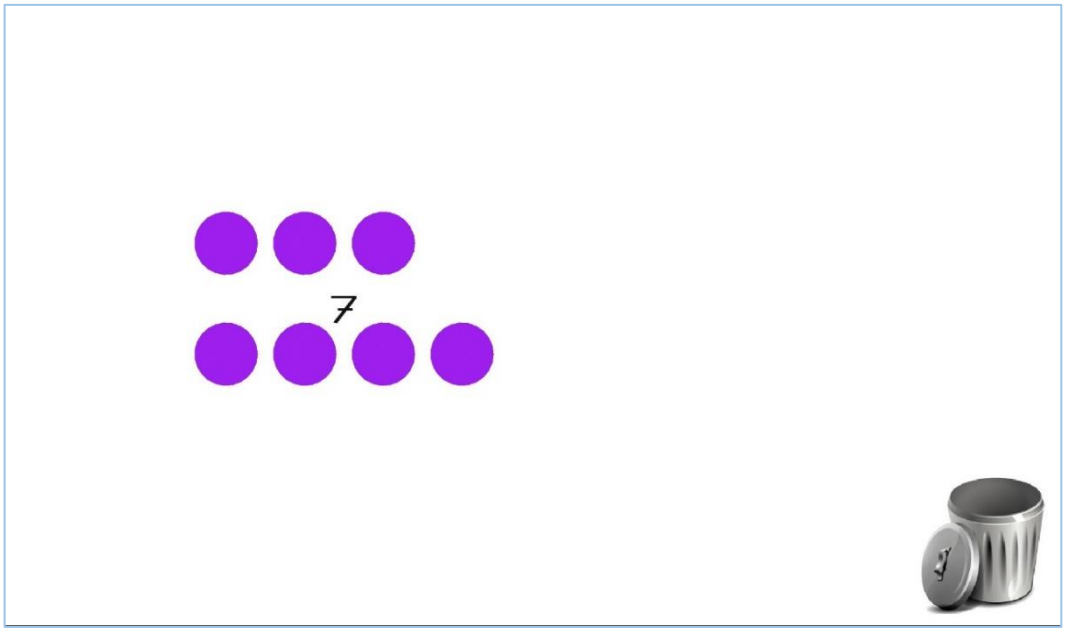

Ilustración 3: Al reconocer un número se crea un objeto que lo represente.

3.2.3.4 Se ha movido algún objeto *moving == true*

Como mínimo un objeto ha sido movido.

- Si el dedo levantado es el primero que se posó sobre la pantalla.
    - Se comprueba si hay un segundo objeto seleccionado, en caso afirmativo y si las bounding boxes se solapan se suman ambos números eliminando estos dos y creando un tercero.

      A la hora de realizar la suma se tienen en cuenta algunos aspectos:

        - Solo se suman números hasta el 20 (inclusive)
        - En general el resultado se muestra manteniendo los colores de los dos sumandos. Es decir, si se crea un 8 (color original marrón) a partir de un 3 (verde) y un 5 (amarillo) la representación de ese 8 será tres círculos verdes y cinco amarillos.
        - Si la suma excede de 10, independientemente de los sumandos que lo formen, se representa el resultado como una suma de diez círculos naranjas (color por defecto del 10) y el resto del color correspondiente.

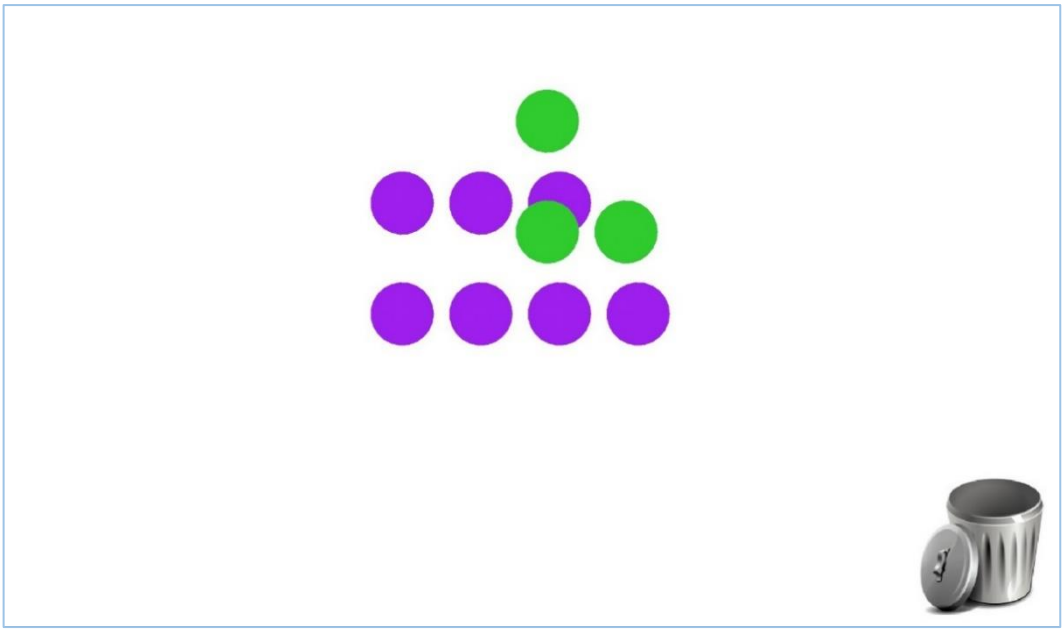

Ilustración 4: Se solapan dos objetos.

    - Si no se ha producido la suma y el objeto seleccionado se solapa con la papelera este objeto es eliminado.

- Si el dedo levantado es el segundo que se posó se procede de la misma forma que en el caso anterior.

Finalmente se limpia el *arraylist* de coordenadas.

