ARamsay: An Augmented Reality Cooking Assistant on HoloLens

RUN HUANG, Fudan University
LINGSHUANG KONG, University of California, San Diego
SHUTONG WU, Fudan University
YIFEI NING, University of California, San Diego

ARamsay, a HoloLens-based app, aims to provide a better solution for people interested in cooking and learning new recipes. Personalized, actionable and accessible, the app introduces a whole new approach to cooking in the modern kitchen.

Additional Key Words and Phrases: Mixed / Augmented Reality, HoloLens, HCI

1 INTRODUCTION

In this paper, we introduce ARamsay, a HoloLens-based AR application that helps people learn new recipes smoothly and interactively. The motivation for this project is to improve people's experience at cooking and to better accommodate some people with special dietary requirements. We also covered the design for our features and introduce the architecture of each part: mobile end, Hololens end, and server-end; then explained the rationales for using AWS server as the database, mobile app as the user-end devices, and Hololens as the display devices. In the evaluation section, we recognized that the built-in gesture recognition isn't as stable as the commercial-grade counterparts. Server connectivity also needs prospective improvements to achieve faster responding time. The mobile end, likewise, could be improved to be more stable. Despite that, our application ARamsay projects a broad market potential that deserves researchers' attention in the field of Augmented Reality.

2 MOTIVATION

The traditional ways to cook new meals require lots of time and energy, not only to choose and learn the recipe but also to ensure consistent quality of the meal that satiates people's dietary needs. The learning process is oftentimes not smooth at all. One needs to switch between different platforms, positioning the devices at a certain angle, and hustling themselves into overwhelming information, all of which discourage them from learning. It's also not interactive in a way that takes into consideration the user experiences to indicate if they have understood the instruction step by step.

More importantly, the selected recipes may not meet the needs of those with special dietary requirements. For instance, some may have allergic restrictions so they need to avoid having certain ingredient(s). Some may also keep a strict standard on the daily calories intake. Randomly choosing recipes online fails to satisfy those people. Additionally, without indicators to tell users the time and the number of ingredients needed at each procedure, the quality of meals could not be secured. Old-fashion learning methods tend to cause more confusion because you need to read instructions from a digital or paper media while cooking at the same time. With occupied hands, people couldn't conveniently check the instructions with clean hands. Switching back-and-forth between different media creates potentially more post-cleaning work. In short, a complete mess.

Therefore, for those who have dietary preferences and eager to learn new recipes in a more elegant and convenient way, we wish to build a solution that addresses this problem in the following dimensions:

(1) keep kitchen hygiene by reducing physical contact with the environments

(2) make the recipe learning experience more smooth and interactable
(3) ensure the consistent quality of prepared meals
(4) recommend personalized recipes that meet people's dietary requirements

The solution should introduce a smooth and interactive User Interface (UI). Traditionally, the media used to contain the recipe instruction were confined in a physical environment, which is not easily accessible to carry around. In the application, we plan to make UI floating so it would no longer be confined to the environment. Moreover, when watching a video instruction on preparing food, the user may need to pause multiple times to precisely understand and execute each step from the recipe book. Observing this inconvenience, ARamsay should display instructions (in the form of either text, video) interactively–only display a step ahead upon request, so the user can have more time to learn and execute each step.

Another design idea is to tell the user the time needed in each step to precisely control the food quality. One of the most important factors to ensure consistent food quality is time. To precisely control the time at each step, it is necessary to build a timer into the system.

Preparing nutritious and delicious meals while meeting dietary needs is a common challenge for most of us. Thus, we will introduce an embedded recommender system that takes user input data (ingredients, weight, height, BMI, blood pressure, allergies, etc.) and provides personalized meal options. The application recommends dishes that best meet the user's demands.

When comes to the decision for devices, we eventually choose HoloLens instead of other AR/VR glasses based on two considerations: first being that we need to keep our users in the real world instead of a fully virtualized scene since our users are doing actual cooking, so Hololens, which provides better Mixed-Reality experience, is more capable of meeting this augmented reality need. The second one is that HoloLens provides better hand-tracking and gesture support for both developers and users.

One final problem is that, many AR/VR devices such as HoloLens are heavy and users may not want to wear it for a long periods of time. In addition, typing in HoloLens is a pain. What's more, imagine such scenario where users need to find a recipe and check all the ingredients needed, it is impractical to have users wear a HoloLens to the market. Therefore, We'll only use the HoloLens as a display and gesture input module. Those that require a lot of input or heavy computing tasks will be done outside of Hololens, e.g. on a mobile application. After users select a recipe on the mobile app, it will then send the recipe data to a remote AWS server and HoloLens will automatically fetch from it.

