Research log

Papers looked into:


Choice of Hardware:
The design and study to be implemented with Unity and with the Microsoft Mixed Reality Toolkit (MRTK) on the Microsoft HoloLens 2. This Head-mounted Display has a resolution of 2048 * 1080 pixels per eye, 8MP camera, and 2 IR cameras for eye tracking.

Set Up:

Unity Set up :
1) Download Unity hub
2) Choose Unity Version (2021.3.11.f1)
3) Install module UWP(Universal Windows Platform), Windows
4) Start new Project 3d core
Mixed reality feature tool:

![Image of Microsoft Mixed Reality Feature Tool](image-url)
Research log

Microsoft Mixed Reality Feature Tool

Discover Features

- MRTK3 (0 of 15)
- Platform Support (0 of 5)
  - Mixed Reality OpenXR Plugin 1.7.0

Last updated 2/23/2023 7:09:48 AM

Show preview releases

Go Back  Get Features
Research log

Microsoft Mixed Reality Feature Tool

Import Features

The Mixed Reality Feature Tool has identified the packages that are required to import your chosen features.

Features

- MRTK Accessibility Early Preview 1.0.0-pre.14
- MRTK Audio Effects 3.0.0-pre.14
- MRTK Core Definitions 3.0.0-pre.14
- MRTK Diagnostics 3.0.0-pre.14
- MRTK Extended Assets 3.0.0-pre.14
- MRTK Graphics Tools 0.4.0

Required dependencies

Go back Validate Import
Research log

Microsoft Mixed Reality Feature Tool

Review and Approve

Manifest

```
{
  "dependencies": {
    "com.microsoft.mrtk.accessibility": "1.0.0-pre.14.tgz",
    "com.microsoft.mrtk.audio": "3.0.0-pre.14.tgz",
    "com.microsoft.mrtk.core": "3.0.0-pre.14.tgz",
    "com.microsoft.mrtk.diagnostics": "3.0.0-pre.14.tgz",
    "com.microsoft.mrtk.extendedassets": "3.0.0-pre.14.tgz",
    "com.microsoft.mrtk.graphics": "1.0.0-pre.14.tgz",
    "com.microsoft.mrtk.input": "3.0.0-pre.14.tgz",
    "com.microsoft.mrtk.spatialmanipulation": "3.0.0-pre.14.tgz",
    "com.microsoft.mrtk.standardassets": "3.0.0-pre.14.tgz",
    "com.microsoft.mrtk.tools": "3.0.0-pre.14.tgz",
    "com.microsoft.mrtk.tts.windows": "1.0.0.tgz",
    "com.microsoft.mrtk.uicomponents": "3.0.0-pre.14.tgz",
    "com.microsoft.mrtk.uicomponents.noncanvas": "3.0.0-pre.14.tgz"
  }
}
```

Files to be copied into the project

```
com.microsoft.mrtk.accessibility-1.0.0-pre.14.tgz
com.microsoft.mrtk.audio-3.0.0-pre.14.tgz
com.microsoft.mrtk.core-3.0.0-pre.14.tgz
com.microsoft.mrtk.diagnostics-3.0.0-pre.14.tgz
com.microsoft.mrtk.extendedassets-3.0.0-pre.14.tgz
com.microsoft.mrtk.graphics-1.0.0-pre.14.tgz
com.microsoft.mrtk.input-3.0.0-pre.14.tgz
com.microsoft.mrtk.spatialmanipulation-3.0.0-pre.14.tgz
com.microsoft.mrtk.standardassets-3.0.0-pre.14.tgz
com.microsoft.mrtk.tools-3.0.0-pre.14.tgz
com.microsoft.mrtk.tts.windows-1.0.0.tgz
com.microsoft.mrtk.uicomponents-3.0.0-pre.14.tgz
com.microsoft.mrtk.uicomponents.noncanvas-3.0.0-pre.14.tgz
```

[Buttons] Compare  ?  Go Back  Approve
Research log

Changing Unity settings:

Build settings:

Switch the platform to that required for Hololens
Change Project Settings:
Assign the MRTK3 default Profile
Change the XR plug in for the windows platforms
Enable the interaction profiles needed in the openXR settings
For the scene setup:
Drag the MRTKInput XR Rig and delete the default camera

To build the application to Hololens:
Build the application from unity to get a solution file
Research log

In the Visual studio

You can find the IP address of the hololens in the Network and security settings

In the visual studio installer make sure to check the desktop development with C++
### Research log

#### Mobile development with C++
Build cross-platform applications for iOS, Android or Windows using C++.

#### Gaming (2)
- **Game development with Unity**
  - Create 2D and 3D games with Unity, a powerful cross-platform development environment.
- **Game development with C++**
  - Use the full power of C++ to build professional games powered by DirectX, Unreal, or Cocos2d.

#### Other Toolsets (6)
- **Data storage and processing**
  - Connect, develop, and test data solutions with SQL Server, Azure Data Lake, or Hadoop.
- **Data science and analytical applications**
  - Languages and tooling for creating data science applications, including Python and F#.
- **Visual Studio extension development**
  - Create add-ons and extensions for Visual Studio, including...
- **Office/SharePoint development**
  - Create Office and SharePoint add-ins, SharePoint...
Found the issue while setting up app in the hololens, for solution tick the individual components as shown Under Universal Windows Platform Development tab.