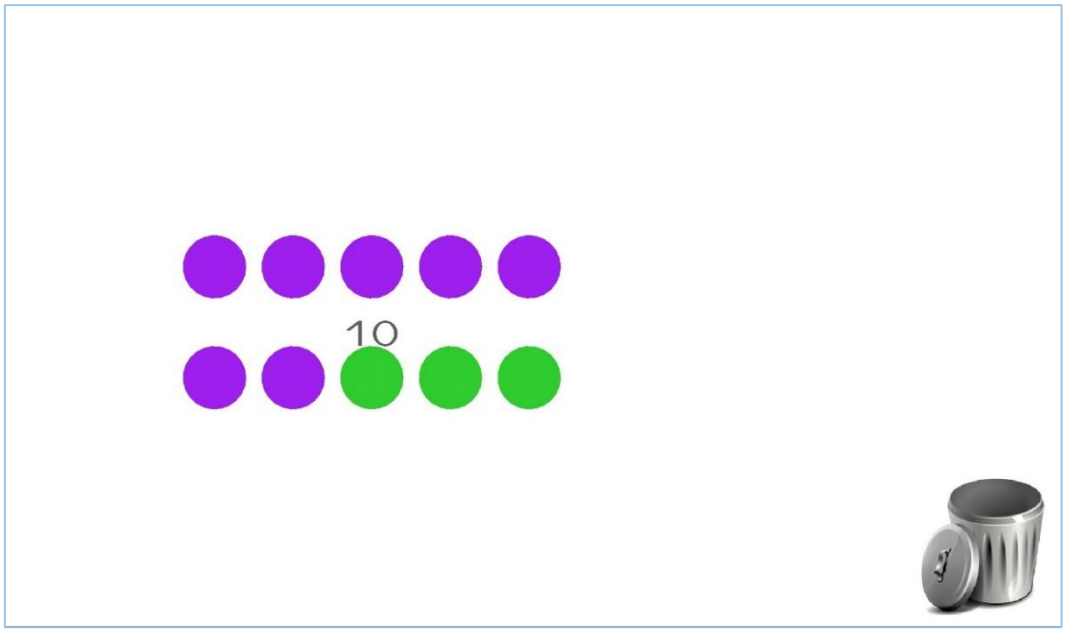

Ilustración 5: Se produce la suma (7 + 3) = 10

3.2.4 Se dibuja en la pantalla render()

El método *render()* es llamado entre 30 y 60 veces por segundo. En él se dibuja todo lo que se desea mostrar por pantalla.

1) En primer lugar se dibuja la papelera.
2) Si se acaba de reconocer un número se dibuja una imagen del dígito trazado en el lugar donde este se realizó. Esta imagen va perdiendo opacidad conforme pasa el tiempo.
3) Se dibuja los puntos almacenados en el *arraylist* de coordenadas.
4) Si la imagen que se estaba dibujando ya es totalmente transparente se pone la variable booleana *drawLast* a *true*. Esta variable permite que el último número que ha sido reconocido no sea dibujado hasta que la imagen de la cifra sea totalmente transparente.
5) Si hay algún número reconocido en el *arraylist* de números se dibuja, siempre teniendo en cuenta el valor de la variable *drawLast.*

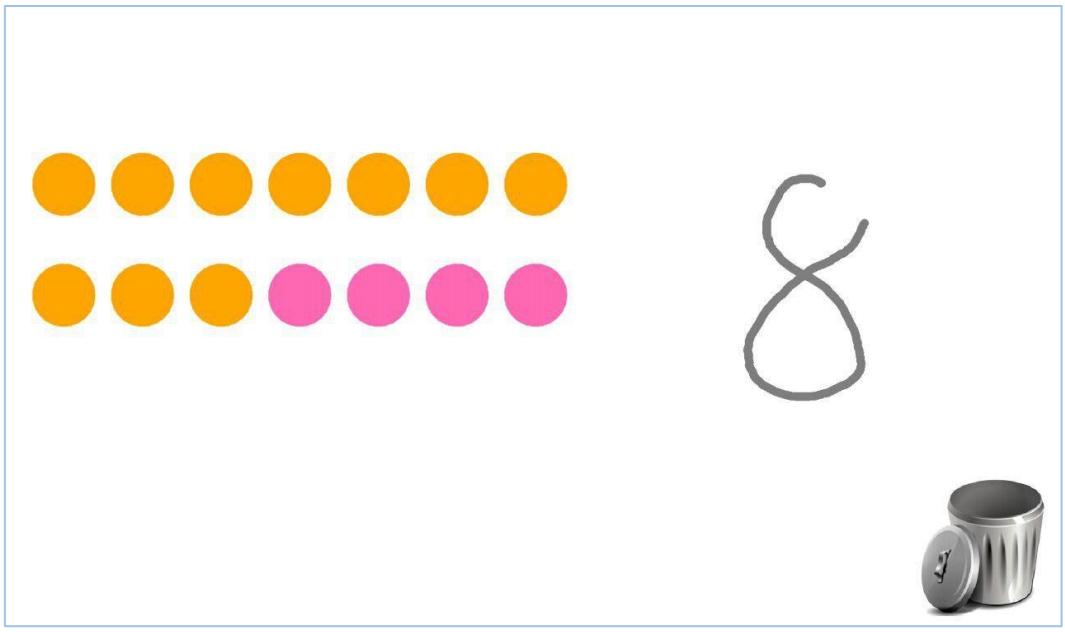

Ilustración 6: Aspecto general de la aplicación

4. PRUEBAS EN CENTROS EDUCATIVOS

El martes 14 de junio de 2016 se realizaron pruebas de campo en el Colegio Irabia-Izaga de Pamplona.

