CS3483 Multimodal Interface Design

Project Final Report - Languaid

Content

Monday Group 2	
Name	SID
55777188	Michael Kurzewski
55797243	Barshon Clinton Sarkar
55551624	Leung Wai Keung
40137023	Eliott Haurat
55742987	Rai Phirens

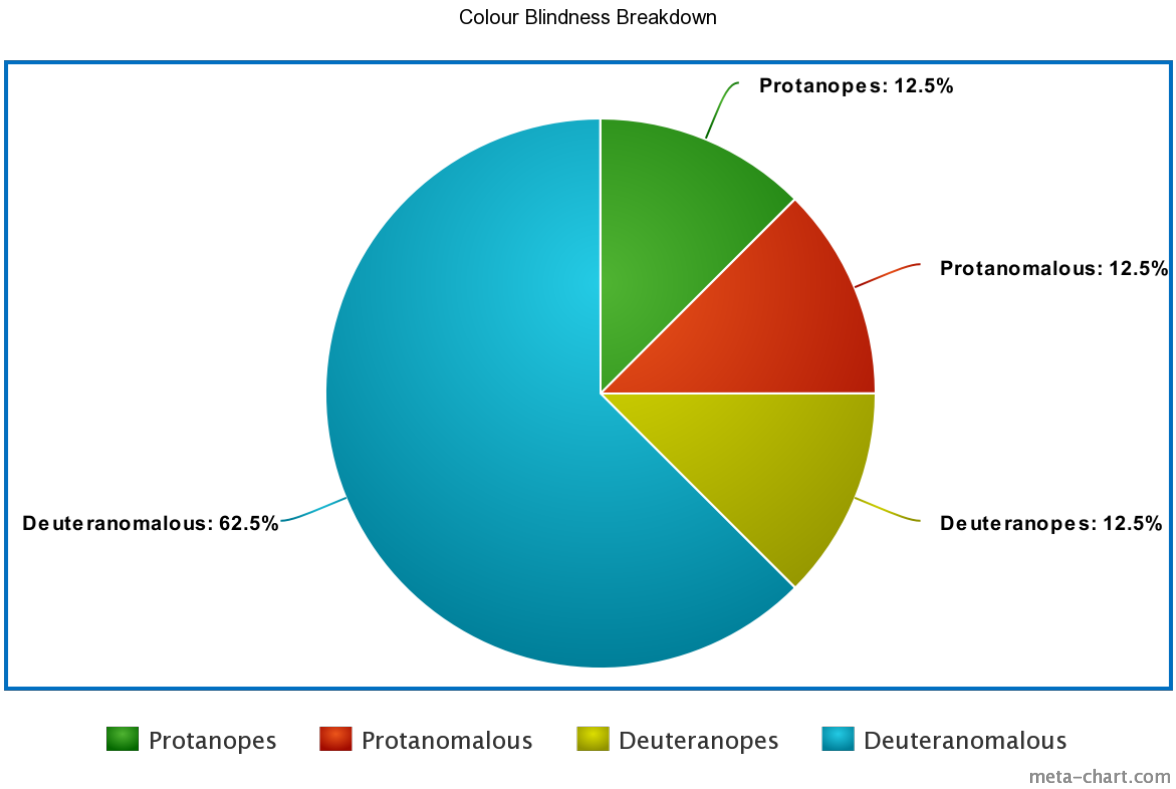
1. User community outline

Our application is primarily aimed at English users who want to learn Chinese. In recent years, the number of people learning Chinese, and in particular Mandarin as a second language, has increased and has even become the second most spoken language in Australia. Nowadays, Mandarin is the second most spoken language in the world with about 1.117 billion speakers [1]. Most people have complained about how hard it is to learn a new language. In Europe, about 60% of the population is proficient in two or more languages, but in the US, only 20% can speak two or more languages, which is extremely low [2]. Our goal is to make learning a new language easier and accessible to all. Most people feel that learning a new language on their own is not interactive enough, that it is boring, and that the user is not active enough overall. A common method to help people, especially children, learn vocabulary is to use flashcards that they can touch and play with. However, this method is not really suitable for adults and cannot be scaled to allow multiple people to participate at the same time.

Many people struggle to learn languages, which means they have not been able to find the right learning technique or method. From online courses to watching movies in the target language, there are a variety of methods to learn a new language, but for us, one important thing was missing, and that was having fun, playing, and enjoying ourselves while learning, just like we did when learning our native language. The goal of our app is to help people learn new words by forcing them to interact with them so that they are active participants in their learning. We want individuals to be active participants in their learning experience, not just passively watching or listening and not memorizing anything. Today, it can cost up to $80,000 for a monolingual English speaker to learn Mandarin [3]. One reason Mandarin is so expensive to learn is that the intonation is difficult to master and therefore requires much more instruction. According to some estimates, it can take more than two thousand hours to master the vocabulary, tones, and characters.

There are several factors we need to consider in the design when it comes to the people who use our application. For example, people of any age group might want to use our application, which means we need to adapt for each case. For example, we could let the user decide on the font size of the words. After research, we found that the ideal default value is 16 pixels [4], which we will choose for our application, but it can be changed by the user if they want to increase or decrease it. Another

factor to consider is that due to the large number of people who want to learn a new language, there is a high probability that there will be users with some kind of disability, such as visual impairment or color blindness. Currently, there are about 300 million people worldwide who suffer from color blindness [5]. Therefore, it is important to make our application suitable for these people. The most common types of color blindness are those in the red-green category. In the colorblind population, deuteranomalous (or green-weak) vision is by far the most common.

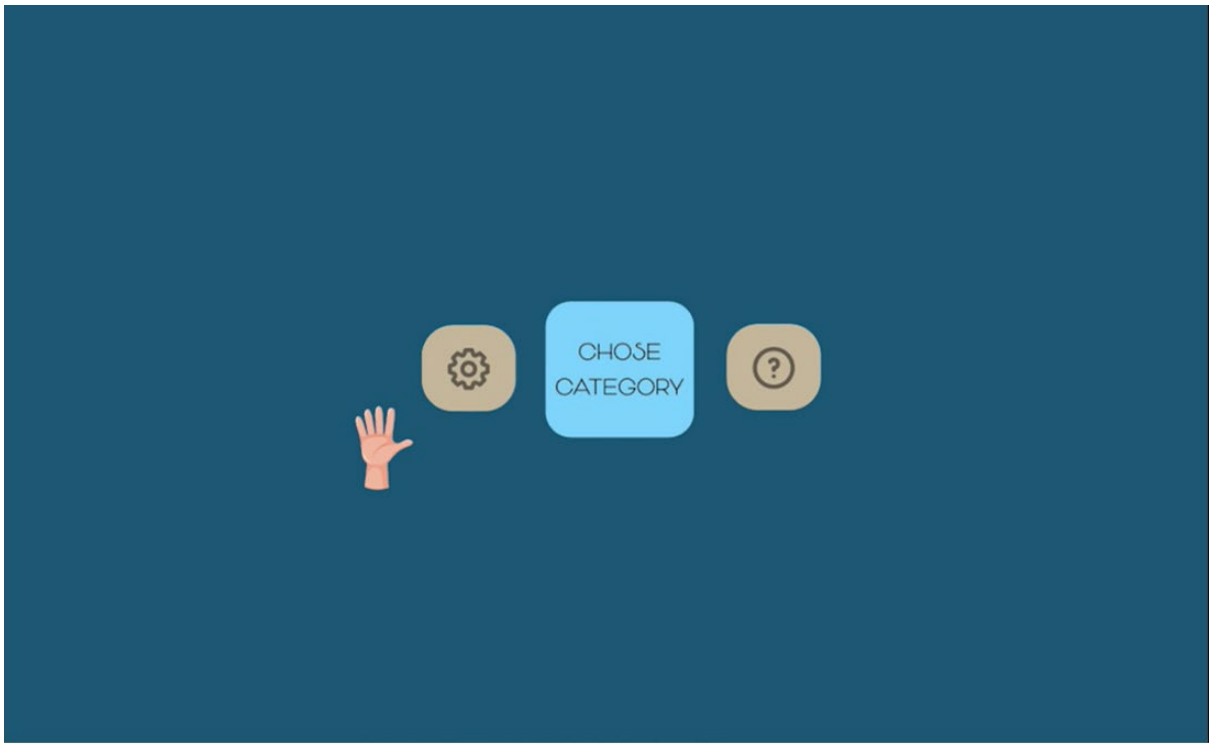


These users should have no difficulty using our application.

