PROJECT
REPORT

FAZE

Tab switching & Window switching
with
Gaze & Facial gestures

VARUN VIKASH KARTHIKEYAN

guided by
Prof. Dr. Michael Kipp
Augsburg University of Applied Sciences

Abstract

In an attempt to explore the domain of using the face as a modality for desktop control, in this project, we use gaze input and facial gestures(tilt head, raise eyebrows, lean backwards) for two common desktop control operations - Tab switching(within application) and Window switching(between application).

A working prototype was built using an eye tracker to collect gaze input and a face detection library to detect facial gestures, both connected to a custom-built dummy desktop and some initial response was taken by showing it to a few people.

Contents

Page	Content
4	Motivation
5	Related Work
5	Target User Group
	Concept Description & Rationales
7	1. Tab Switching
9	2. Window Switching - to adjacent window
11	3. Window Switching - from overview
	Technical Realization
14	1. Dummy Desktop
15	2. Gaze Point Tracking
16	3. Facial Gesture Detection
17	Visual Design of Prototype
19	How can it be tested?
20	Initial Response
21	Future Extension
22	Acknowledgement

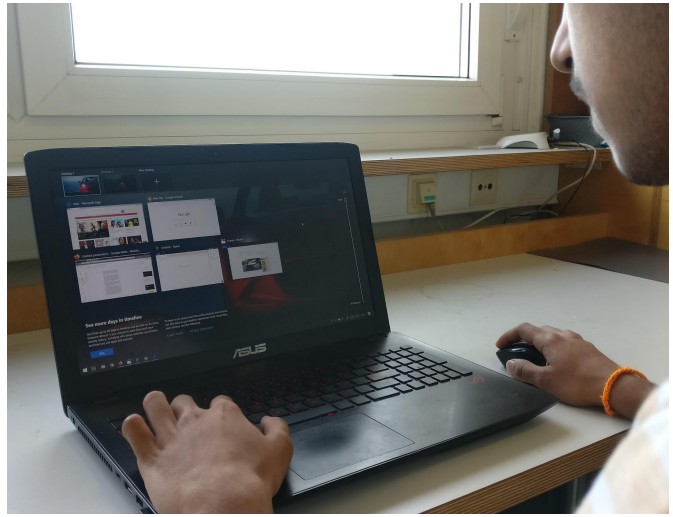
Motivation

In cases where there is a constant need to switch between applications while working on a computer, like for example, writing the code for a website where there is a constant need to switch to the browser from the code to view the preview of the webpage.

In such cases, people who either do not have access to a trackpad or simply prefer to use the mouse do not have the option to input certain multi-tasking inputs like the three-finger swipes on the trackpad to switch between application and hence have to rely on heavily moving the mouse around to get the job done. Even in the case of using the trackpad, switching to an application that is not adjacent(not recently opened) to the current application is time consuming.

For the case of switching between tabs on say, a browser, the tabs are usually located on the top of screen and the pointer is usually somewhere around the centre of the screen and for a quick peek into another tab, there is a considerable amount of mouse movement that is to be endured.

In such scenarios, bringing in the face as a modality for interaction with the computer in the form of either only a facial gesture as a multi-tasking command or using gaze to locate the target and an appropriate facial gesture to select the target can have multiple advantages. One, since there is no involvement of the hands in the entire process, and hence they can stay at their place to continue with the next task. Two, using the face can prove to be faster than using the hand. Three, interactions like leaning back to see the overview are a step in the direction of making the interaction with the computer into something that is more humane or more closer to the real word.



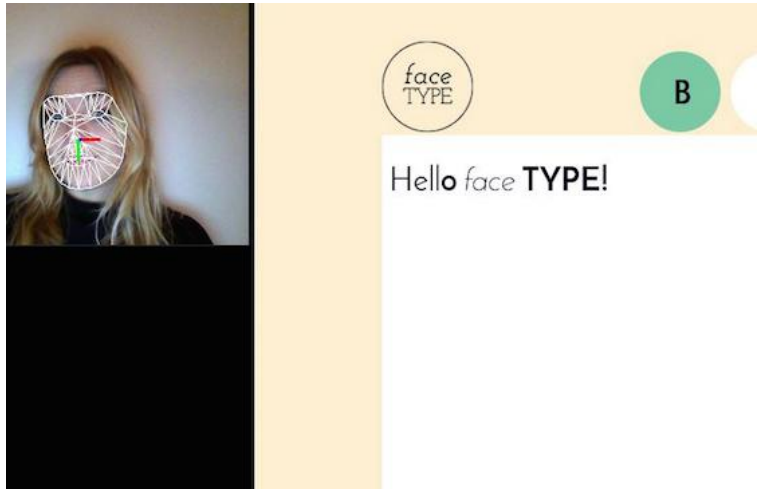
Related Work

faceTYPE

Student project that tries to implement facial gestures in a text editor. The gesture commands are as follows:

- Eyebrow raise - Bold
- Tilt head right - Italics
- Purse lips - Underline

Project Link
http://michaelkipp.de/interaction/projects/?/201516w/FaceType/



Target User Group

Since the nature of the project is exploratory, we do not have any specific profession or purpose as the focus and consider the average computer user as the target user group and not any particular group of power users.

Concept Description & Rationales

There are three kinds of interactions that are proposed in this project:

1. Tab switching
   Gaze input is used to select the tab and eyebrow raise to open it.

2. Window switching - to adjacent window(left/right)
   Head tilt to right or left side is used to switch to the window on the right or left respectively.

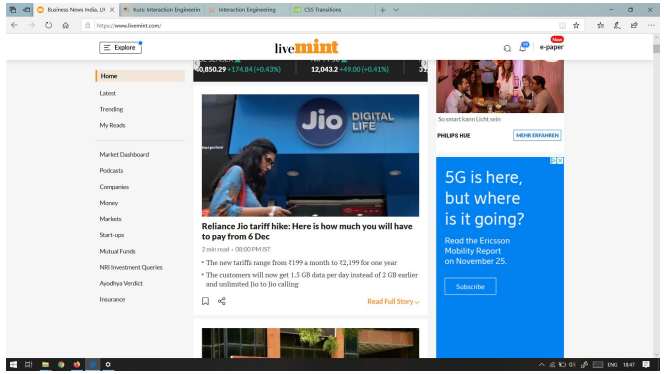
3. Window switching - from overview
   Lean back to go to overview of all open applications, use gaze input to select the window and eyebrow raise to open it.

1. Tab Switching

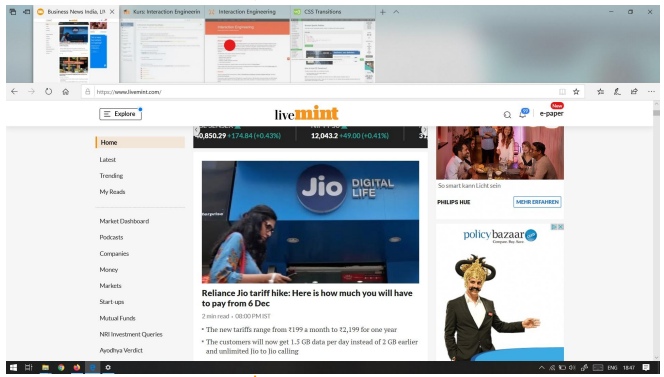
When the user is inside an application say, a browser, it looks like a normal browser window until the user gazes at the part of the window containing the tabs.

Once the user's gaze goes into the area containing the tabs, the tabs get extended downward to show a preview of the content of the tabs along with a visual feedback of the gaze point displayed in real-time, which stays for as long as the gaze point moves out of the area(now extended due to the tab preview) containing the tabs.

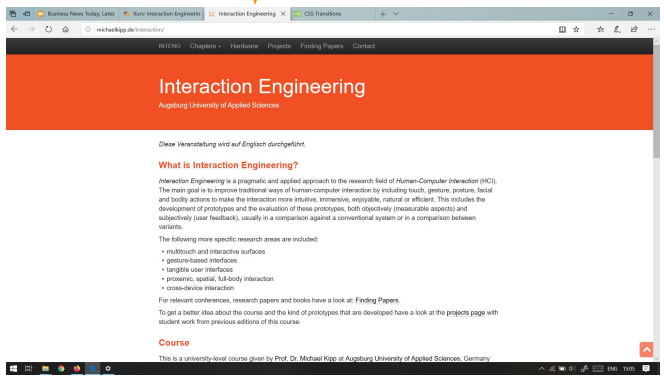
Concept video
https://drive.google.com/open?id=1FCsDM_6OYv2UJ9ucRnx5hB6IUV7QM45l



Gaze at tab area



Raise eyebrows



1. Tab Switching

Tab preview

Showing the tab preview not only gives information to the user on what is inside the tabs, but more importantly, increases the area covered by the tab on the screen, making it an easy target to hit with gaze input.