4.1 Objetivos de las pruebas

El objetivo de las pruebas era comprobar si la aplicación desarrollada cumplía un requisito básico de usabilidad. A saber, que el usuario es capaz de realizar la tarea que desea o que se le encomienda en un periodo razonable de tiempo y sin mayores dificultades.

También se buscaba ajustar algunos valores como el umbral del reconocedor o el tiempo de espera entre toques sobre un mismo objeto.

4.2 Pruebas realizadas

Las pruebas fueron realizadas con niños de 5 y 6 años de edad. Por parejas se les presentaba la aplicación, se les sugería que dibujasen un número (del cero al diez) y se observaba su reacción. Algunos enseguida continuaban tratando de dibujar el resto de números.

Tras esto se les dejaba que explorasen la aplicación y que fuesen descubriendo las funcionalidades por sí mismos. Con esto se trataba de comprobar si la aplicación era lo suficientemente intuitiva o si iba a ser necesario realizar algún tipo de modificación sobre las operaciones.

Si en algún momento en niño se atascaba o se ofuscaba, bien porque no sabía qué hacer, bien porque el reconocedor no identificaba el número que trataba de dibujar, enseguida se le reconducía sugiriéndole que trazase algún otro número o mostrándole como sumar dos conjuntos.

Durante la prueba se observaron los siguientes hechos:

- La aplicación no reconocía todos los números de todos los niños que probaron la aplicación. Esta observación sugiere añadir más plantillas para algunos de los números.
- En alguna ocasión el niño dibujaba un 4 y el reconocedor identificaba un 5.
- Varios de los niños descubrieron por sí mismos que si se arrastra un conjunto sobre la papelera éste era eliminado.
- Ninguno de los niños descubrió que ocurría si realizabas un doble toque sobre la papelera (se limpia la pantalla).

- Algunos de los niños expresaron su deseo de poder sumar números mayores que el 20.

- ¿Por qué no suma? Una de las parejas situó dos conjuntos de forma próxima y en el espacio que quedaba entre ambos conjuntos dibujó un signo más (+). Al ver que no se producía la suma preguntaron por qué no ocurría lo que ellos esperaban.

4.3 Conclusiones

Tras la realización de las pruebas fue posible extraer una serie de conclusiones:

- La idea gusta. Los niños acogieron la aplicación con interés y muchos de ellos exploraban la aplicación por sí mismos sin necesidad de ser guiados.

- Despertaba interés. Los usuarios deseaban conocer todas las funcionalidades que ofrecía la aplicación y deseaban probarlas, si no eran capaces de conseguir algo no dudaban en preguntarlo.

- El error relativo al reconocimiento del 4 y el 5 puede deberse a un grado de certeza demasiado bajo. Un umbral más exigente junto a un mayor número de plantillas para estos números es probable que solucionase el problema.

- Comprendían el funcionamiento de la aplicación. Los niños eran capaces de completar las actividades que se les sugería en un tiempo más que razonable y sin apenas dificultades.

- El hecho de que los niños no descubriesen la funcionalidad del doble toque sobre la papelera sugiere que se debería incluir una breve guía, muy visual, dentro de la propia aplicación. En ella deberían mostrarse todas las funcionalidades de la aplicación.

- Refuerzo del conteo. Cuando los niños formaban números mayores que diez y eran preguntados sobre qué número era contaban los círculos que aparecían en la pantalla. Esto confirma que la aplicación sirve como herramienta que refuerza el conteo.

- Fomenta el compañerismo. Otro aspecto muy positivo que se extrajo de las pruebas es que al realizarla por parejas si uno de los dos niños tenía algún tipo de dificultad para realizar una tarea el otro no dudaba en ayudarle.

- Fomenta el pensamiento razonado.

    - A una de las parejas la aplicación no era capaz de reconocerles el número 7 y uno de los miembros dijo: “Si no funciona el 7 hacemos un 6 y le sumamos el 1”. Esto sugiere que la aplicación fomenta el pensamiento razonado (que se está desarrollando en esta etapa según Piaget) lo cual resulta muy interesante. No debemos olvidar que esta aplicación fue desarrollada con fines educativos.

    - El hecho de que los niños tratasen de sumar dos conjuntos dibujando el operador de suma, es otro indicio más de que la aplicación despierta interés y fomenta el pensamiento razonado de los niños. Esta idea se considera un razonamiento muy válido y no se descarta que sea incluido en futuras versiones de la aplicación.

En definitiva se puede decir que la aplicación tuvo una buena acogida entre los que serían los usuarios finales de la misma y que introduciendo algunas mejoras se puede acabar convirtiendo en una herramienta de interés dentro del ámbito para el que ha sido ideada.

4.4 Modificaciones tras las pruebas

Debido a la proximidad de la fecha de entrega no se pudieron realizar todas las modificaciones deseadas. Se reajustó el umbral de reconocimiento y se añadieron algunas plantillas para los números que resultaron conflictivos en las pruebas.

Sigue sin ser posible sumar números mayores que 20 ya que esto requería un redimensionamiento de los conjuntos presentes en la pantalla (inviable en tan poco tiempo) ya que no se considera interesante que la aplicación tenga *scroll*.

5. CONCLUSIONES Y TRABAJO FUTURO

5.1 Conclusiones

Tras la realización del proyecto es posible sacar varias conclusiones.

Se considera que se han adquirido los conocimientos presentados en los objetivos del proyecto y que además se ha logrado en un periodo razonable de tiempo.

Poniendo en práctica estos conocimientos se ha logrado desarrollar una aplicación estable, funcional, útil, que satisface las necesidades que se deseaba cubrir y que se aproxima mucho a la idea original. Todo ello indica que se realizó un análisis y un diseño correcto que han permitido que el producto final no haya sufrido grandes desviaciones respecto a lo planeado originalmente.