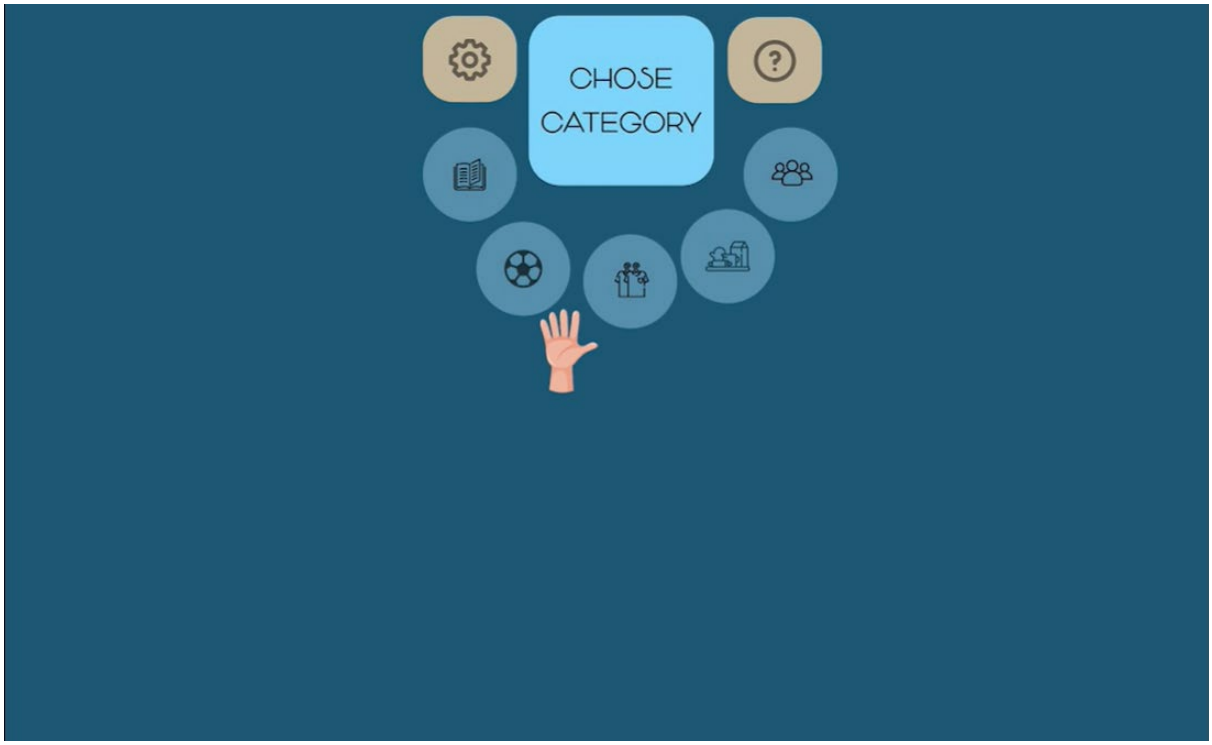
Overall, our goal is for our users to achieve their goal of learning a new language and for our application to help as many people as possible without excluding anyone.

2. Presentation of the final Interface

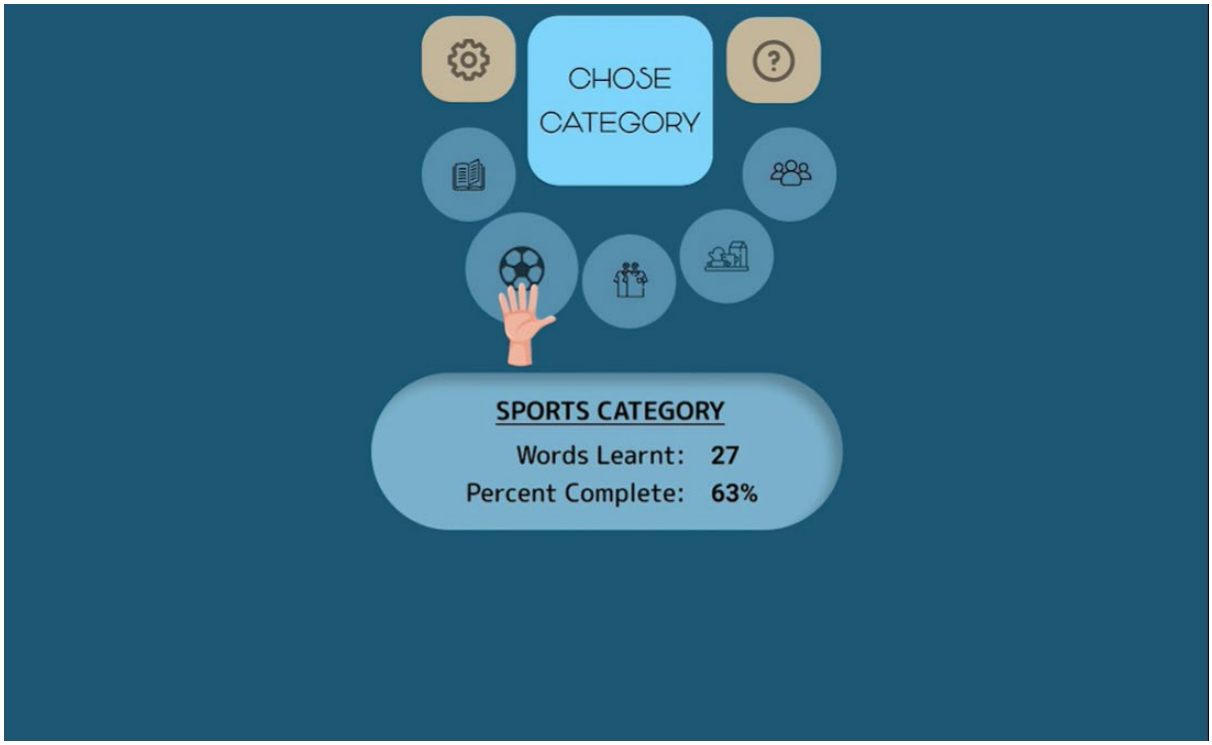
The **Welcome Menu** consists of three buttons. The **Play**, **Settings**, and **Help** buttons. It is a clean and clear design with minimum options so that the user does not get overwhelmed with choices when they start the application. The name of the app is prominent and positioned at the centre of the screen. The buttons are originally faded, but when the user hovers over one of them it becomes more opaque and the size of the button increases. This main screen sets the user's understanding of how to interact with the application as they can observe what the buttons look like and how they react when they interact with them. This is a form of feedback that we implemented to ensure user's immersion with the interactions. We have implemented this feedback design throughout the entire application.



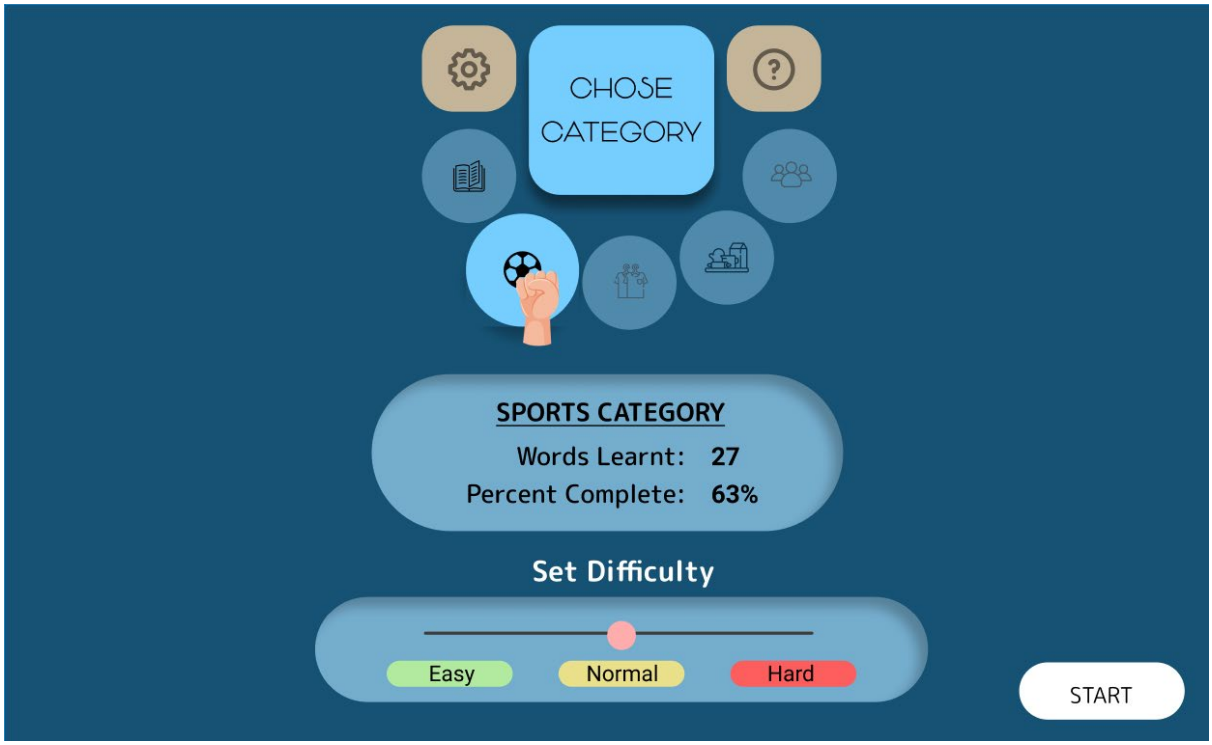
When the **Play** button is selected on the **Welcome Menu**, the **Play** button changes to the **Choose Category** button. The app title is removed here, and there is more space between the **Play**, **Settings**, and **Help** buttons. These buttons are now positioned at the centre of the screen.