Finally, the usage of our solution would feel like this. Users first input their data into the mobile-end application. The application takes user input data to output a batch of recipes. Users choose which recipe to send to Hololens on the mobile end. Ones may choose from a batch of recommended recipes on the Hololens menu interface. After selecting their favored dishes to learn, the fetched recipe data will be displayed in the instruction scene. With simple hand gestures such as click, tap, and drag, they could then learn the recipes interactively step by step. A timer placed on the screen can tell users the time to prevent food from being overcooked or undone. Conclusively, the proposed solution is centered around the four dimensions defined above, turning cooking into something more joyful for users.

3 DESIGN

In our design, the ARamsay mobile app has features including recipe search, recipe list, recipe detail, and recipe transmission. The holoLens app features include recipe selection, recipe display and page turning.

3.1 Mobile App Design

In the first two weeks, we designed a low fidelity prototype for ARamsay mobile App (As shown in Fig. 1). Users can enter the ingredients they have and click the search button. The mobile app then display what the user can cook with the ingredients they entered. Users select the recipe they want to cook and we will show them the details of the recipe, such as summary, nutrition information, ingredient.

You can try our prototype in Figma: Low fidelity prototype Link

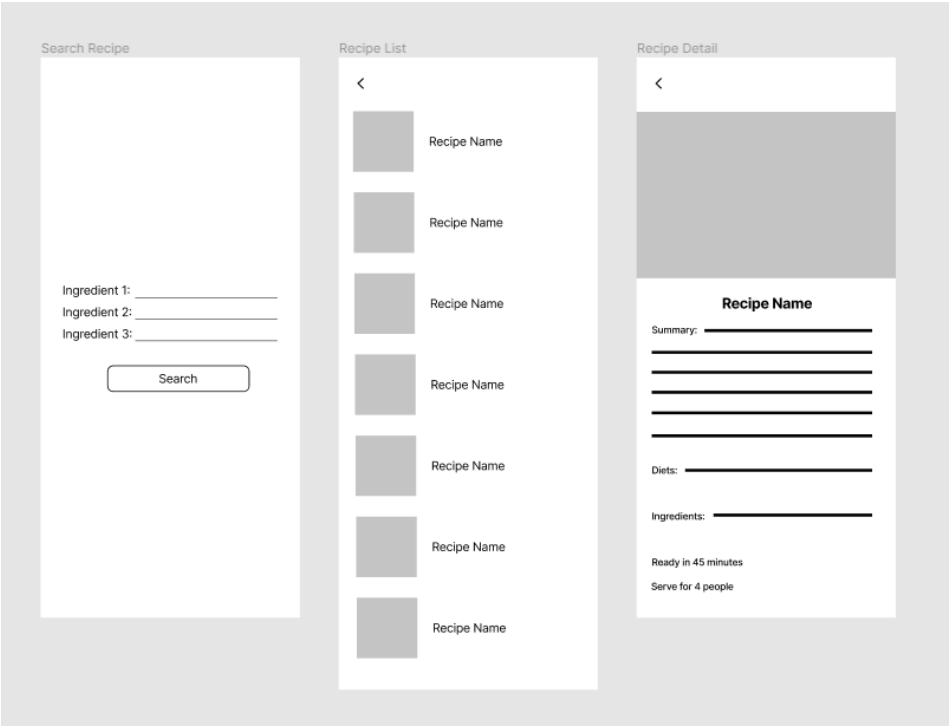


Fig. 1. Mobile App Design

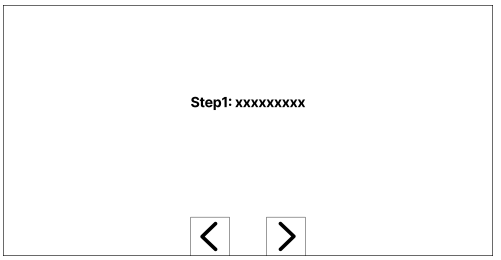
3.2 HoloLens Design

We mainly focus on the way users interact with the HoloLens in this section. Because users often need to navigate through the recipe steps as quickly as possible when cooking, we need to make the interaction simple, fast and friendly to give users a pleasant cooking experience. During the development sprint, we designed three ways for users to turn pages, and finally chose the third one.