Además el desarrollo de esta aplicación ha supuesto la oportunidad de realizar el ciclo de vida completo de un proyecto, desde los estadios más prematuros, hasta las pruebas con el usuario final y la retrospectiva correspondiente. Por todo esto ha resultado una experiencia enriquecedora en numerosos aspectos.

5.2 Trabajo futuro

Una vez concluido el proyecto quedan abiertas algunas líneas de trabajo, las más interesantes son:

- Incluir la posibilidad de sumar dos conjuntos dibujando un signo más (+) entre ambos.
- Incluir una guía de uso dentro de la propia aplicación.
- Introducir una operación, que realizando una sección sobre un conjunto A, divida a éste dando lugar a dos conjuntos independientes, B y C, de forma que B + C = A.
- Añadir todavía más plantillas para asegurar que se reconoce el máximo de números diferentes.
- Eliminar la restricción que impide sumar números que obtengan un resultado superior a veinte.
- Añadir un botón que active/desactive el sonido.

6. REFERENCIAS Y BIBLIOGRAFÍA

1. Android Studio: Manual de usuario
2. Libgdx: Manual de usuario
3. Piaget, Jean, Psicología el niño. Editorial Morata, 2000.
4. Teoría de las inteligencias multiples (psicologiaymente.net)
5. $1 Unistroke Reconigzer: University of Washington (Jacob O. Wobbrock, Andrew D. Wilson, Yang Li)
6. $1 Unistroke Recognizer: Java implementation by Alex Olwal (github).

7. ANEXO I: Guía de usuario

7.1 Inicio de la aplicación

Al arrancar, la aplicación muestra un lienzo en blanco y queda a la espera de que el usuario escriba un número (*ilustración 1*).

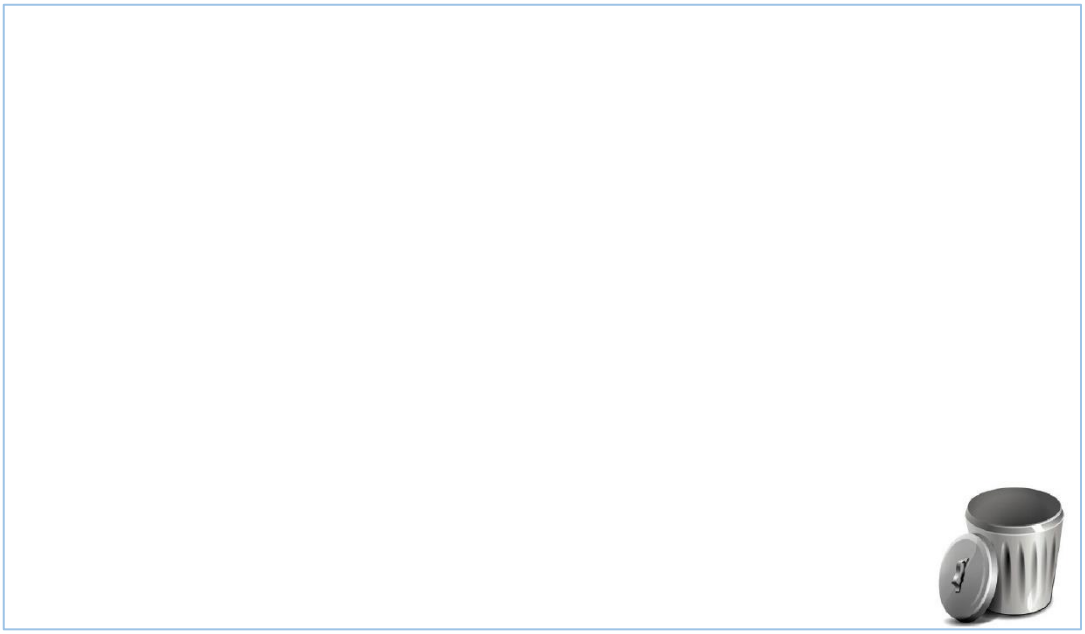

Ilustración 1: Inicio de la aplicación

7.2 Escribir números

Es posible escribir números comprendidos entre el 0 y el 9 (ambos inclusive). Se puede comenzar a realizar un trazo en cualquier parte de la pantalla y en cualquier orientación. Una vez terminado, se muestra una representación de la cantidad correspondiente (*ilustraciones 2* y *3*).

Si la aplicación no reconoce el número, realizar un toque sobre la pantalla para eliminar el trazo y volver a escribirlo. Es necesario tener presente que números como el 4 o el 5 pueden ser dibujados en uno o dos trazos y otros como el 7 únicamente es posible dibujarlo en dos.