Visual feedback of gaze point

While hover effect might be sufficient to show which tab the user is currently gazing at, it is very important to let the user know when they have to interact with their gaze. So now, whenever the user sees the visual feedback of their gaze point on the screen, they know that they have to interact with their gaze.

Since the gaze point doesn't always stay on the screen and is shown only at the time of interaction(looking at area containing tabs), there is no question of the displaying of the gaze point interfering with the user's experience of viewing the contents on the page.

Eyebrow raise for selection

While not being the most natural mapping, it is something that can be learned easily and is very less energy consuming. Since selection is an interaction that can end up being performed very frequently and its less energy consuming nature will help avoid fatigue.

2. Window Switching

to adjacent window (left/right)

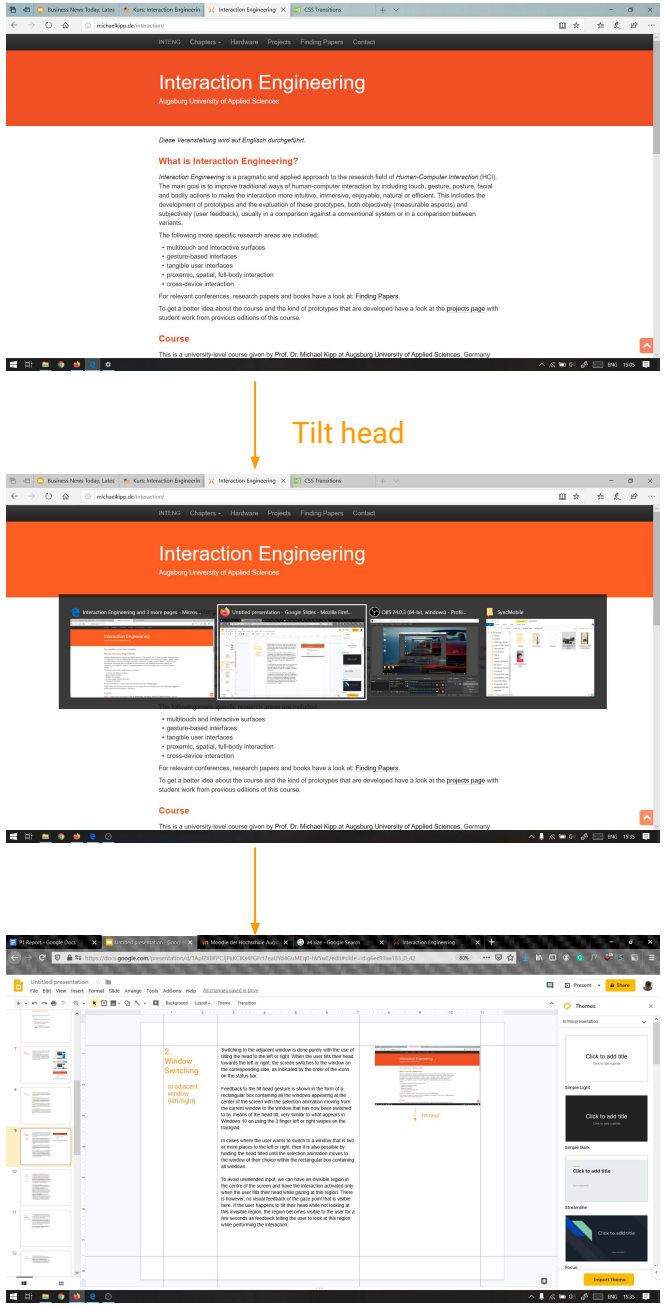
Switching to the adjacent window is done purely with the use of tilting the head to the left or right. When the user tilts their head towards the left or right, the screen switches to the window on the corresponding side, as indicated by the order of the icons on the status bar.

Feedback to the tilt head gesture is shown in the form of a rectangular box containing all the windows appearing at the center of the screen with the selection animation moving from the current window to the window that has now been switched to by means of the head tilt, very similar to what appears in Windows 10 on using the 3 finger left or right swipes on the trackpad.

