Circular Menu

A Future User Interface Project

MARC-OLIVIER TRICOT, MATTEO BADARACCO AND DAVID BERGER

{marc-olivier.tricot, matteo.badaracco, david.berger}@unifr.ch

May 2018

Study supervisors:

Prof. Dr. Denis Lalanne

UNIVERSITÉ DE FRIBOURG
UNIVERSITÄT FREIBURG

HUMAN-IST
Department of Informatics
University of Fribourg
(Switzerland)

Table of Contents

1 Introduction

Websites are often filled with content. Cramming everything on the main page is obviously not possible, so it is needed to classify everything on different pages and then being able to access them. This was solved by creating navigation menus. From that small part of the website, usually at the top, people could easily access whatever part of the website they wanted, with help from drop-down sub-menus when needed.

Since then, nothing much as changed to access this navigation menu. Be it with the mouse, or the keyboard, or a mix of both, it seems pretty widely accepted that this is the definite version of menus and accessible them.

However, with the advent of different technologies, it might be time to change the way people interact with that pretty fundamental element of websites (and surely multiple others). Our group thought it could be a good idea to use the Eye Tracker to interact with the navigation menu, which would make it faster.

The standard menu is not suitable to eye tracking because of its small size, but creating a large, circular menu at the center of the screen would be a good way of offering an easy to see way of choosing rapidly. Hence we started working on the Circular Menu, which aims to offer a easy way of accessing the navigation menu of website with the aid of eye trackers.

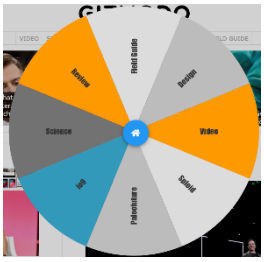
2

The Circular Menu

To make it easier for users to access the elements of the menu, we created the Circular Menu. The Circular Menu is composed of three core elements, the Extension, the Circular Wheel and the Eye Tracker. We will now detail these three elements.

2.1 The Extension

We implemented an extension on Google Chrome to use the Circular Menu. The aim here is to provide an easy access to the Circular Menu and being able to switch it on and off just as easily. In addition, there could be some other possibilities accessible through there, such as hiding the original menu bar.



When the application is activated, the user can see a small blue semicircle at the bottom of the screen, which can be activated with the mouse or the keyboard shortcut Ctrl+M.

2.2 The Circular Wheel

The first part of the creation of the wheel is the identification of the navigation bar. As every website (or almost) have their own way of disposing their menu, we had to manually identify the id or class of the menu then add it to the main algorithm in charge of building the menu. Creating an automatic retrieval function was deemed too time-consuming and over our abilities because of the sheer variety of websites.

Once the navigation menu has been identified, the program extracts the content of the menu and places them in the Circular Menu at the center of the screen. The different options are placed in equally-sized slices of the circle and resized so they are easy to read quickly. By clicking on them or using the combination of the Eye Tracker and keyboard, one can access the pages that were originally linked in the menu. The text's size is made to be responsive to avoid having names to small or overflowing. At the center is the home button.

When the menu has too many elements, we limit the amount at 9 visible options, and the others are placed in an "extra" menu represented by an ellipsis (…). Those options can be seen by simply hovering over the ellipsis, and are displayed as another circle around the original one. This limits the total of options at 28, but we find it to be large enough to accomodate the vast majority of websites.

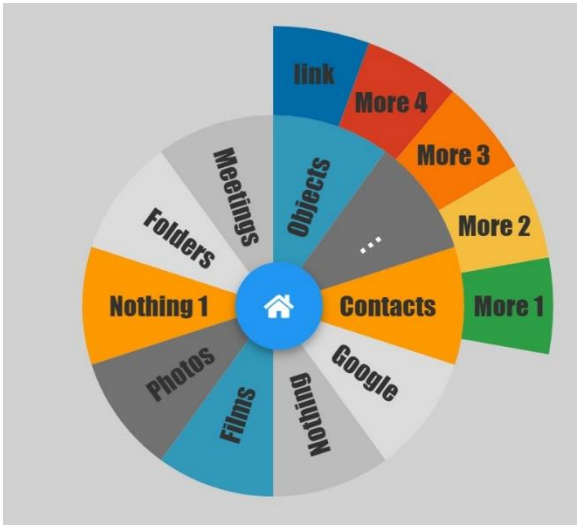


Figure 1: Cercle Menu with open ellipsis

Another important element is how we deal with sub-menus. As quite a few websites have sub-menus (such as drop-down menus inside the first level), it is necessary for us to offer a way to access them. We were originally planning to have it go on the side, but that causes multiples problems, like overlapping with the ellipsis and the impossibility to have too many sub-menus because of lack of space. Instead, we simply made it replace the current menu with the new options, which lets us have infinite sub-menus without any cost in space. To return to the starting menu, one can simply click on the home icon at the center.