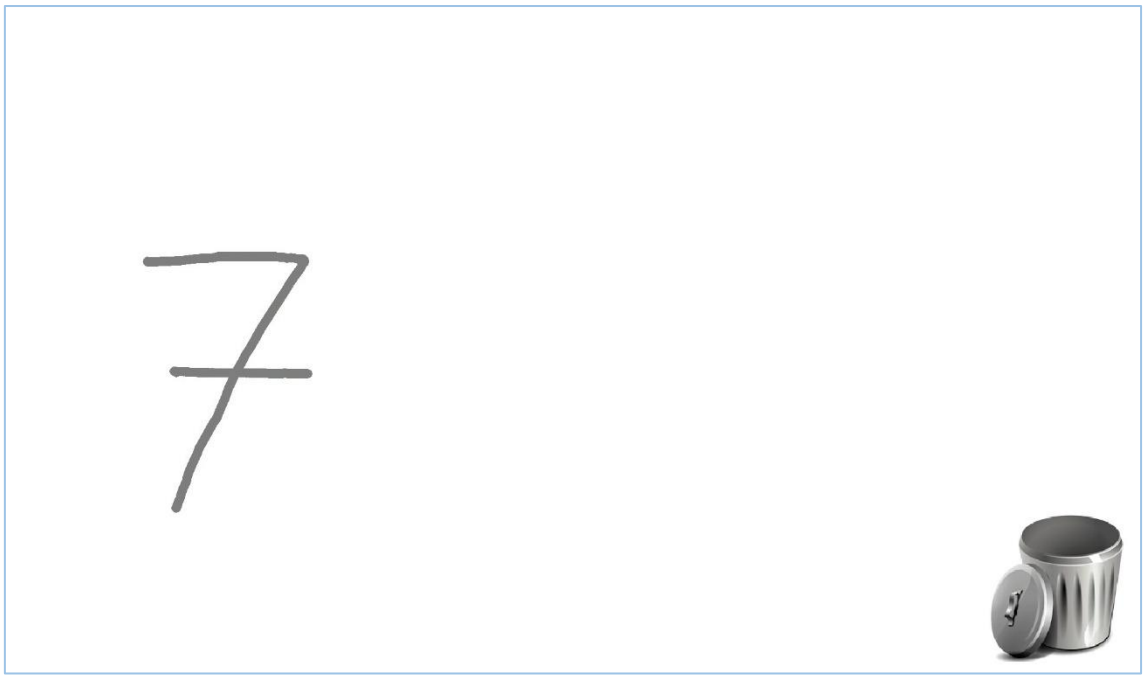

Ilustración 2: Escritura del número

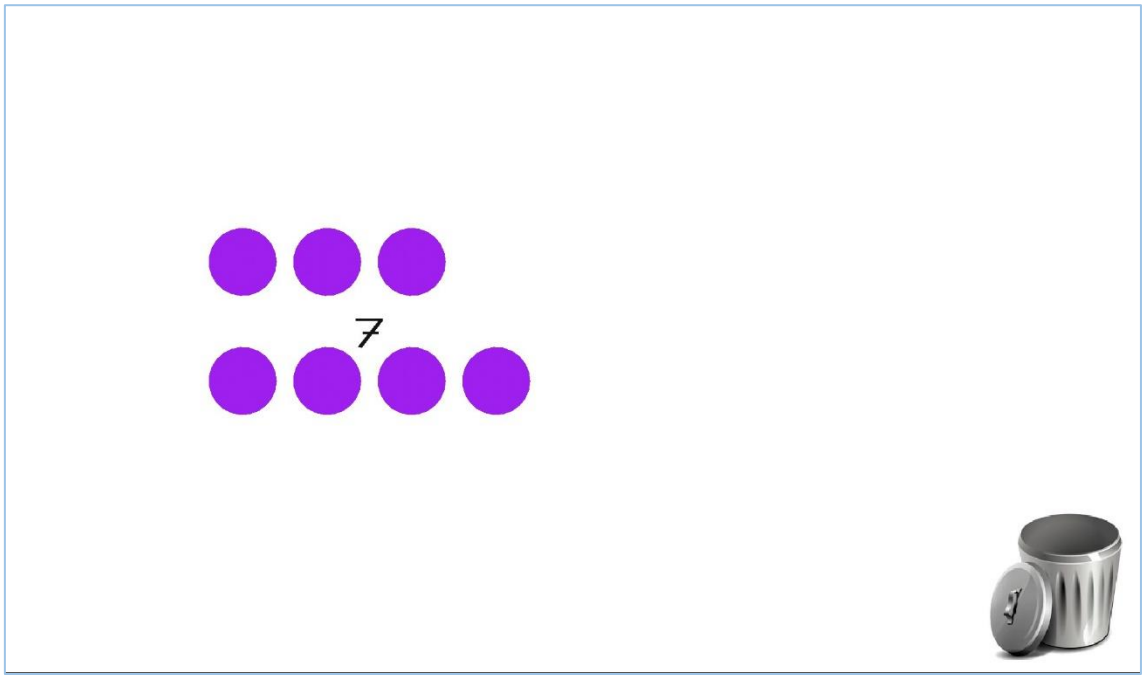

Ilustración 3: Representación que aparece tras la escritura del número

7.3 Operaciones permitidas

Una vez se tiene algún conjunto en pantalla se puede interactuar con él de diferentes formas:

7.3.1 Escribir otro número

Siguiendo el procedimiento anterior, se puede escribir otro número y que ambos permanezcan en la pantalla *(ilustraciones 4 y 5)*.