3.2.1 Gestures.

In the first week of development, we adopted an interactive way of turning pages with the "Pinch and Drag" gestures (As shown in Fig. 2). Users were required to pinch at the panel and then drag it from one side to the other for at least 10 centimeters in order to turn the page. However, we soon discovered the drawbacks of this gesture. HoloLens needs to recognize three user actions: pinch, drag, and release. Any wrong recognition will lead to failure when turning the page. Therefore, although this gesture may look intuitive, it is not reliable and thus not suitable for this project.

(a) 1st version of Hololens App



(b) 2nd version of Hololens App

3.2.2 Button Click.

In the next two weeks, we designed and developed the second interactive method, which is to click buttons to turn pages As shown in Fig. 3. Clicking the button, as an interactive method that the public has long been accustomed to, undoubtedly reduces the learning cost. Moreover, clicking a button requires only one operation compared to the gesture method, and that is just clicking. But during the experience, we found that clicking on AR devices is not as easy and accurate as on mobile phones or computers. Users need to target specific buttons to operate, which will also consume the user's time. This is not the most ideal way for users to interact.

3.2.3 Airtap.

By reflecting on the first two methods, we further thought about whether users could perform just one action without spending time to find and align a specific button to turn pages. We finally proposed the Airtap gesture (As shown in Fig. 4). First, we vertically split users' view into two parts. When the user taps on either side, the page can be turned accordingly. This interaction way greatly saves users' time to turn a page.

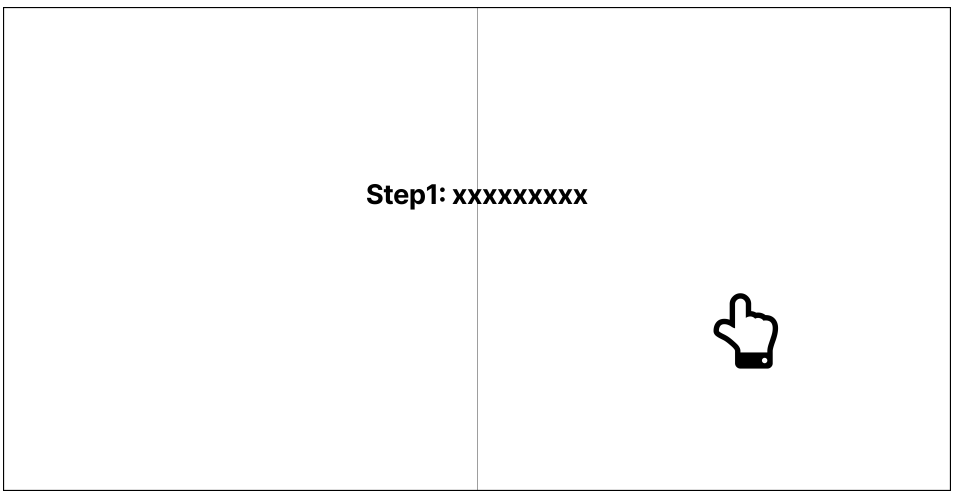


Fig. 3. 3rd version of Hololens App

3.2.4 Timer.

We also implemented a timer panel (As shown in Fig. 5) to help users keep time and provide them with a more friendly experience.

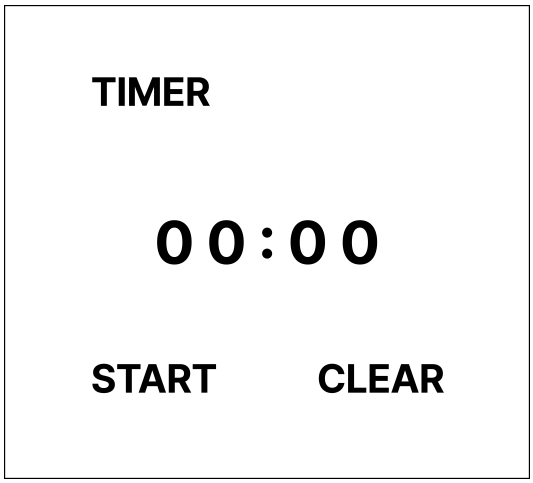


Fig. 4. Timer

4 DEVELOPMENT

4.1 Architecture

The overall structure of Aramsay is shown in Fig. 6, which mainly includes server, Android client, and Hololens Client. The back-end server is built using the Node.js and Express framework. Mainly divided into interface layer, business logic layer. The Android client of Aramsay realizes information interaction by calling the back-end interface. The Android client also calls a third-party API to request recipe information.