In cases where the user wants to switch to a window that is two or more places to the left or right, then it is also possible by holding the head tilted until the selection animation moves to the window of their choice within the rectangular box containing all windows.

To avoid unintended input, we can have an invisible region in the centre of the screen and have the interaction activated only when the user tilts their head while gazing at this region. There is however, no visual feedback of the gaze point that is visible here. If the user happens to tilt their head while not looking at this invisible region, the region becomes visible to the user for a few seconds as feedback telling the user to look at this region while performing the interaction.

Concept video
https://drive.google.com/open?id=1M4bTNnjdPy_IALFXCsW1cJh8lDza6M2q



2. Window Switching

to adjacent window (left/right)

Slight deviation from Windows 10 implementation

The only difference is that, while Windows 10 shows the windows arranged in the order of most recently opened from left to right, we arrange them in the same order as that of the icons displayed on the status bar. This is done because in the case of constantly switching between only two applications, going the Windows 10 way would mean tilting the head in the same direction and that would feel very unnatural. In addition to that, arranging the windows in the same order of the icons of the status bar also lets the user to change the order according to their preference by changing the order of the icons.

Alternative feedback considered

Since the current form of feedback requires waiting for the window changing animation to be completed before the user is taken to the new window, it does have a time gap before the user can start working on the new window.
In an attempt to avoid this, another method of showing feedback was considered wherein the new window directly appears on head tilt input and feedback is shown in the form an animation for the change of the ‘selected’ status of the icons on the status bar. Here, the ‘selected’ highlight jumps from the current icon well above the status bar into the icon of the window which was just selected.
But having both the way of interaction(head tilt) and the feedback(jumping icon select highlight) as something the user is not used to, it might hamper their relatability to the existing system. Moreover, since our target group is not the power user, it was decided to go with the feedback that resembles the Windows 10 implementation for the sake of better relatability.

3. Window Switching

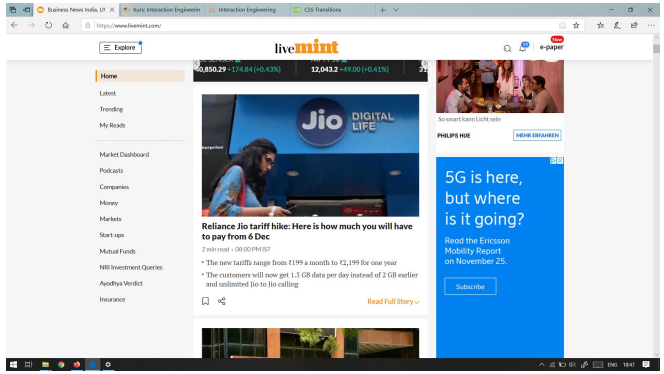
from overview

Window switching by going to the overview involves two steps - one, leaning backwards to go to the overview and two, using gaze and eyebrow raise to select the desired window and open it, just like in the case with the tab switching.

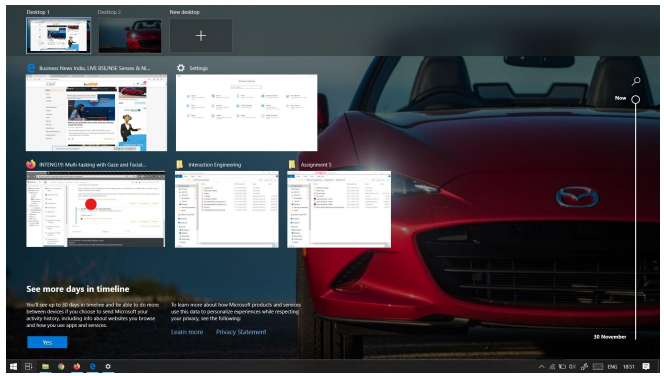
When the user leans backwards, they are taken to the overview page containing all the windows that are currently open, just like the 3-finger swipe up implementation in Windows 10. Here again, to avoid unintended input, an invisible gaze region at the centre of the screen can be implemented as in the previous case and only when the user leans back while gazing at that invisible region, the input is recorded.

Once on the overview page, visual feedback of the gaze point appears and then the user can raise their eyebrows while gazing on the desired window to open it.