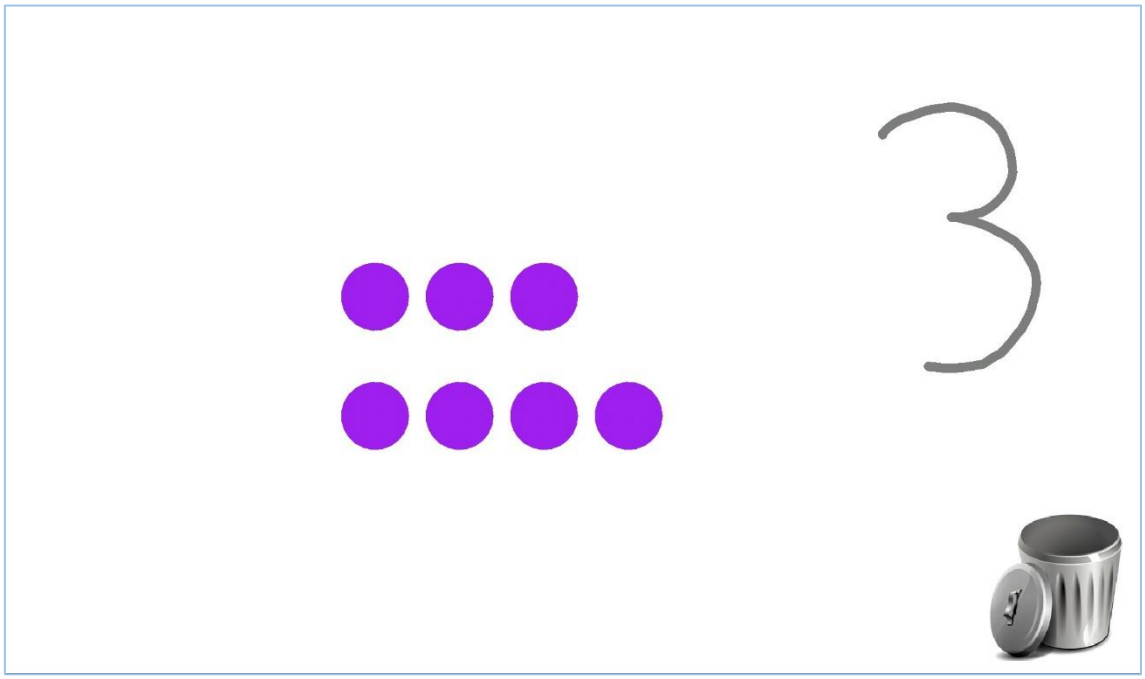

Ilustración 4: Escritura de un segundo número

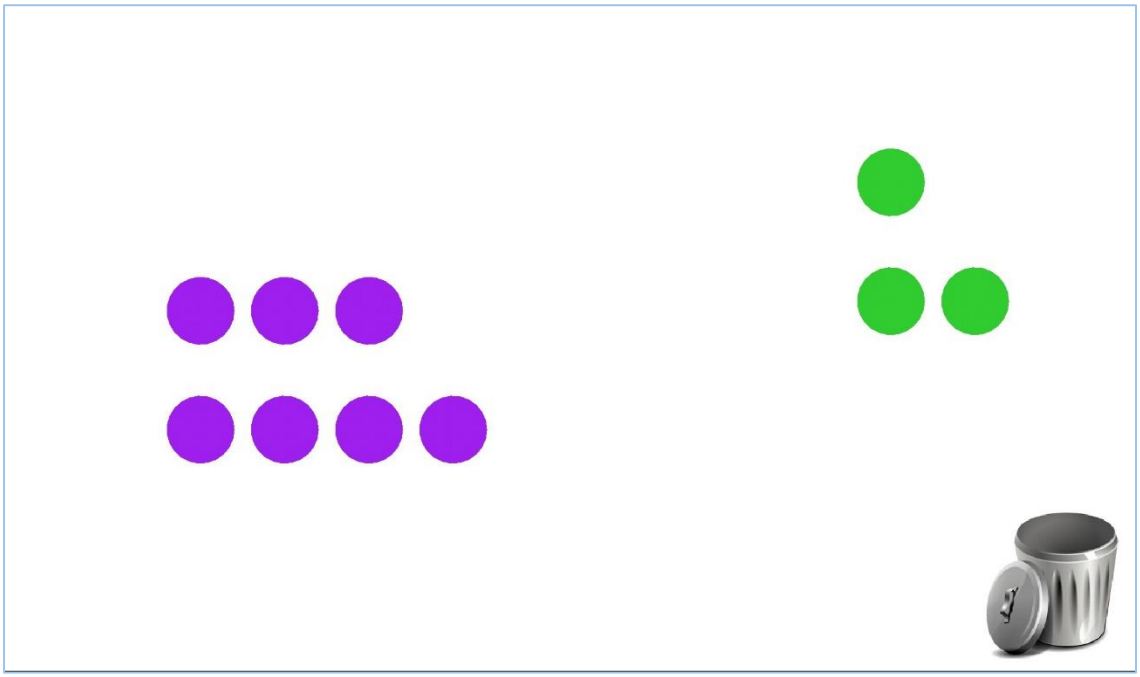

Ilustración 5: El número es reconocido

7.3.2 Eliminar uno o varios conjuntos

Es posible eliminar el conjunto que se desee arrastrándolo al cubo de la basura que aparece en la esquina inferior derecha *(ilustración 6)*.

Para eliminar todos los conjuntos de la pantalla tocar la papelera dos veces rápidamente.

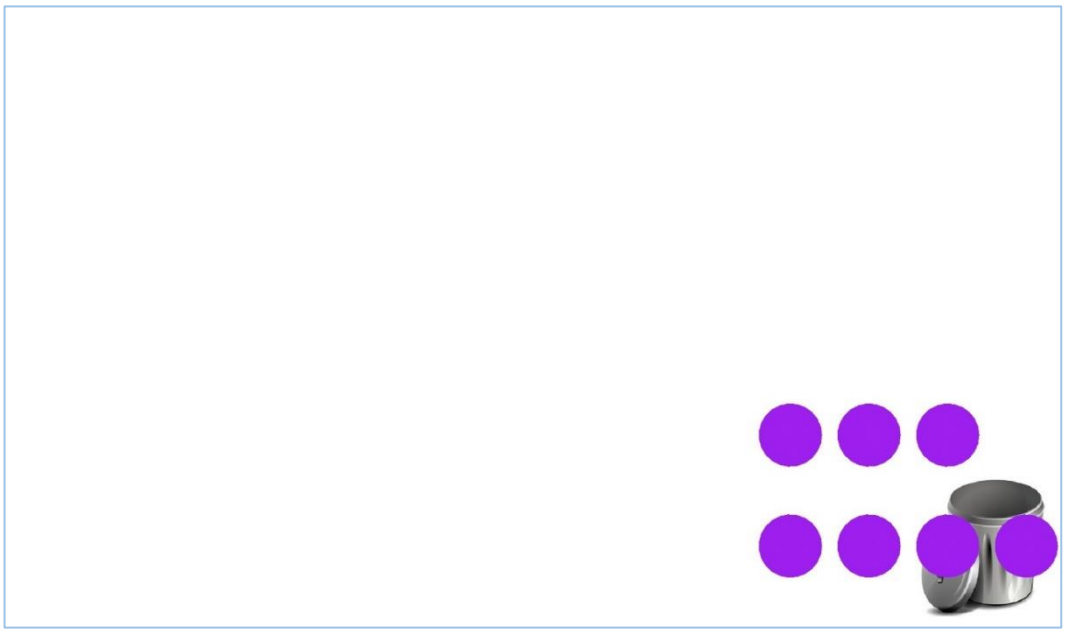

Ilustración 6: Se elimina el conjunto deseado

7.3.3 Recordar número

Cuando aparece un conjunto, se muestra en el centro del mismo el número que representa, aunque se va difuminando con el tiempo. Para que vuelva a aparecer, únicamente es necesario realizar un doble toque sobre el conjunto.