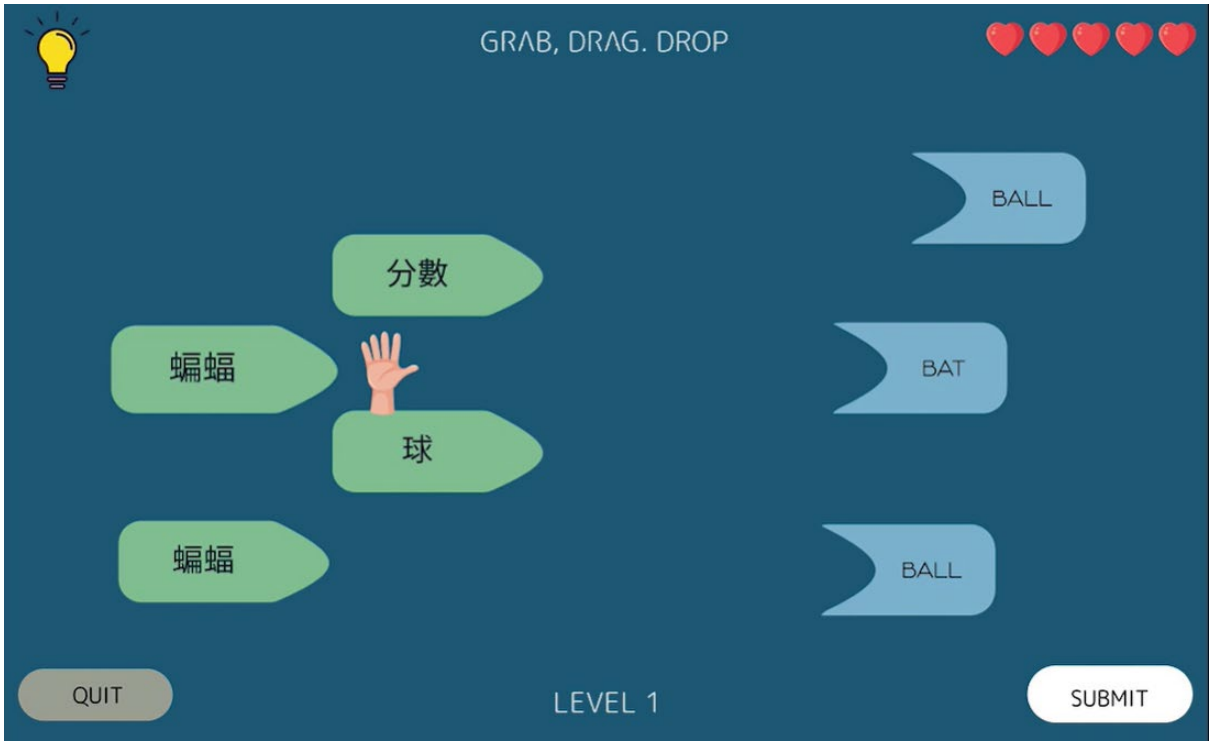
If the user chooses the **Choose Category**, the three buttons animate upwards towards the top of the screen. Meanwhile, the categories pop out underneath the **Choose Category** button. The current form of the page can be identified as the **Category Menu**. The page can be divided into three main blocks: the category block, where the user can select the desired category for their gameplay. The users still have the freedom to directly choose **Settings** or **Help** buttons in this block.



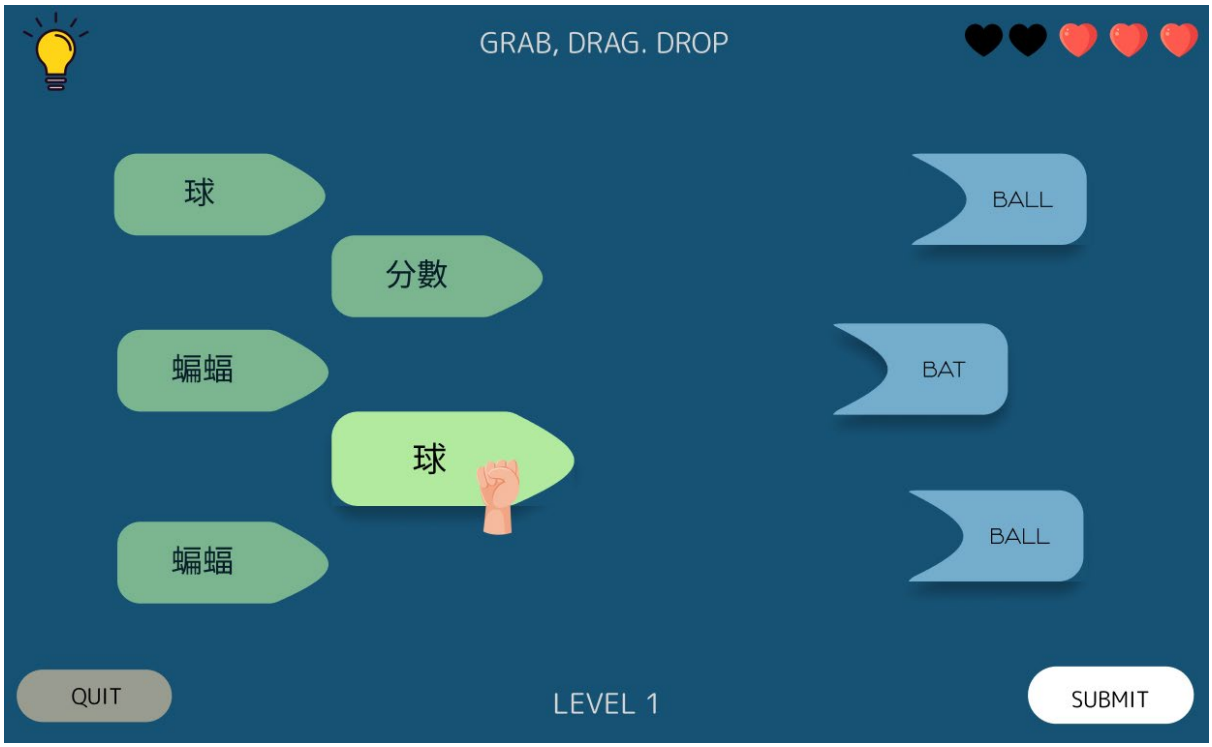
We then have the information block that helps the user track their progression such as the number of words they have learned in the category which they have selected and also the percentage of words learnt compared to the total number of words. This block appears as the user hovers over that specific category's button.



The third block appears when the category is selected and it has the **Set Difficulty** slider letting the user decide how hard they want the levels to be. This block also contains the **Start** button in the bottom right of the screen; it is used to initiate the gameplay session.

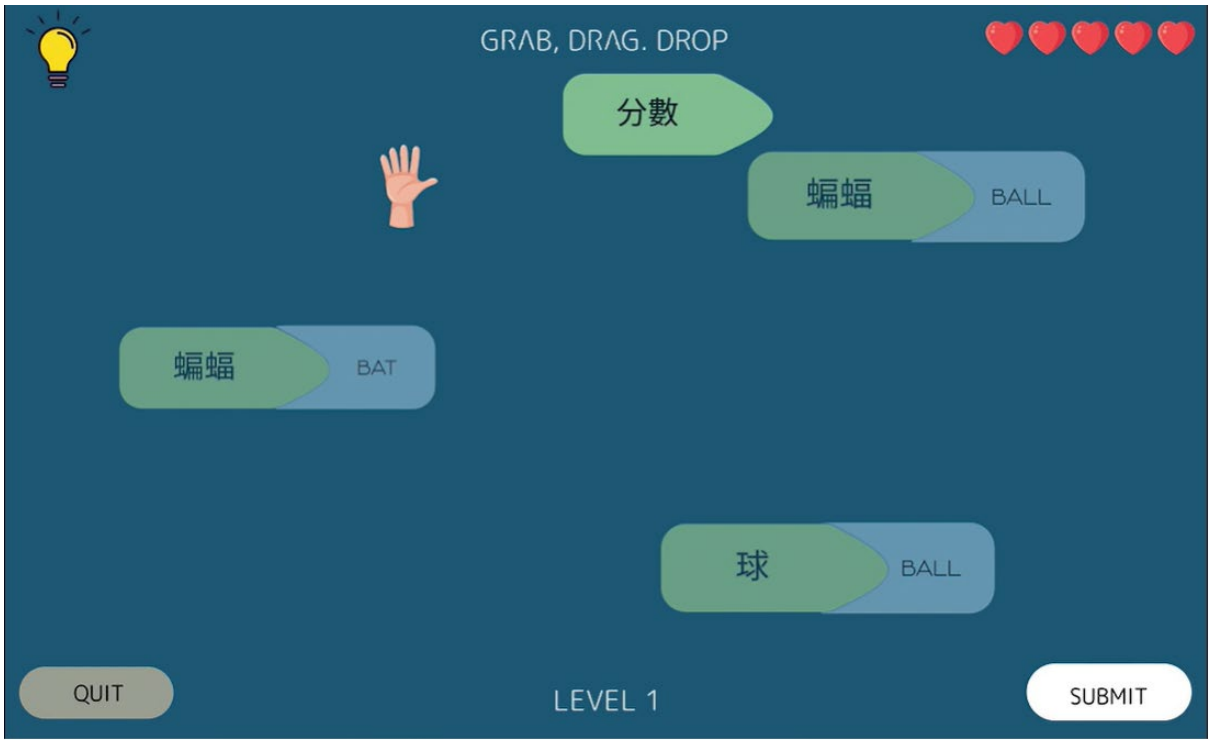


Clicking the **Start** button takes the user to the **Game Screen**. Here, the basic **instructions** to play the game are displayed at the top centre of the screen, the **level** you are on is displayed at the bottom centre of the screen, the **hint** icon (a bright yellow bulb) is on the top left corner, the number of **lives** (will be discussed in details in Game Mechanics section) is on the top right corner, the **Quit** button is on the bottom left corner, and finally the **Submit** button is on the bottom right corner. At the centre area of the screen, the interactable word puzzle **pieces** are present. Users can drag and drop these **pieces** to match with its counterpart.