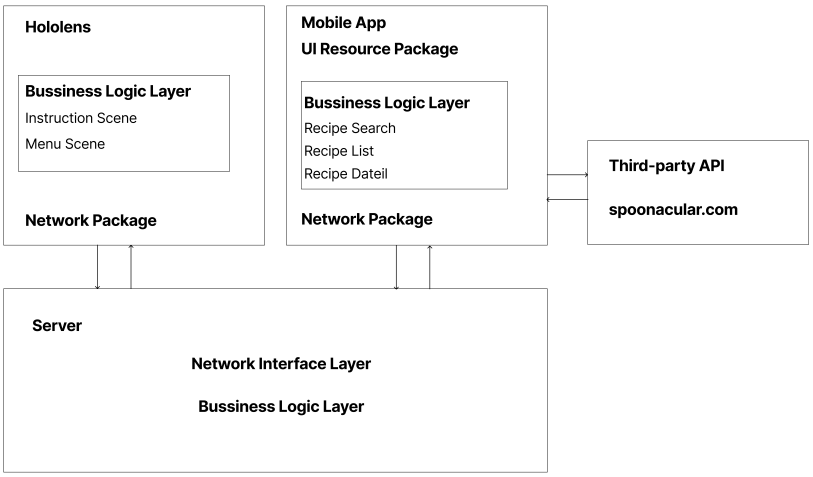


Fig. 5. Architecture

4.1.1 *Server Architecture.*

The AWS server has 2 layers, from the bottom to the top is the business logic layer and the network interface layer.

The business logic layer mainly processes HTTP requests from Android and Hololens clients. According to the requested URL, different business logic is processed. In this layer, the logical processing of getting and posting requests.

The network interface layer defines interfaces and routing tables, which can help the server differentiate and process different HTTP requests.

4.1.2 *Android Client Architecture.*

The Android client has 3 layers, from bottom to top are the UI resource layer, business logic layer, and the network layer.

The UI resource layer defines the resources that need to be used in the Android development process, such as XML layout files, pictures, colors.

The business logic layer defines the logic of each functional module, which mainly includes the code of Android activities. Each Android activity displays an Android screen, and the layout file of the UI resource layer can be displayed on the client in this layer. This layer also includes the adapter for the list and the item view holder. View holders are used to load the view file of recipe items. Adapters can load recipe data to the view holder.

The network layer contains all the network APIs, which can be used by the client to send HTTP requests to the backend server. This layer also implements network request methods, and we use Retrofit [1], Rxjava, and Okhttp to realize chain operation and thread switching.

4.1.3 *Hololens Architecture.*

The Hololens Architecture consists of two scenes construction: Instruction Scene, and Menu Scene. In Menu Scene contains a drop-down panel where users can choose favored recipes fetched from the mobile devices. When clicking on the sub-elements within the panel, a "ClickEvent" will be sent and subsequently, "onClick()" method will be called. The "onClick()" method was linked to customized functions that act correspondingly to different buttons (Left, Right, Add, etc.) The Instruction Scene was used as the display window for recipe break-downs. It contains two interfaces–a instruction panel for recipes and a timer. The gestures such as "airtap" were likewise linked to function calls.

4.1.4 *Data Security.*

To elevate the data security of Aramsay, we deployed our server onto AWS. AWS server can help us with data security because:

(1) Scale Securely with Superior Visibility and Control: With AWS, we control where our data is stored, who can access it, and what resources our organization is consuming at any given moment. Fine-grain identity and access controls combined with continuous monitoring for near real-time security information ensure that the right resources have the right access at all times, wherever the information is stored.
(2) With AWS we can build on the most secure global infrastructure. All data flowing across the AWS global network that interconnects data centers and regions is automatically encrypted at the physical layer before it leaves secured facilities. Additional encryption layers exist as well; for example, all VPC cross-region peering traffic, and customer or service-to-service TLS connections.