7.3.4 Sumar/Unión

Una vez que tenemos más de un conjunto en pantalla, es posible juntarlos (sumarlos). Para ello tan solo hay que mover los dos conjuntos de forma simultánea (con dos dedos) y hacer que se solapen (*Ilustración 7*).

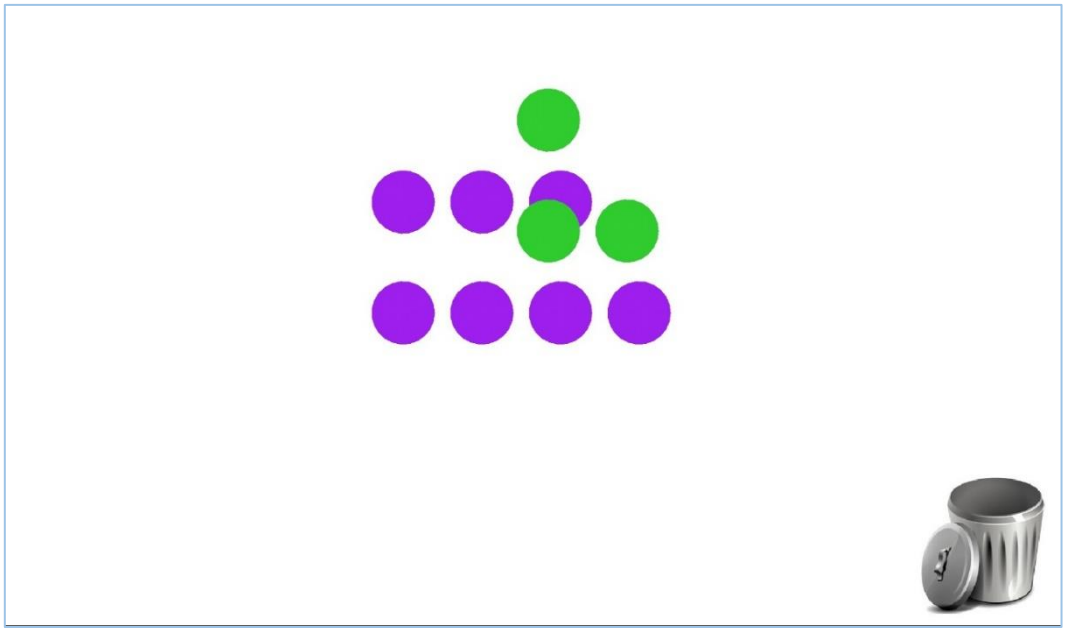

Ilustración 7: Conjuntos solapados

Al levantar los dedos se forma el conjunto resultante de la suma (*Ilustración 8*).

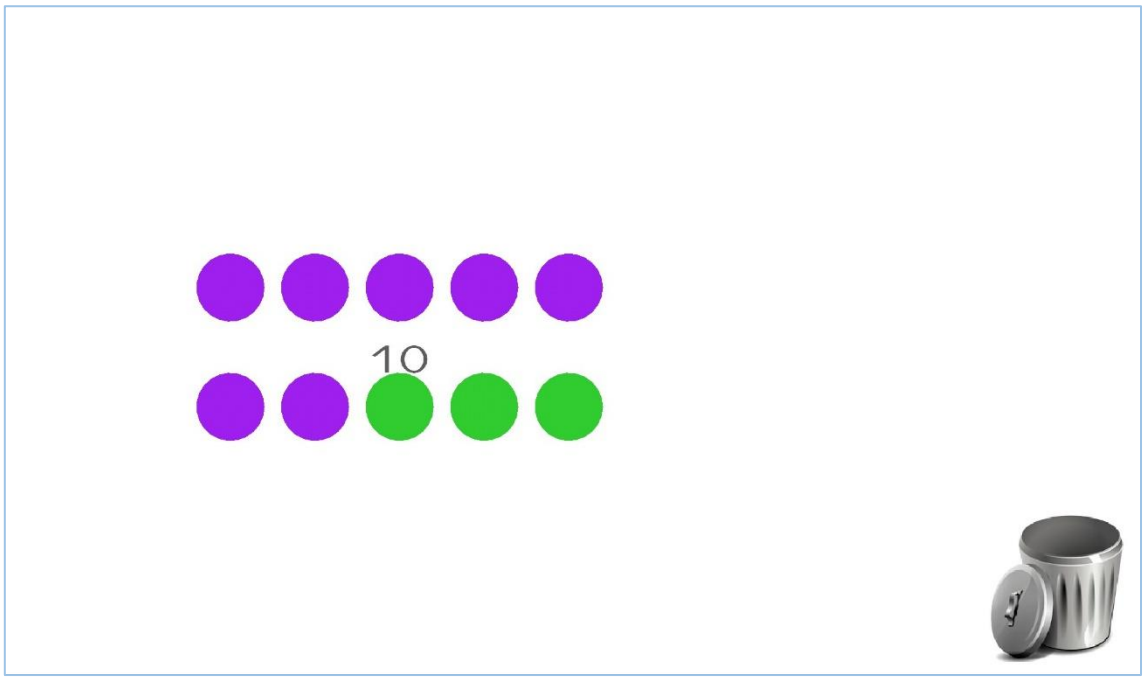

Ilustración 8: Conjunto resultante

Al realizar la suma es aconsejable tener en cuenta lo siguiente:

- El resultado se muestra manteniendo los colores de los dos sumandos. Es decir, si se crea un 8 (color original marrón) a partir de un 3 (verde) y un 5 (amarillo) la representación de ese 8 será tres círculos verdes y cinco amarillos.
- Si la suma excede de 10, independientemente de los sumandos que lo formen, se representa el resultado como una suma de diez círculos naranjas (color por defecto del 10) y el resto con el color correspondiente.
- Solo se suman números hasta el 20 (inclusive).