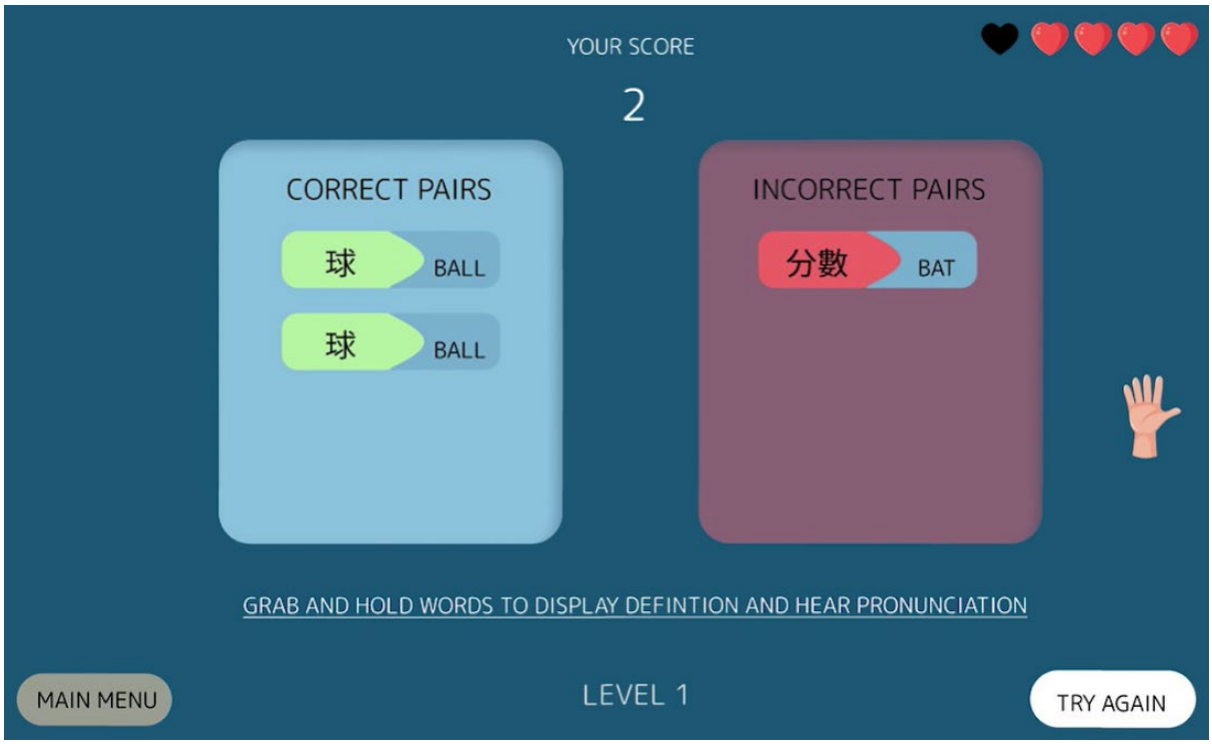
When the **pieces** are selected, they have the previously mentioned feedback. They are enlarged,

become brighter, and more prominent to the user's eyes.

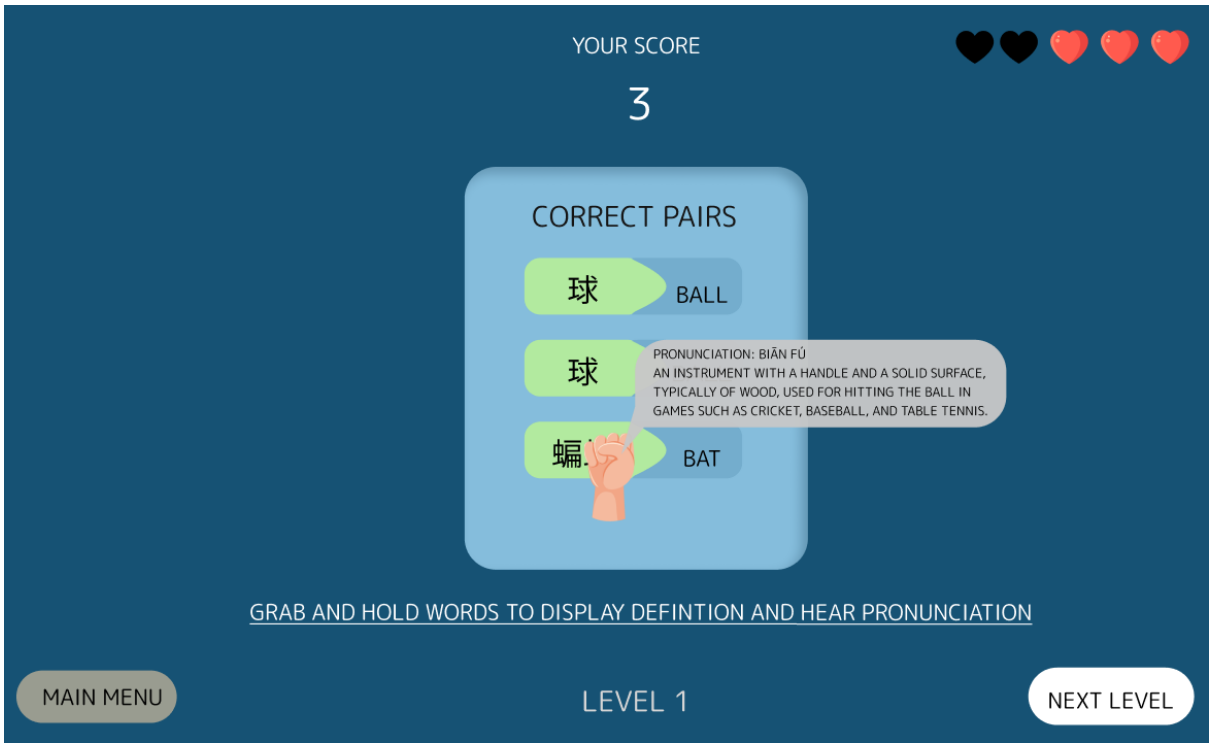


The matched **pieces** become faded and less prominent to the users. However, these **pieces** can still be unmatched and matched with others.

Users have the option to see a hint if they hover over the **hint** icon and select it with the appropriate gesture. After the user is done matching and wants to check their answers, pressing **Submit** will take them to the **End Screen**.

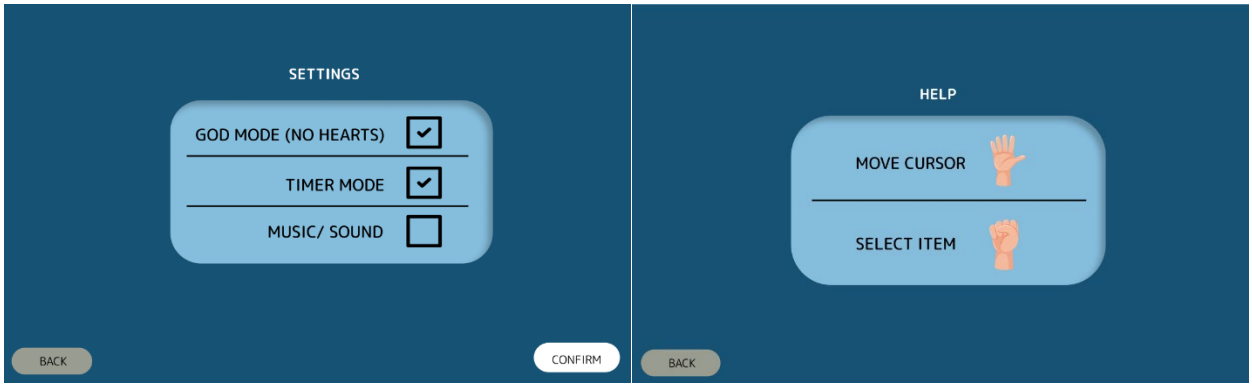


On the **End Screen**, the score is displayed on the top centre of the screen. At the centre we can now see the answers. There are two sections: **Correct Pairs** and **Incorrect Pairs**. The rest of the layout remains similar to that of the **Game Screen**. Instead of **Quit**, we have the **Main Menu**. The function of this button remains the same, both taking the user back to the **Category Menu**. Because of this, it is ideal that the position of the button remains the same and users can remember the function of the button. **Submit** is replaced with **Try Again** if all the answers are not correct. **Try Again** lets the user replay the same level.



If the user gets all the answers correct, only **Correct Pairs** are displayed on the screen. Below, an

instruction reads how to learn about the word. When the words are grabbed, a pop up bubble shows the pronunciation and definition. Additionally, the word's pronunciation can be heard out loud. This action is repeatable. So the users can hear the pronunciation through this interaction as many times as they want. On this version of the **End Screen**, the **Submit** is replaced by **Next Level**. Note that all the buttons on the bottom right corner of the screen in our application takes the players to the next new screen most of the time. This was important for maintaining consistency throughout the button positioning design. This button takes the players to the next level, i.e. different combinations of words.