Enable the Developer mode in the Hololens

1. Select the Update menu item.
2. Select the For developers menu item.
3. Enable Use developer features to deploy apps from Visual Studio to your HoloLens
**Research log**

**Development in Unity:**

1. **Make Keys For the flower text entry layout**
   Make parent and child game objects such that it supports space key, capital letters and number entries

   Make canvas to represent the key names:

2. **Make input text area layout:**
   Make input text area to show the typed keys to the user and also give a reference text to follow.
3. **Map Parent game object Key to Keys script:**
   Keys script handles all the input text entry

Methods:
- **InsertChar**
  ```java
  public void InsertChar(string c)
  {
    int l = inputField.text.length;
    if (l <= reference.Length)
    {
      Debug.Log("number is " + a);
      char x = reference[a];
      char z = c[0];
      if (z == x)
      {
        inputField.text += "<color=green>" + z + "</color>";
      }
      else
      {
        inputField.text += "<color=red>" + c + "</color>";
      }
      a++;
  }
  }
  ```
  handles the insert character. Checks if the character matches the character int the reference string and changes the color of text to red if not matched and green if matched
- **DeleteChar**
  ```java
  public void DeleteChar()
  {
    if (inputField.text.length > 0)
    {
      inputField.text = inputField.text.Substring(0, inputField.text.length - 1);
    }
  }
  ```
  for the backspace key press
- **InsertSpace**
  Handles the space bar key press
  ```java
  public void InsertSpace()
  {
    inputField.text = " ";
  }
  ```
- **CapsPressed**
  Handles the Caps lock
4. Map each key game object to KeyboardButton Scrypt:
   This script handles the input triggers and oncliks.

   ```csharp
   public void CapsPressed()
   {
      if (caps)
      {
         normalButtons.SetActive(false);
         capsButtons.SetActive(true);
         caps = true;
      }
      else
      {
         capsButtons.SetActive(false);
         normalButtons.SetActive(true);
         caps = false;
      }
   }
   ```

5. Make Animation that can be called when key is pressed:
6. Add Object manipulator component to each key and map the Keyboard button scripts to the OnEnter component in the is gaze hovered:
7. Change the color of button and start the animation once the gaze hovered is confirmed

```java
public void Button_click_register()
{
    keys.InsertChar(buttonText.text);
    m_Collider.enabled = false;
    GetComponent<Animator>().Play("Button_rotate");
    m_Collider.enabled = true;
}
```

8. Change the Animation to rotation on triggered and move the animation to OnClickRegister

9. Change the reference text to Phrase sets for evaluating text entry techniques given by I. Scott MacKenzie
10. Make another input Text area for words typed and error display.

11. Write code to get the number of words typed and error, i.e. introduce variables and increment them in code.

```java
if (z == x)
{
    inputField.text += "<color=green>" + z + "</color>";

    // inputField.image.color = Color.red;
}
else
{
    inputField.text += "<color=red>" + c + "</color>";
    error++;}
```

12. Put dwell time in the code for each key
13. Create a Center grey button
14. Create the Center button script that has methods to change the material when gaze ray hovers on it

```csharp
void Update()
{
    // Method to change material when in center button is hovered
    public void In_center_button()
    {
        Debug.Log("in the collider");
        button.GetComponent<SpriteRenderer>().material = In_center;
    }

    // Method to change material when out of center button
    public void Out_center_button()
    {
        button.GetComponent<SpriteRenderer>().material = Out_center;
    }
}
```

15. Map the script to the on-enter and on-exit function calls in the MRTK events of the center button
16. Create the other buttons to complete the layout of the flower Text entry

17. Create a clear button to allow the user to restart the study if need be

18. Create a program to enable the clear button functionality

```java
1reference
public void Clear()
{
    inputfield.text = "";
    j = 0;
    wordcnt = 0;
    error = 0;
    countInput.text = "Word Count : " + wordcnt + ", " + "<color=red>" + "Error : " + error + "</color>";
}
```
19. Map the Clear function to the KeyboardButton script

```csharp
public void Button_click_register()
{
    if (!String.Equals(gameObject.name.ToLower(), "Spc".ToLower()) && !String.Equals(gameObject.name.ToLower(), "clear".ToLower()))
    {
        keys.InsertChar(buttonText.text);
    }
    else
    {
        if (String.Equals(gameObject.name.ToLower(), "clear".ToLower()))
        {
            keys.Clear();
        }
        else
        {
            keys.InsertSpace();
            button.GetComponent<SpriteRenderer>().material = SelectMaterial;
            //m_Collider.enabled = false;
            GetComponent<Animator>().Play("Button_rotate");
        }
    }

    timer = 1.00f;
    //select_flag = false;
    //m_Collider.enabled = true;
```
Research log

User Study:
- Use the Borgs scale for a rating on exertion and eye fatigue.

![Borg CR10 Scale](image)

- Interview questions to understand the demographic including familiarity or previous usage of Augmented reality devices and eye-tracking applications.
- Making the user instructions and ideas on getting the user familiar with the interface and the device i.e Hololens 2.
- Informing users of the timer and the goal to achieve.
- Analyzing the data from the application i.e the words typed and the error to get the error rate and the words per minute.
- Analyzing the data from the interview and ratings from the user.

Ex User data

<table>
<thead>
<tr>
<th>Dwell times</th>
<th>Words Per minute</th>
<th>Error rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Research log**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4 seconds</td>
<td>3.5</td>
<td>0</td>
</tr>
<tr>
<td>1 second</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>0.5 seconds</td>
<td>11</td>
<td>5</td>
</tr>
</tbody>
</table>

**Interaction variations**

1. Layouts and interface methods tested.
   a. Rotating Flower text entry.
   b. Standard Gaze typed keyboard.
   c. Air typing keyboard of hololens.
2. Different Dwell times tried on the Rotating flower text entry layout.
   a. 4 seconds (for trial and familiarizing the interface to user)
   b. 1 second (hypothesized to be the best dwell time for regular users)
   c. 0.5 seconds (hypothesized to be done well by advanced users)