El comportamiento del conjunto resultante es el mismo que el de los anteriores, por lo que se le puede sumar otra cantidad obteniendo un nuevo resultado (*Ilustraciones 9 y 10*).

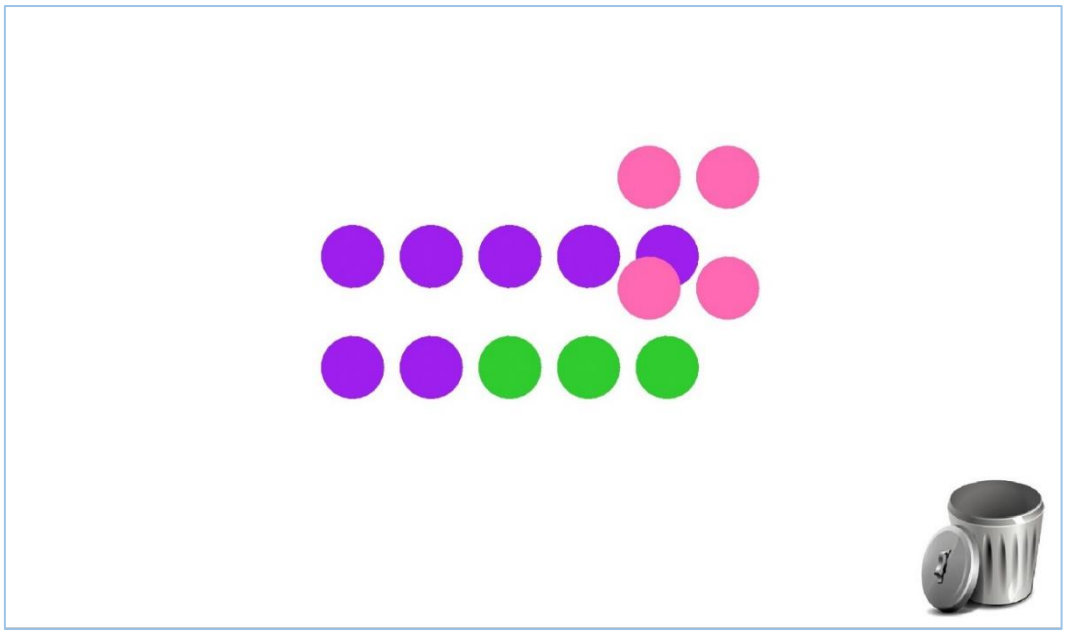

Ilustración 9: Se suma un nuevo conjunto

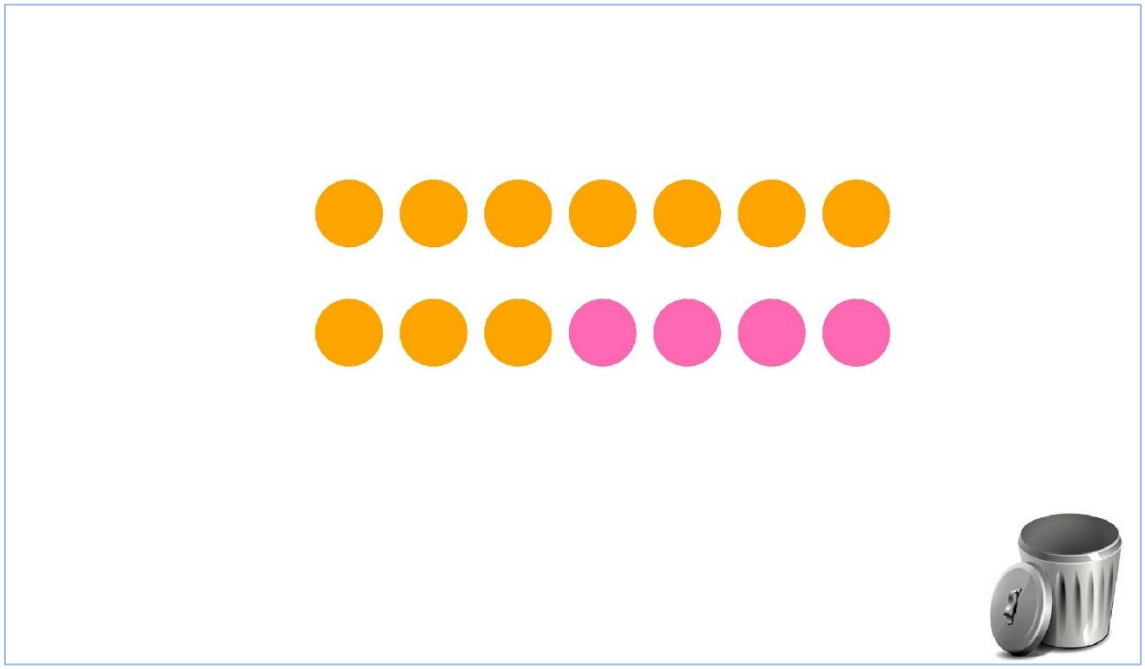

Ilustración 10: Se representa el resultado de la suma