Finally, the **Settings** screen shows the different modes we have implemented into the Game Mechanics. The **Help** screen shows the basic instructions to control the interface of the application.

3. Game Mechanics

First, the category descriptions show the progress a user has made, giving them a sense of progression with statistics. This gives users extra motive to continue and learn all the words available in our application's library.

Before the user starts, they have the option to select a difficulty for the game. The **Easy** setting has 5 **hearts**. **Normal** has 3 **hearts**. **Hard** has only 1 **heart**. When the player submits their matches and they have any incorrect pairs, they lose 1 **heart**. They can only proceed to the **Next Level** if they get all pairs correct. When all **hearts** are lost, players must restart the entire category. This may seem frustrating, however it is crucial for implementing the words into a user's long term memory through repetition. As the levels increase, the number of words and the difficulty of the words increases. When you complete a level, your score increases by 1 and the cumulative score is displayed on the **End Screen** after each level.

The **God Mode** is available for players who do not want a challenging experience. This removes the **hearts** and the players can keep retrying a level until they get all pairs correct. However, this mode is not recommended as it takes away the opportunity to learn through repetition. The **Timer Mode** makes the game much harder, giving players a limited time on each level. If they fail to submit before time runs out, they lose a **heart**, and need to restart that level, with a brand-new timer.

4. Comparison with Duolingo

Duolingo is actually a really great way to learn language through traditional means. They make good use of **listening exercises**, **flashcards**, and **multiple-choice questions** to drill you on new words, phrases, and sentences. However, one of the problems with that is, these are traditional, meaning a lot of students, especially kids are very familiar with these kinds of questions and they wouldn't be stoked to study through this manner. Therefore, we believe that our app has an edge. Since our goal is to make learning language fun and interesting, with targeted ages being children, it can lead to them having new experiences, a unique way to learn.

Gesture interactions are more interactive for the users, and they can retain the children's retention with being packaged as a game but it is educational. Not only this, as shown in the study [6], using a gesture interactive game-based learning approach improves children's **learning performance** and **motor skills**. Not only are they learning a new language, but also they are developing their motor skills which can hit two birds with one stone. It is an effective learning method that improves both learning performance and motor skills to a greater extent compared to learning language through traditional means such as Duolingo.

5. Demo Application Code Description

We have devised a working early version demo for the app with a few functioning properties.

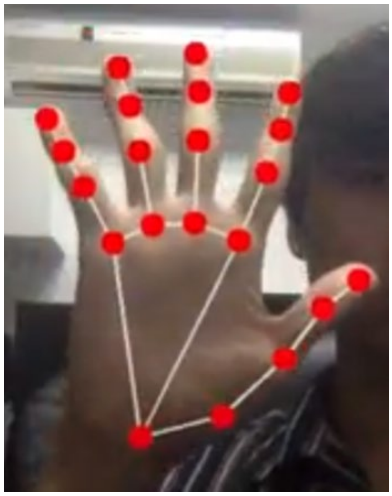
```
import cv2
import mediapipe as mp

cap = cv2.VideoCapture(0)
cap.set(3, 1280)
cap.set(4, 720)
mpHands = mp.solutions.hands
hands = mpHands.Hands()
mpDraw = mp.solutions.drawing_utils
red = (0, 0, 255)
green = (0, 255, 0)
white = (255, 255, 255)

font = cv2.FONT_HERSHEY_SIMPLEX
org = (200, 50)
fontScale = 1

totalCircles = 4
centerPoints = [[200, 100], [200, 270], [200, 440], [200, 610]]
circleColors = [red, red, red, red]
text = ["sad", "boy", "dog", "sky"]
positions = ['U'] * totalCircles

inside = False

radius, color, thickness = 70, red, 10
```

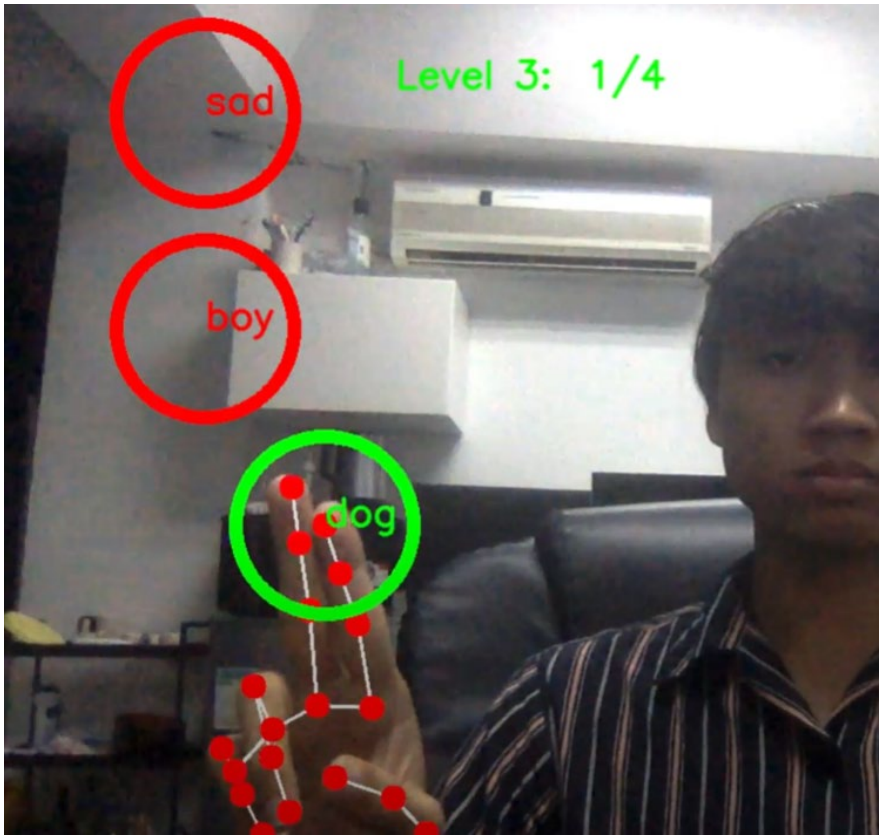
First and foremost, we decided to go with "**mediapipe**" which is a cross-platform, customizable ML solutions for hand gesture recognition, which is a Python package.

These first lines of code are basically to set up the environment and give names to the variables that are going to be used and also to circles which will be dragged in the later code.

```python
while True:
    success, img = cap.read()
    img = cv2.flip(img, 1)
    imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    results = hands.process(imgRGB)
    multiLandMarks = results.multi_hand_landmarks

    if multiLandMarks:
        handPoints = []
        for handLms in multiLandMarks:
            mpDraw.draw_landmarks(img, handLms, mpHands.HAND_CONNECTIONS)

            for idx, lm in enumerate(handLms.landmark):
                # print(idx,lm)
                h, w, c = img.shape
                cx, cy = int(lm.x * w), int(lm.y * h)
                handPoints.append((cx, cy))

        for point in handPoints:
            cv2.circle(img, point, 10, (0, 0, 255), cv2.FILLED)

        firstFingerOpen = False
        secondFingerOpen = False

        if handPoints[8][1] < handPoints[6][1]:
            firstFingerOpen = True
        if handPoints[12][1] < handPoints[10][1]:
            secondFingerOpen = True

        for i in range(len(centerPoints)):
            d = int(radius ** 2 - (
                        (centerPoints[i][0] - handPoints[8][0]) ** 2 + (centerPoints[i][1] - handPoints[8][1]) ** 2))
```

In this code, we are setting up the hand gesture recognition and what can be done with what hand signs. For instance, we used **openCV** to draw a circle of all the points in the hand with this code.



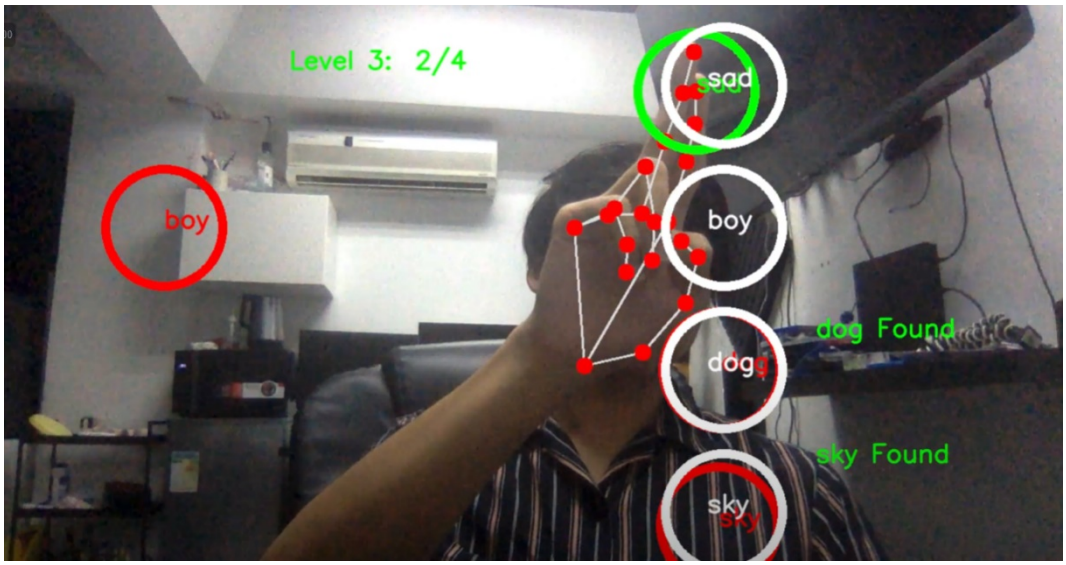


When two fingers are together in the code, the red circle turns into green circle and also the text changes into green text, to visually show that they can now drag the object with **two-finger** gestures. The circle will only be draggable if and only if both the first and second finger are open like shown in the image.