4.2 Technology Used

4.2.1 *Server.*

We use Express as the backend framework of this project. Express is a flexible Node.js web application development framework that keeps the smallest scale, which can provide much support for the back-end development of Aramsay. Native Node.js back-end technology has caused many difficulties for developers, such as:

1. Rendering a static page is troublesome, which means every request from the client must be processed.

2. The routing processing code is not clear, and many regular expressions and string functions need to be coded.

So developers can't concentrate on developing business features, because there are many other things to consider. The Express framework can process different requests from clients by defining routing tables. Makes routing processing code becomes clear and intuitive. And there are many middlewares, which help developers to obtain the request parameters from the client and help convenient to send the response to a client, which greatly saves the developer's development time.

4.2.2 Mobile App.

Our app ARamasy is developed through Android Studio using Java. The code mainly consists of two parts : xml files which manifest the layout of the app, and activity files that contribute to the logic flow. In activity files, a bunch of view-holder java classes are introduced to manage the layout structure modularly, through which xml files and activity files are connected. Among these view-holder classes, the most typical one is "RecyclerView" [2] with which the fetched recipes are displayed in a list form. Different from common and older version of ListViews, RecyclerView manages the present items dynamically where a buffer of recent shown items are preserved and not-that-recent items are "recycled" to save storage and enhance loading speed.

The recipe fetching process is achieved by a "get" request to a famous recipe website spoonacular.com. This website provides a prodigious amount of recipes and a variety of ways to search for them(e.g according to ingredients which we are using,or name, diets). On success the request returns a string in JSON format.

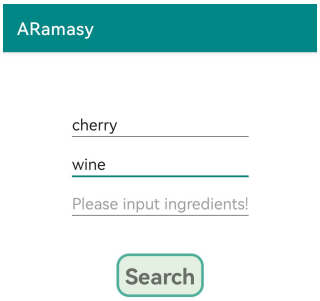
To parse and analyze the JSON result we use "Gson "[3], an open-source Java library to serialize and deserialize JSON to Java objects.

It's also worth mentioning that the recipe list contains rich pictures that we want to show in the app. In order to display them in a calm manner(and not cause trouble to the main thread) we used "Glide" [4], a fast and efficient open source image loading framework for Android which takes advantage of thread pools to achieve concurrency.

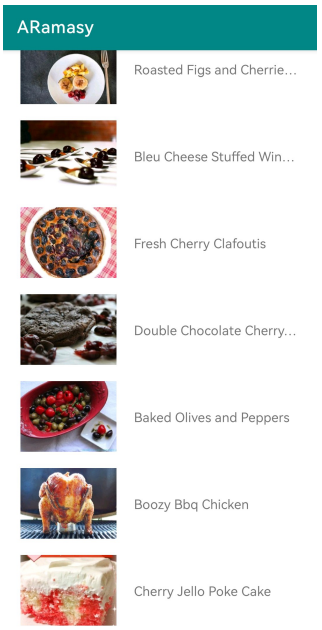
Aramsay also uses Retrofit, a popular network request library. Retrofit is an HTTP request framework based on Okhttp and RESTful. It defines request types and request parameters in the form of annotations, uses a large number of design patterns to simplify network requests, and Retrofit also supports synchronous or asynchronous network requests and supports multiple data types and The analysis of serialization format provides good support for both Android development and backend development.

4.2.3 Hololens.

A Hololens 2 is used to display processed recipe data. It is used as a light duty device for display only and designed not to involve too many time-consuming computations, since processing data on this end devices would significantly slow down its responding speed. Developing Hololens 2 with unity saves lots of time to search for assets, libraries (e.g. MRTK), and documentations. The built-in API enables us to utilize its hand-tracking, gesture support (e.g. air-tap, click, hover), and pre-defined "Monobehavior" Class.



(a) Front page of mobile app



(b) Second page of mobile app

(c) Third page of mobile app

We use WebRequest class to send HTTP Get requests to the AWS server. After successfully requesting the recipe data, we apply the Jsonutility class to parse the response and finally get the recipe list.

4.3 Features

4.3.1 Mobile app.

The one single feature for the app is simplicity. We wanna make it new-user-friendly and easy to get a hand on because cooking itself is hard work. The instructions are embedded within the contents, either shown as a pre-text or emerge after impropriate trials.

This is the front(main) page of the app. User can input at most three ingredients and then click "Search". On clicking, the content of the text entries are fetched and examined. If all of them blank, a hint will pop up urging user to input something before clicking "search". Otherwise the recipe fetching process will be activated.