2.3 The Eye Tracker

The eye-tracking device we use was an Eye Tracker 4C from Tobii. It was disposed at a stable emplacement and used as a modular interface to help the users get faster to their destination. It follows the pupils and shows a translucent bubble at the position on the screen where the user is watching. Our code follows the bubble and moves the mouse at the position and clicks when asked to.

As the Tobii SDK doesn't support JavaScript or any other web-languages yet, it had to be implemented separately. We used Visual Studio and the .NET language (often used in Windows applications), both developed by Microsoft. We also had to import the Tobii SDK with a NuGeT Manager. Luckily, the Tobii SDK is very well documented and "user-friendly". We managed to find solutions to our problems thanks to it. The code is viewable in the appendix.

3 Evaluation

3.1 The Experiment

To figure out if the Circular Menu brings anything to the users, we designed an experiment that we tested on a small sample of web users. The aim of the experiment was to determine how the users react to the Circular Menu, how comfortable they feel using it and also, wether they prefer the Circular Menu with or without the Eye Tracker.

The experiment was designed to last less than 15 minutes. The initial part would be a couple minutes of explanations about the tasks, the privacy of the data and why we were doing the experiment. After that we let the user play a game in order to getting him use to the Eye Tracker. Then another couple minutes trying out the Eye Tracker and Circular Menu on a certain website, before switching to another website where they had to complete some simple tasks like accessing some elements in inside the sub-menus. This would last for 3-4 minutes, and the last part would be filling in the questionnaire given out at the end.

The survey was mostly quantitative elements with a few qualitative ones. It asked about the usability and speed of different versions of the menus (with and without the Eye Tracker), and whether the users would use it if it became easily accessible, with a little statistical data (age, sex, etc) at the end. We asked 11 different people to test our Circular Menu and answer the survey. We also conducted some quantitative measures for the speed of the users while completing the simple tasks, and measured them with a manual chronometer.

3.2 The Evaluation

The survey was made with Google Forms, and was composed of three sections. The first section is a general overview about how the users liked the Circular Menu. The first question was an evaluation which compared it a linear scale with the standard menu with 1 being liking the

Circular Menu the best and 5 being liking the standard menu the best. The second question was a qualitative question which asked for the favourite element of the menu. The third question was a simple choice asking which was the favourite between the standard menu, the Circular Menu with the Eye Tracker and the Circular Menu without.

The initial results were pretty clear, as the vast majority (90.9%) of our test subjects liked the Circular Menu with the Eye Tracker more than the standard menu, with 36.4% liking it a lot more (1 on the scale). From the qualitative question, we could see that the speed of interaction and the ease of access thanks to the Ctrl+M shortcut were the most recurrent reasons why. The third question gave slightly less enthusiastic results, as 27.3% said they their favourite was nonetheless the standard menu. Some comments to explain this was it was too new/confusing for some users.

The second part concerned itself on whether the Circular Menu was fast and user-friendly. To see the influence of the eye tracking, we decided to separate the questions between the Circular Menu with and without the Eye Tracking.

On the speed side, 81.8% of users said that it was faster than the standard menu with the Eye Tracker, the others saying it was the same. Without the Eye Tracker, the results were less positive, with 72.7% saying it was faster, 9.1% the same speed and 18.2% slower.

For the user-friendliness, we noticed again a change between both cases, but mostly at the extreme. With the Eye Tracker, 72.7% find it easier to use than the standard menu (54.5% find it much easier), 18.2% the same as the standard menu and 9.1% harder. Without the Eye Tracker, 63.7% find it easier (but only 27.3% find it much easier), 27.3% find it of the same ease and again 9.1% find it harder.

The final question in this section asked whether the users would use the Circular Menu if it was made easily accessible, and the vast majority (90.9%) said yes. Only 9.1% said they would prefer the standard menu, and the same amount said they would rather use the version without the Eye Tracker.

The last section was some statistical data about the respondents, with 63,6% being men and 36,4% women and no others. The vast majority of respondents (81.8%) were between 20 and 29 years old, with 9.1% in both the 30 to 39 category and the 40 to 49 category. The overwhelming majority (90.9%) visited websites 6 times or more per week, with only 9.1% who did 4 or 5 times.

With the speed measurements, we wanted to check if our Null Hypothesis, that the Circular Menu with the Eye Tracker does not bring faster navigation for users on average, held true. The

Alternative Hypothesis (H1) is that the Circular Menu with the Eye Tracker is indeed faster. In our measurements (which include two the results of ours which we forgot to remove before anonymising), we found that the users were in average 557 milliseconds faster. We passed these results through the p-test with a 250 milliseconds bias to provide for human error. We found that we were unable to reject the Null Hypothesis with sufficient confidence, as we had less than 80%.

3.3 Results

From these answers, we can see that the Circular is very popular and generally rated more usable and faster than the standard menu, which would suggest a possible market to be exploited in the future. The Eye Tracker is a rather fundamental part of it however, which might make it necessary to wait until it becomes more widely used or even integrated in most portable computers, tablets and screens.