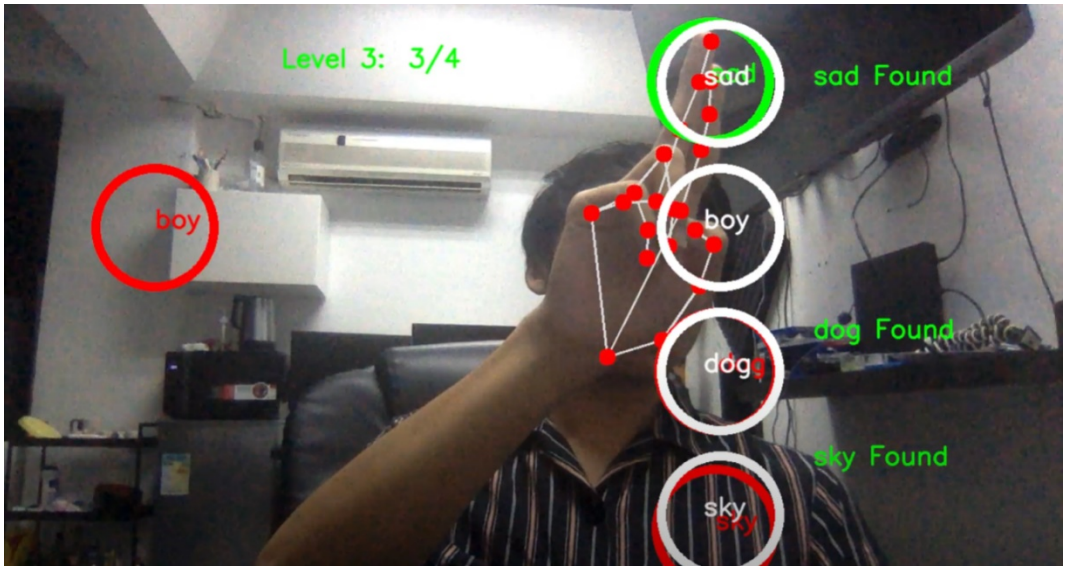
```
if inside:
    if firstFingerOpen and secondFingerOpen:
        circleColors[i] = green
        centerPoints[i] = handPoints[8]
```

The idea is to drag the object into its own pair. **Tuple** could be used but we figured for the demo, to just show that if the position of the circle is close to the position of its pair, then it is a match.

BEFORE



AFTER



When it is dragged to its position, it will say that it is found, and it will also update the **count** by 1. This is to let the player know that they still have one more to match, or items to match. Please note that both the pairs are in English for demonstration purposes. One circle would be Chinese and the other would be in English.

After all 4 is found, it will say it is a success and take you to the next level.

```
totalCircles = 4
centerPoints = [[200, 100], [200, 270], [200, 440], [200, 610]]
circleColors = [red, red, red, red]
text = ["sad", "boy", "dog", "sky"]
```

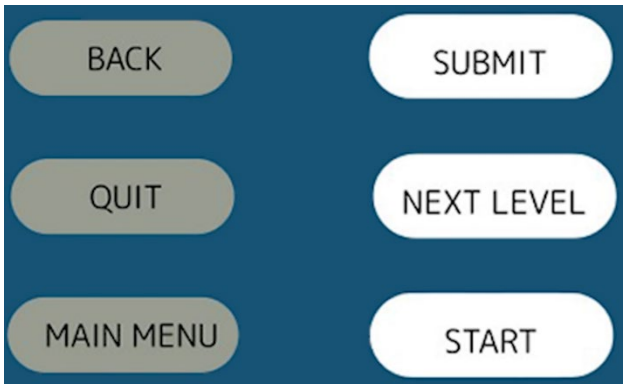
```
for point, color, arrtext in zip(centerPoints, circleColors, text):
    cv2.circle(img, point, radius, color, thickness)
    cv2.putText(img, arrtext, point, font, fontScale, color, 2, cv2.LINE_AA)
```

We have made the code easier so different words and circles can be added without much effort. This will be like an array of new words if we decide to add more in the future. Even if there are hundreds. The *zip* in python grouped together the centerPoints, circleColors, and the text arrays together, when it is dragged or moved, it updates in all 3 of them. For instance, the **centerPoint** for circle A is x200,100 and its **circleColor** is red, and the **text** of it is sad.

6. Theories, Principles and Guidelines applied

8 golden principles of interface design

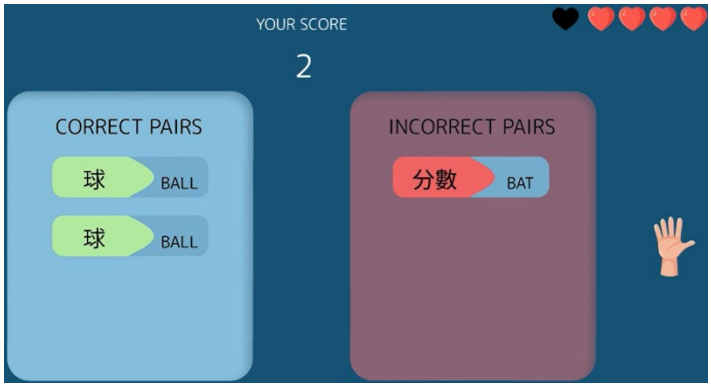
Consistent layout



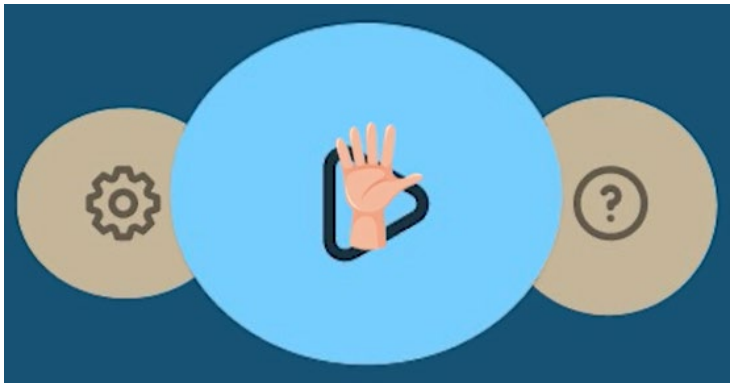
We have implemented same shape designs for buttons that have similar functions. In the screenshots provided above, we can see the examples.

- ✓ All menu item buttons are squares with rounded corners.
- ✓ All sub-menu item buttons are circles.
- ✓ All navigation buttons are rectangles with rounded corners.
- ✓ Same colours are used for buttons that serve the same purpose. For example: White colour is used for primary navigation buttons while a grey colour is used for secondary navigation buttons.

Offer informative feedback



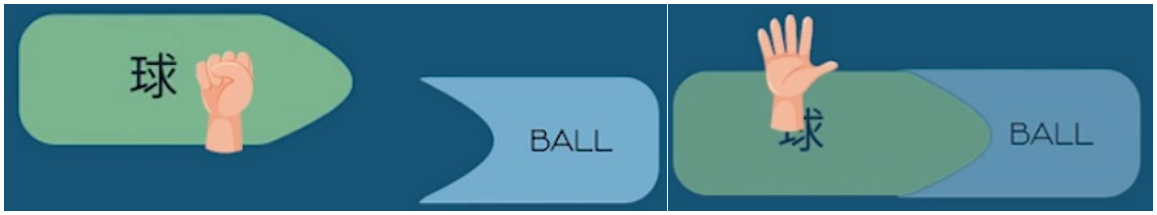
Number of hearts on the top right of the **Game Screen** represents the number of lives/attempts left. **Correct** and **Incorrect Pairs** are separated into 2 groups for clarification.



When we hover our Hand over a button, the menu items are highlighted and expand in size. Offering informative feedback to the users.

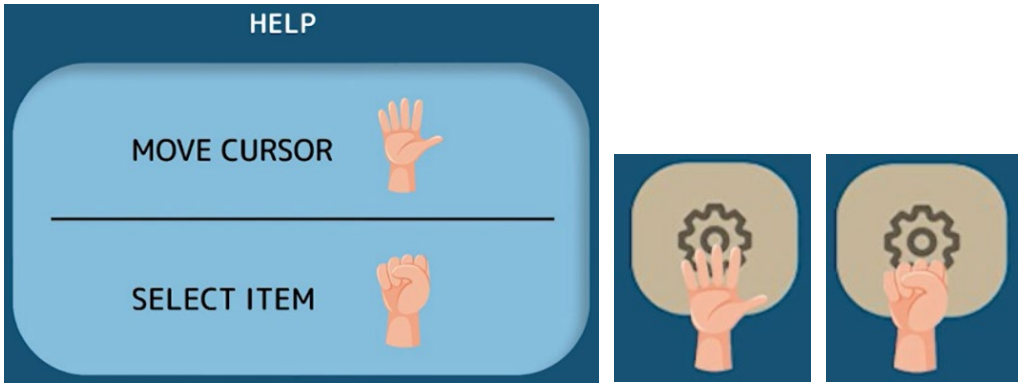
An information block helps the user track their progress; such as, the number of words they have learned in the selected category and the percentage of words learnt relative to the total number of words available in the application's library.