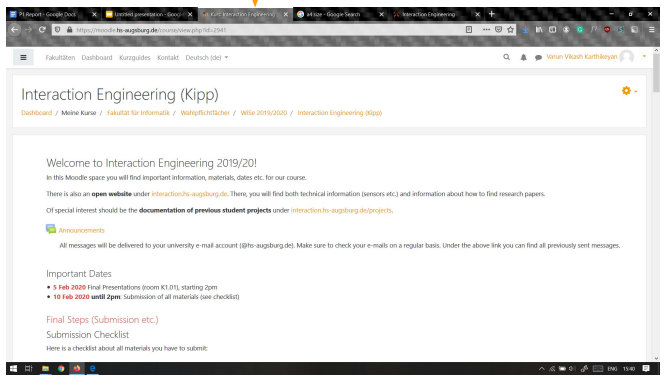
Concept video
https://drive.google.com/open?id=1Yaf_ay5P89G1lsAGXMTCBV-1WlElе8p8



Lean back



Gaze on desired window
+ Eyebrow raise



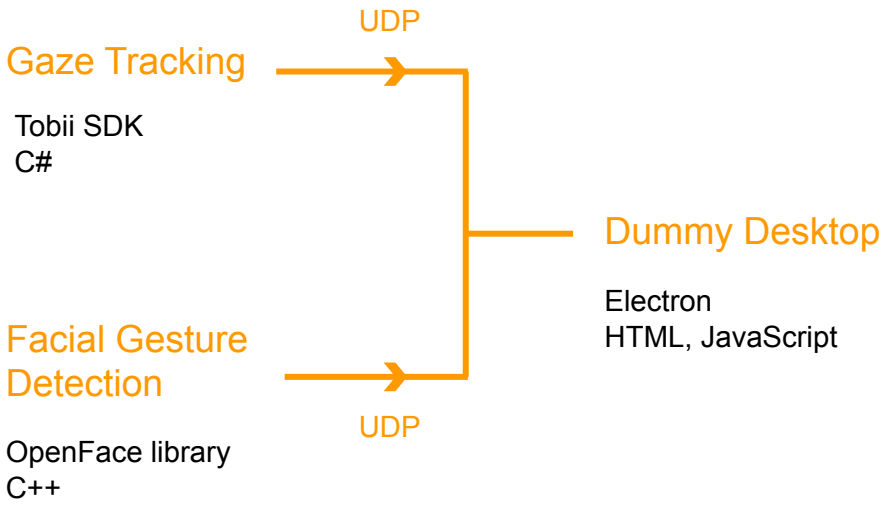
3. Window Switching

from overview

Alternative way of selection considered

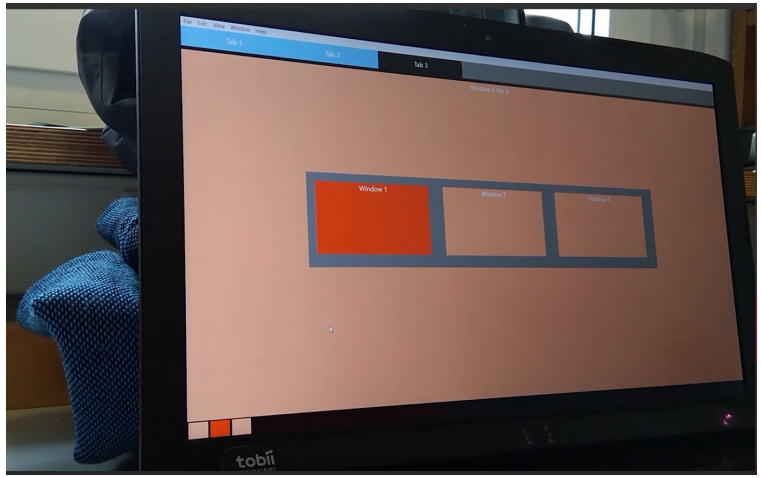
An alternative to opening the desired window with raising eyebrows would be to lean forward while gazing at the desired window to open it. But it was decided to stick to raising eyebrows for selection for the sake of two reasons - one, to maintain consistency in the mode of selection across interactions(tab selection) and two, moving the gaze pointer on the screen while leaning back would be difficult both for the user and also for the eye tracker to detect the gaze position.

Technical Realization



1. Dummy Desktop

Electron, which is a barebones stripped-down version of a browser was used to build the dummy desktop and as a result, the dummy desktop is essentially a website, which is much easier that making GUI on languages like C++ or C#. The code for the dummy desktop was written using html and javascript.