This is what shows up after the recipe fetching process. A list of recipe entries(list size hard coded by far) with only effect picture and title is displayed. User may scroll up and down to view all of them, and click them if interested to see the detailed page.

Above is the detailed page. The effect picture(placed at the very front) is omitted to save space. The green circle at the bottom right corner is the "save and transfer" icon which sends the necessary information of this recipe to our server on clicking. Users may go back to the previous page by simply sliding left in any stage.

4.3.2 Hololens.

The program developed with Unity Engine consists of two major scenes: instruction scene and menu scene. See Fig.7 and Fig 8.

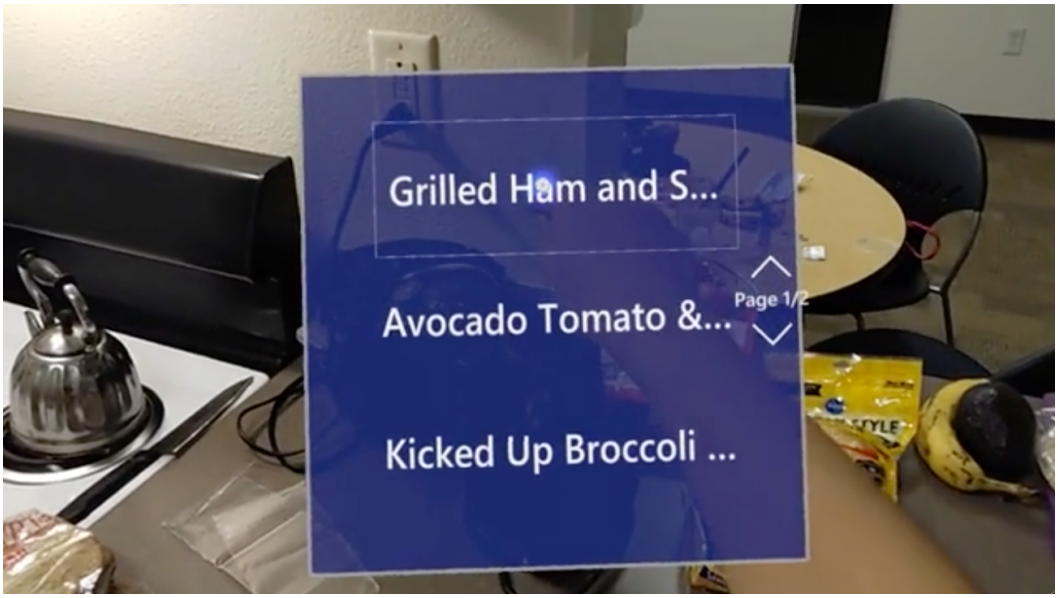


Fig. 7. Menu Scene in HoloLens 2 UI

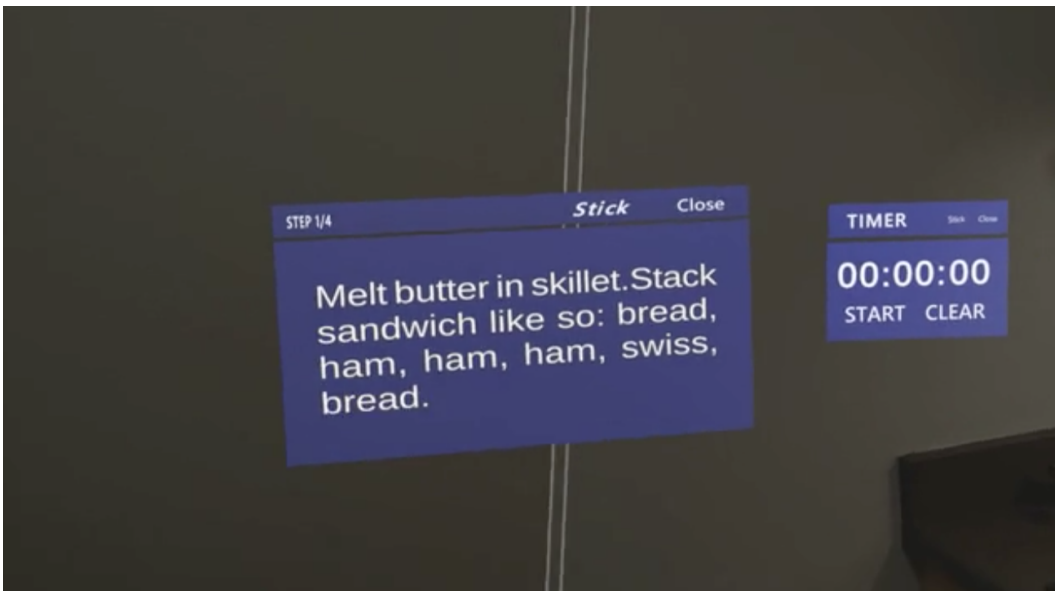


Fig. 8. Instruction Scene in HoloLens 2 UI

5 EVALUATION

5.0.1 Hololens.

- Gesture recognition

Users interact with hololens using a variety of gestures, so a precise and smooth gesture control is vital for a good user experience. By far we've made several attempts, each with a different feature and has been tested mainly on ourselves. We experience through these approaches:

Drag: when a user wants to switch to another page of recipe, they first make a "grasp" which activates the recognition, then drag through the intended direction(right for next page, left for previous). The dragging distance needs to be longer than a limit to activate a page switch.

However, both "grasp" and "drag" seem hard to be recognized when testing. User has to make very exaggerated movement to be seen, which is tiring even if you are doing nothing, let alone during cooking.

Button: we designed a control panel with "left" and "right" buttons where users simply click the button to switch pages. This model is precise in recognition. However, we find it troublesome finding a proper position for this panel. It cannot be too small, which makes the button area hard to click. Considering its size, it would be implausible for it to float within the eyes' range, so we put it down: each time you wanna switch, you bend your head down.

We used this method for a long time until one day one volunteer commented it's annoying having to look for the panel for each step switch. After that we started to look into still another approach.

Airtap: in this version we draw a vertical line in the middle of the view. Each time you wanna switch, make a tap gesture on the designated side of the line.

This method turn out to be the best one among all models, with a high recognition rate and elegant to implement. Still, according to some side-viewers, they reckon the middle line unnecessary and annoying. However, without this line there's no better way of telling the boundary between the two directions, so we simply put on with it.