Offer error prevention



When the puzzle **pieces** are dragged close to each other and released, the pieces stick together, mimicking magnetic-like properties. This is crucial to minimize users not being able to match the **pieces** properly.

Cater to universal usability



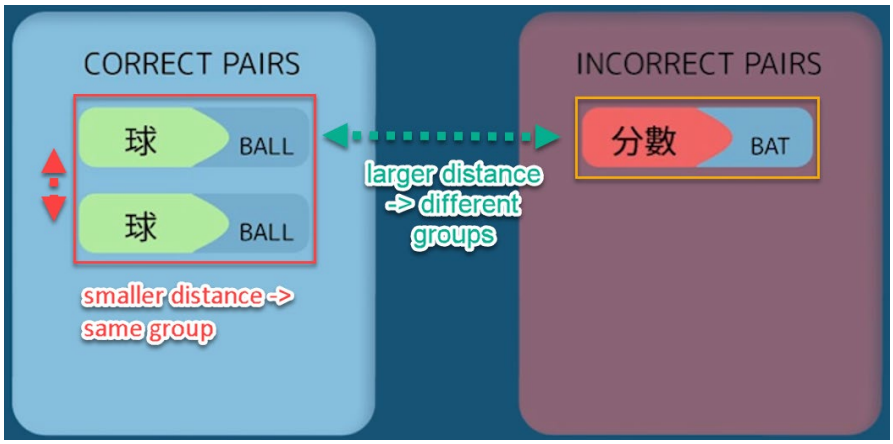
The controls for our application is simple, intuitive, and immersive because it uses hand gestures. It is easy to learn for any user types. First-time users can learn how to use the gesture controls by using the **Help(?)** button. Moreover, this **Help Screen** is displayed for 10 seconds when the application is launched for the first time.

With a fist open, the cursor can be moved freely.

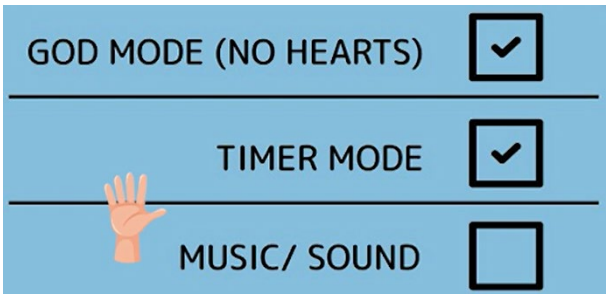
Closing the fist is used to select items on the screen.

Gestalt Principles of Perception

Proximity principle



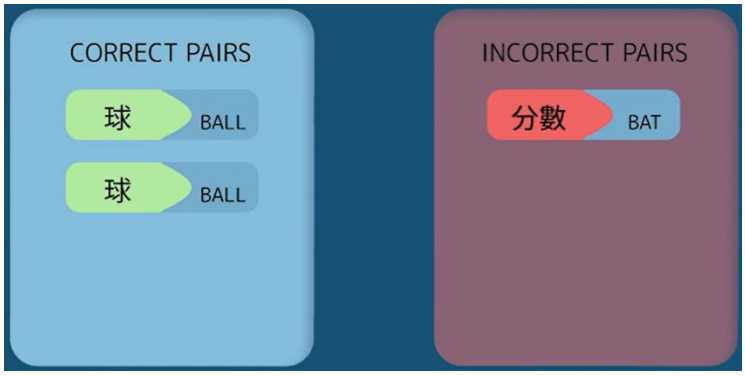
Objects close to each other are regarded as objects under the same group. An example of this is shown in the screenshot.



Texts are aligned to the right, narrowing the gap between two columns. Although the time to search for the beginning is longer, our alignment method has been proved to be better by an eye-tracking experiment.

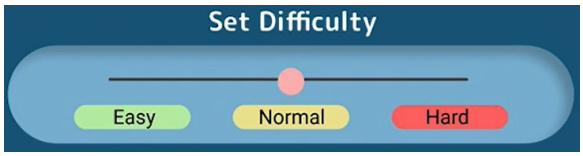
Similarity principle

Similar properties are regarded as the same group.



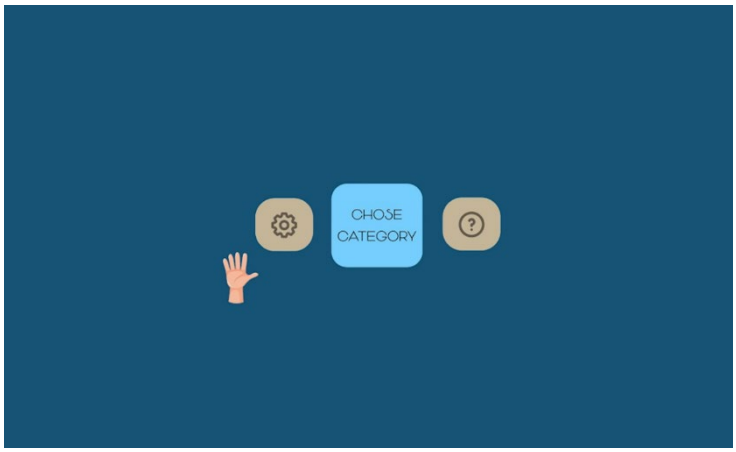
Correct and incorrect matching pairs are separated into two groups with different contrasting backgrounds.
Chinese characters in convex shaped **pieces**, while English characters in concave shaped **pieces**.

Continuity principle



Easy, **Normal** and **Hard** divides the slider into 3 segments from left to right in one horizontal line. Users tend to see things as smooth, continuous representations.

Area principle and surroundedness principle



Based on the area principle, the buttons with a small area are seen as a foreground.
Based on the surroundedness principle, the dark blue area is the background that surrounds the buttons (foreground).

Sense theory

Users with Disabilities

As humans are one of the essential factors in human-computer interaction design, the application is supposed to cater to anyone, including those visually impaired with color-blindness. A specific color scheme is adopted for 4 major color blindness. In addition, there are different shapes of matching buttons. It is easy for color-blindness to identify buttons by shapes.

Stimulus intensity

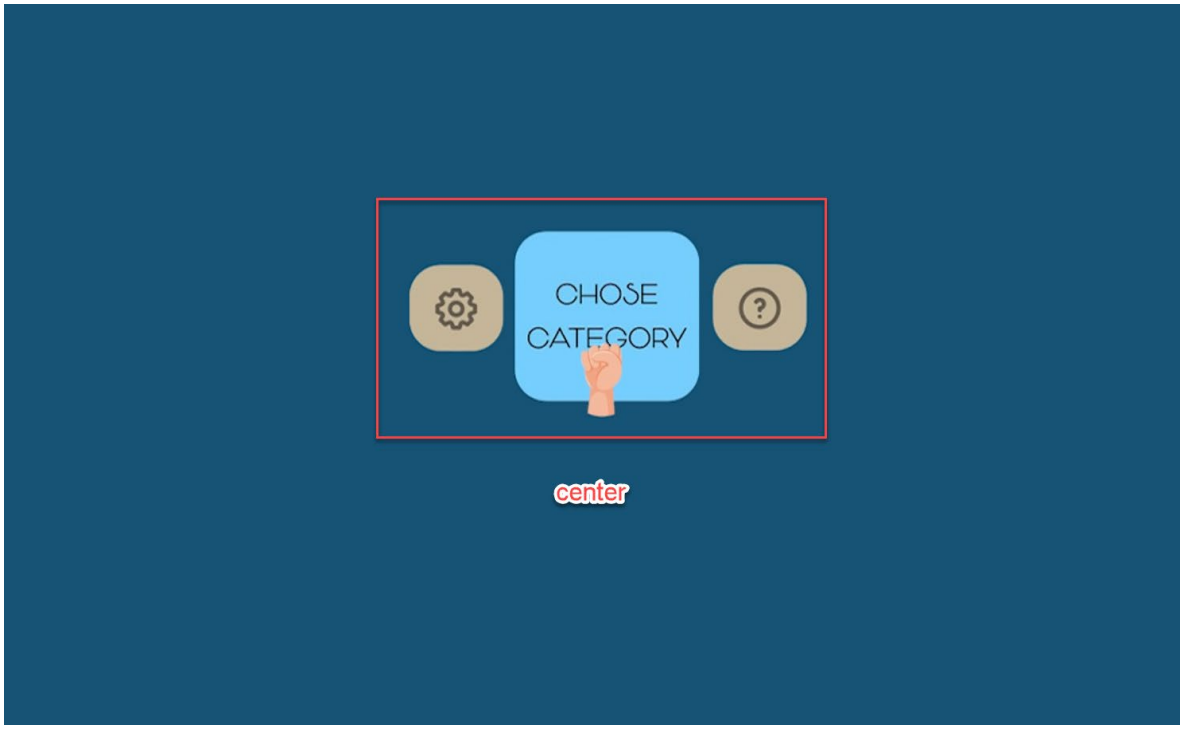
Only 1% of cone cells are responsible for high resolution vision but 99% of rod cells are responsible for low light vision. The title, numbers of **Words Learnt** and **Percentage Complete** are bolded as they provide more important information to users. Rod cells detect these key characters by guiding fovea