2. Gaze Point Tracking

Eye tracking was done with the help of the Tobii Eye Tracker 4C, which can be attached placed at the bottom of the display and connected to the computer via USB.

Here, help was taken from a project on github called “tobii-electron-streaming” by frocker. The project made use of the C# SDK by Tobii to output the real-time gaze coordinates to an UDP port and the same was received on an electron app using javascript.

Link to project on Github
https://github.com/frocker/tobii-electron-streaming

https://www.techadvisor.co.uk/review/accessories/tobii-eye-tracker-4c-review-head-eye-tracking-make-4c-one-of-best-available-3652454/

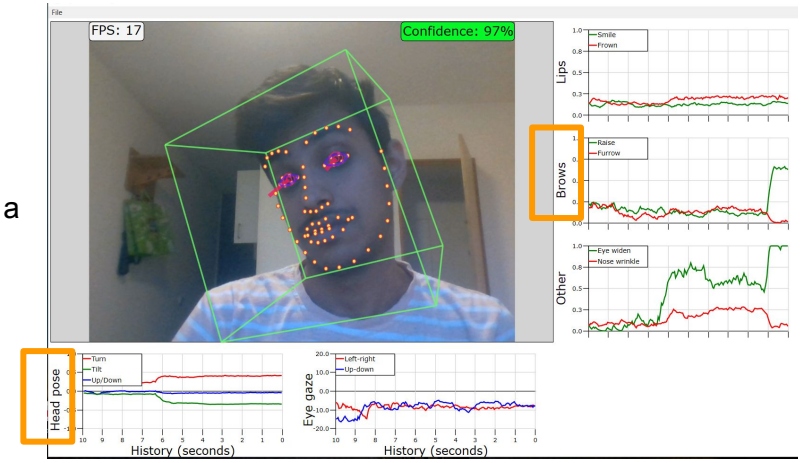
3. Facial Gesture Detection

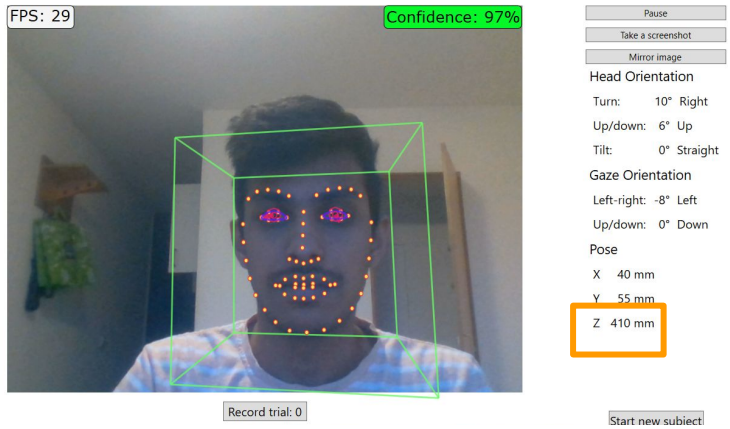
Detection of facial gestures - eyebrow raise, head tilt and leanback was done with the help of face detection library “OpenFace” by TadasBaltrusaitis, which is also available on github. It is a C++ library that uses OpenCV’s facial landmark detection along with some computation and is able to give numerical values for the level of eyebrow raise and head tilt in a sample C++ file that was available along with the library.

For detecting leanback, initially, the distance of the face from the camera was calculated by measuring the volume of the cube that is drawn around the face by the library. However, on trying it out with users, it proved to be unreliable as the user had to be really close to the screen before leaning backwards for there to be a significant change in the volume of the cube, which some users felt was not practical for then.

As a result, another part of the same library that was directly able to give the X, Y and Z distances of the object from the camera was used.

Tweaks were made to the sample C++ file generating these values so that the values could be sent over UDP to the dummy desktop, which was then received there using javascript.