Moreover, this method sometimes seems too easy to be recognized(a rather high false negative rate). When simulating doing kitchen work, we found that it's highly possible that the finger movements will be mis-considered as "switch" gesture. A possible solution to this is to adjust the time needed to activate a "switch".

- Connection to server

Hololens needs to fetch the recipe from the server before user could start choosing. We wrote a loop logic where it sends a request to server every 10 seconds until the fetch succeeds. When testing on this step we found the fetching process rather slow on average. Either the hololens or the server side network could be down, causing a potential long wait. We might add a loading sign here in case user run out of patience.

To summarize, the hololens has realized the basic functions we initially designed, but there's still a long way to go to make it a satisfactory and enjoyable experience. Due to limited time, we haven't tried obejct detection or video instruction. So right now what we've achieved is just a prototype, awaiting further exploitation.

5.0.2 Mobile app.

Data transmission: Since the app serves as a primary exchange station of data, the priority of testing lies in the stability of the data flow.

- recipe fetching

The target website requires an apikey(related to each account) to use its api and each account has a max free access of 150 times a day.

To test its integrity we simply keep on clicking "search" and see what happens after free access is used up. It turns out the GET request still succeeds with an answer code 200 but with NULL value in the returning string and would through a JSON exception.

Currently we don't have a large user base so having 5 apikeys are just enough for us. To solve this problem we modified the fetching logic to be structured as a loop where it iterates through the apikeys until fetching process succeeds.

Now we have 750 free access a day and it never male-functioned ever after the modification. If one day it does, we will be very proud.

- recipe to server

It turns out this action is highly dependent on the network condition. With a poor network the after-click notification will lag behind a lot and often shows up as "fail to send". Meanwhile the server's state also matters. We notice that the server misbehaves sometimes and won't accept new arriving recipes until we refresh it. We're still trying to figure out a solution for this phenomena.

Data parsing:before sending recipe detail to the server, we feed the raw data into a function which filters the data, extracts useful details for hololens's display and then do some pre-processing. By far only the "steps" column is extracted as this is the only thing used in hololens. In pre-processing we parse the sentence extracted into sub-strings of length below a certain limit. The way we detect "string" is to use periods as separation sign. But according to observation, this parsing might return blank text sometimes, which might be caused from the use of more than one periods. And since the recipe text is not formalized, some gibberish sequences might be contained which also pose a challenge to the parsing function.