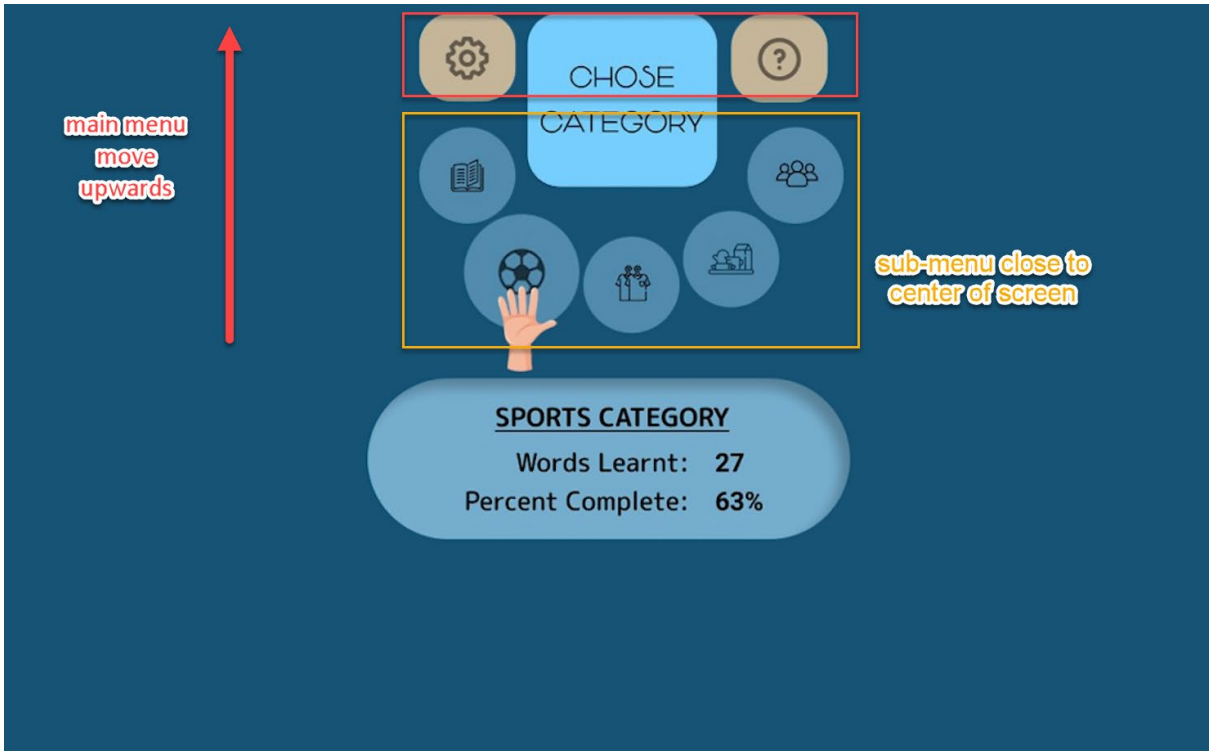
to focus. It is easier for them to notify these key information as users respond first to the intensity of a stimulus before interpreting their meaning.

Cognitive theory

If the pairing is correct, the application can pronounce it many times by opening and closing fists. As humans remember sounds better than words and rehearsal imprints the information stored in the short-term memory into our long-term memory, users can memorize the vocabulary effectively.

Fitts' law



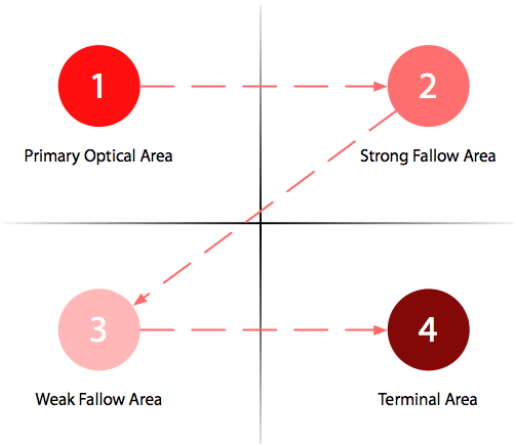


The main menu is at the center of the screen when this page is accessed. According to Fitts' law, the access time is inversely related to distance. The access time is shorter as there is a smaller distance between the cursor and main menu items.

After pressing the main menu items, the main menu moves upwards and sub-menu items pop up. The access time is also shorter as there is a smaller distance between the cursor and sub-menu items.

The area of each hot zone covers both a label and a checkbox, the access time decreases as the button size increases.

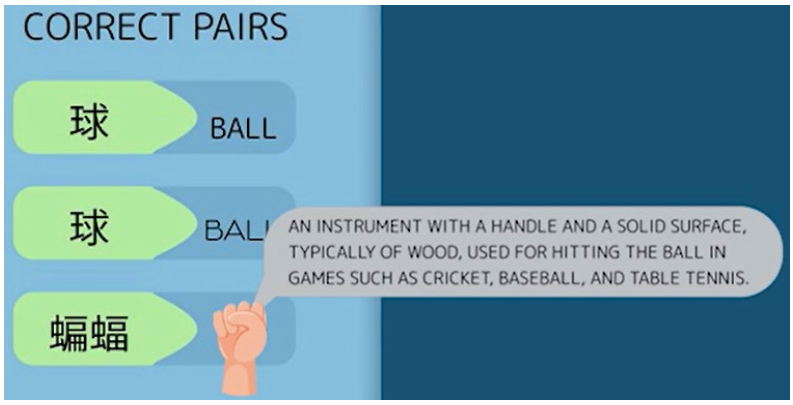
Gutenberg's diagram



Submit button is located at the terminal area because this area is for users to make a decision.
Quit button is located at the weak fallow area because we want to discourage users to quit the game.

Others

Tooltip



When the hand hovers on the matching pair in the **End Screen**, a tooltip pops up to show the definition of the word and an audio is played that pronounces the Chinese word. By using the tooltip, more space can be saved.

7. Description of the alternative modality of interaction implemented in the design

When developing our application, we tried to make it very interactive and engaging. We wanted our application to be interactive from the start. We decided to use gestures to interact with the interface and wanted gestures to be used for navigation. An example of this is requiring the user to close their fist to click a button and using the user's hand, or more specifically their fingers, instead of a mouse. All these options can be disabled in the settings menu if the user wishes but doing so takes away the foundation of our application. We decided to replace the cursor with this interactive option because it helps keep the user engaged and the simple action of opening and closing the fist gets the blood pumping, which helps the user stay aware of their action and not get lost in their thoughts.

At the end of the day, our goal is to teach our users a brand new language through immersive and interactive gameplay.

8. Possible future extensions of current design

We have many exciting extension ideas for our current design.

Our first idea is to implement speech. Speech is a very broad field, but that's also why we have so many ideas about how our application pronounces new words so the user can learn them, or how the user himself pronounces the word in Mandarin that is displayed in English, or vice versa. Studies have shown that pronouncing words or phrases out loud helps people remember what they read better.

Our second extension idea is to draw the characters or write words. We could use gesture recognition and have the user write the word in the air to learn it. It has also been proven that writing down learning content helps people remember the learned information better.

Our third implementation idea is, of course, to develop our application to offer a wide range of languages to our users. After Mandarin, we would like to implement the languages that our users would like to see, like French, Spanish...

Our fourth and last idea would be to create a community website or subsection. Our goal is for our users to be able to share their progress with friends, but also get help from other community members, like mnemonic methods or just share their experiences.

Another tweak, we would probably add before releasing our application would be to give users more

control over their interaction with the application. We had the idea of letting the user choose which gesture has which effect on the application from a set of implemented gestures.

9. Workload Breakdown

Michael Kurzewski:
Animated the interface functions in the video demonstration and compiled all audio and videos.

Barshon Clinton Sarkar:
Wrote the structure of the final design and the relationship between each screen. Presented these screens with screenshots and highlighted the important features. Revised the whole report and edited accordingly.

Leung Wai Keung:
Provided a detailed analysis of the principles/guideline/theories learnt in this course, applied to our interface design. Proofread the document and perform Word processing of the whole project.

Haurat Eliott:
Wrote the analysis on the user community, described the alternative mode of interaction and wrote possible future extensions. Also did the voice over for the video demonstration.

Rai Phirens:
Wrote code to demonstrate interface interaction. Wrote the report for the code and showed a demonstration of it in video format. Also compared Languaid with Duolingo and wrote a report on it.

References

[1] https://www.statista.com/statistics/266808/the-most-spoken-languages-worldwide/
[2] https://www.washingtonpost.com/local/education/half-the-world-is-bilingual-whats-our-problem/2019/04/24/1c2b0cc2-6625-11e9-a1b6-b29b90efa879_story.html
[3] https://www.grammarly.com/blog/language-learning-costs/
[4] https://w3-lab.com/website-font-size-guidelines/
[5] https://enchroma.com/blogs/beyond-color/interesting-facts-about-color-blindness
[6] Hsiao, Hsien-Sheng & Chen, Jyun-Chen. (2016). Using a gesture interactive game-based learning approach to improve preschool children's learning performance and motor skills. Computers & Education. 95.10.1016/j.compedu.2016.01.005.