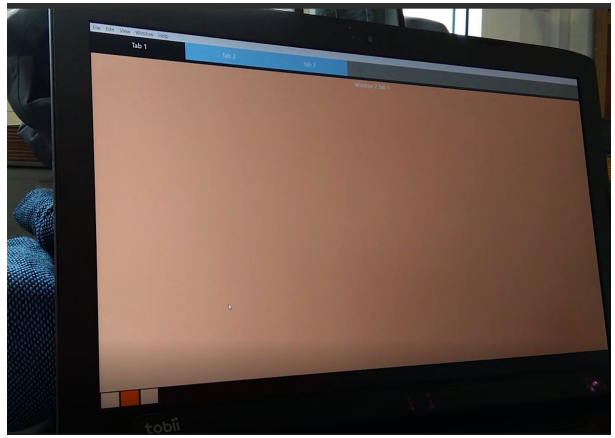
Link to library on Github
https://github.com/TadasBaltrusaitis/OpenFace

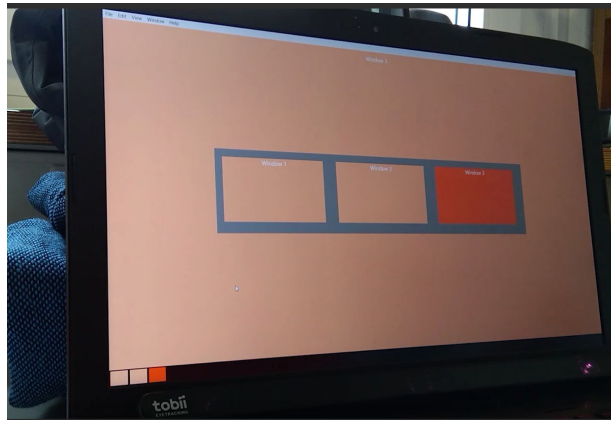
Visual Design of Prototype

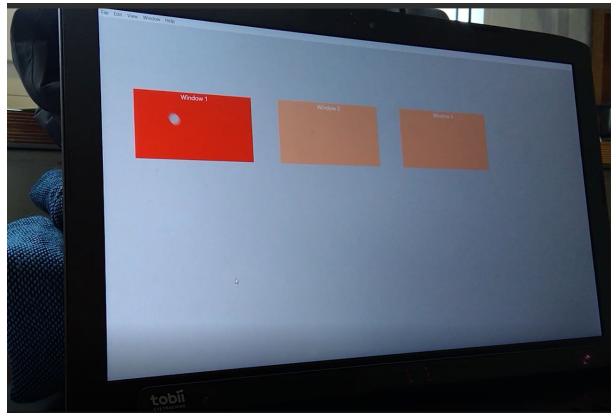
In the dummy desktop, all the elements were reduced to abstract elements so that there is no distraction of the other elements on the screen, which might influence the time taken by the user to perform a task.

In addition to that, all similar elements(Tabs, windows, content of the windows) were made to look similar(color, size, etc) and differentiation was made only with the help of the text on them indicating their name. This was done so that the contents of the tab/window do not have any influence.

Animations were however included in all cases as animations for hover, window shrinking into overview, switching between windows are very important to give the user feedback of what is happening and also to make the dummy desktop relatable to a real one.







Video Demo of Prototype
https://drive.google.com/open?id=1S0P5DdBmiaytAw2AfheiuBlp17zSRtG7

How can it be tested?

The prototype can be tested by comparing it to both a mouse and a trackpad on two grounds - time taken, which is quantitative and convenience, which is qualitative.

In the test setup, the user is explained about the new interactions and is given some 2-3 minutes to try them out. Once that is done, they are asked to perform the same set of tasks using Faze, Mouse and Trackpad separately. The tasks are will be like:

- Go to Tab 3
- Switch to Window 1 by going to the overview

The time taken for each task is measured and at the end of the test, the users are also given a questionnaire where they are asked to rate the three modes on interaction on a scale of 1-5 on the basis of convenience.

Initial Response

A couple of fellow students we asked to try out the prototype and tell what they felt about it.

At this stage that some pointed out that they had to stay closer than their natural position to the screen before leaning backwards to go to the overview for the input to be detected.

With one student wearing spectacles with a large frame, his eyebrow raise was not always detected.

Apart from that, while no one had any other issues with using the prototype, one student did mention that he found the eyebrow raise to select being a bit ‘unnatural’.

Future Extension

Apart from trying it out for other desktop controls like opening a folder, etc., the same set of interactions could also be extended to a video streaming application like Youtube with the following mappings:

Tilt head
skip/rewind

Lean back
exit fullscreen

Gaze + Eyebrow raise
select from suggestions

Gaze interaction area

Acknowledgement

Special thanks to

- **Jayasurya Salem Sudakaran**
- **Sourav Dutta**

for help with tackling the technical difficulties encountered during the technical realization phase.