The mobile part address the initial goal of our project beautifully as it successfully realized its purpose of recipe data exchange and user interaction. Further exploitation of the data it fetched is on the way. However, a blank in a counter app suitable for IOS system still requires to be filled.

6 COLLABORATION

6.1 Structure of the team

overall division:

- mobile app: Shutong Wu, Lingshuang Kong
- hololens: Run Huang, Yifei Ning
- server: Lingshuang Kong

weekly progress:

- week7: Run and Yifei together trying to find api for hololens gesture interaction and built a basic prototype using grasp and drag. Lingshuang working on data communication between hololens and PC. Shutong managed to fetch recipe using java and taught to use Android Studio by Lingshuang.
- week8: Run and Yifei fixed the bug that recipe page turning in hololens isn't working, and tried to use the button to switch pages. Run iterated on implementing the feature that application screen rotates as user's head rotates Shutong coded the first page of the mobile app with the help of Lingshuang. Lingshuang succeeded in sending a string to hololens using java.
- week9: Run managed to code a panel with button for page turning. Yifei learning about how to implement object detection in Unity. Shutong coded the second page of the mobile app with the help of Lingshuang. Lingshuang connected the mobile part to hololens by socket.
- week10: Run tried out the "airtap" gesture to turn pages and added a timer to hololens. Shutong coded the third page of the mobile app with the help of Lingshuang and finished the recipe fetching and parsing process. Lingshuang finally gave up the idea of socket and

established a server. The whole data flow model, including sending recipe to server and receiving recipe on hololens was fully constructed.

6.2 Encountered issues

6.2.1 Server.

(1) We first used TCP communication to transmit data between mobile app and Hololens, but because we were not familiar with the TCP communication principle, we did not connect the Hololens and the mobile phone to the same local area network, so the communication was unsuccessful. After searching for solutions online, we knew how to fix it and connected the Hololens and the mobile phone to the same local area network, and then two devices can successfully exchange data.
(2) Hololens often turns off the socket port automatically, which makes it impossible to receive recipe data multiple times. Users must restart the app to reopen the port. This gives the user a very unfriendly experience. So we used a server to manage and transmit data. Server is more stable than TCP communication, because it won't be turned off by device. What's more, users don't need to connect both devices to the same local area network. Moreover, using a server allows us to extend our features in the future. For example, we can design and implement login, sign up and other features.

6.2.2 Mobile app.

(1) using java base library to parse JSON objects failed again and again until we used Gson, which saved our lives.
(2) It might take a long time to fetch the recipe and the waiting is torture. We didn't speed up the network but added a loading sign, which miraclely eased the torture.

7 CONCLUSION

In this paper, we explored the vast possibilities of applying augmented reality to assisting cooking and introduced our implementation: ARamsay. By employing hand tracking and gesture recognition technology, ARamsay provides an interactive and elegant way for users who are busy cooking to navigate through recipes. It improves users' cooking experience and can even inspire them to learning new recipes. With the growing popularity of AR/VR devices, ARamsay has a broad market outlook in general users who love to cook and culinary schools.

8 FUTURE WORK

Currently, ARamsay is still a prototype and is imperfect in many aspects. In future work, we hope to improve ARamsay in the following areas:

- A more intuitive and friendly user interface:
  For now, the instruction panel in ARamsay can only display plain text content. We plan to add images, videos and some interactive features, such as page scrolling and font resizing to the instruction panel.
- User system:
  We need a user system to manage recipe data stored in the AWS server uploaded by different users.
- Multi-device support:
  We only implemented Android application for ARamsay. The web application for ARamsay is under developing. After that, users who are using non-android devices can also use ARamsay.
- Object-recognition: Recognize needed ingredients and equipment to better help users cook.

REFERENCES

[1] Xiaomin Kong, Binwen Fan, Wei Nie, and Yi Ding. Design on mobile health service system based on android platform. *2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, pages 1683–1687, 2016.

[2] Yi Liu, Chuanchang Liu, and Zhiyuan Su. The diversity layout of e-commerce applications based on android. *2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*, pages 715–718, 2018.

[3] Priyanka Jichkar, Kalyani R. Gawande, Arshadkhan Pathan, and Gangotri Nathaney. Android based department app-using pda system. 2017.

[4] Yoojeong Song, Soo bin Ou, and Jongwoo Lee. An analysis of existing android image loading libraries: Picasso, glide, fresco, auil and volley. *DEStech Transactions on Engineering and Technology Research*, 2017.