Furthermore, we have measured a small speed increase in the realisation of small tasks, which show the potential of the time-saving that could be done, especially as the users and the technology progress in their adaptation to these new applications. However, we couldn't reject the Null Hypothesis, which gives us only the users answers (feelings) as a motivation to promote the product.

There is a need for more extensive surveys however, as ours is skewed towards young and generally tech-aware people, which might not represent the market properly, even if it at least has niche potential. It is also a small sample size; better results might come out of a large-scale experiment.

4

Conclusion and Future Work

In conclusion, we can say that the Circular Wheel is a good idea. The user return was positive, they find it fast, easy to use and useful. Therefore, there is definitely some potential for implementing this idea at a larger scale.

As for future works, there are a few things that come to mind. The first one concerns the Circular Menu and would be a requirement to make it a widely-used application. Making the extension compatible with all website (or a majority) would be a large progress, but that would require either a lot of work and/or advanced knowledge of CSS/JS. An idea we had to solve that is to have the user select the menu when starting the extension, which should simplify the automated finding.

Another practical idea would be the implementation of the Circular Wheel for Bookmarks, which is easier to automatize (browser-based instead of website-based), less easily accessible to users and also quite practical for usual navigation. We also thought about adding options to the extension menu, such as hiding the original menu after activation.

All in all, this was an interesting experiment with some potential with the future, and we are glad to have taken this class and had the possibility to experiment with the Eye Trackers.

5 Appendix

The complete project can be found on Github: https://github.com/Wuschelbueb/FUI

5.1 Eye Tracking Code

```
using System;
using System.Diagnostics;
using System.Runtime.InteropServices;
using System.Windows.Forms;
using Tobii.Interaction;
using Tobii.Interaction.Framework;

namespace ConsoleApp2
{

    public class Program
    {
        private const int WH_KEYBOARD_LL = 13;
        private const int WM_KEYDOWN = 0x0100;
        private const int PM_REMOVE = 1;

        private delegate IntPtr LowLevelKeyboardProc(int nCode, IntPtr wParam,
IntPtr lParam);
        private static LowLevelKeyboardProc handler = HookCallback;
        private static IntPtr previousHook;
        private static bool endPressed = false;

        //some declartions
        private static Host _host;
        //private static Rectangle _chromeRectangle;

        //set cursor position
        [DllImport("user32.dll")]
        private static extern bool SetCursorPos(int X, int Y);

        //get cursor position
```

```
        [DllImport("user32.dll")]
        public static extern bool GetCursorPos(out POINT lpPoint);

        //simulate left mouseclick
        [DllImport("user32.dll")]

        public static extern void mouse_event(int dwFlags, uint dx, uint dy,
int cButtons, int dwExtraInfo);


        private const int MOUSEEVENTF_LEFTDOWN = 0X0002;
        private const int MOUSEEVENTF_LEFTUP = 0x0004;

        [StructLayout(LayoutKind.Sequential)]
        public struct POINT
        {
            public int X;
            public int Y;

            public POINT(int x, int y)
            {
                this.X = x;
                this.Y = y;
            }
        }

        //allows to run the program in the background
        //source https://social.msdn.microsoft.com/Forums/SECURITY/en-
US/1f05497d-d036-49c8-bb10-6cd4ceb7f725/run-console-in-
background?forum=csharpgeneral
        [DllImport("user32", CharSet = CharSet.Auto, SetLastError = true)]
        private static extern IntPtr SetWindowsHookEx(int idHook,
LowLevelKeyboardProc lpfn, IntPtr hMod, uint dwThreadId);

        [DllImport("user32", CharSet = CharSet.Auto, SetLastError = true)]
        private static extern IntPtr CallNextHookEx(IntPtr hhk, int nCode,
IntPtr wParam, IntPtr lParam);


        [StructLayout(LayoutKind.Sequential)]
        public struct NativeMessage
        {
            public IntPtr handle;
            public uint msg;
            public IntPtr wParam;
            public IntPtr lParam;
            public uint time;
            public System.Drawing.Point p;
```

```
        }


        [DllImport("user32")]
        static extern bool PeekMessage(out NativeMessage lpMsg, IntPtr hWnd,
uint wMsgFilterMin, uint wMsgFilterMax, uint wRemoveMsg);


        private static IntPtr HookCallback(int nCode, IntPtr wParam, IntPtr
lParam)
        {
            if (nCode >= 0 && wParam.ToInt32() == WM_KEYDOWN)
            {
                var key = unchecked((Keys)Marshal.ReadInt32(lParam));

                switch (key)
                {
                    case Keys.F7:
                        {
                            JumpToPosition();
                            Console.WriteLine("left click detected");
                        }
                        break;
                    case Keys.F8:
                        {
                            Console.WriteLine("<END> detected");
                            endPressed = true;
                        }
                        break;
                }
            }

            return CallNextHookEx(previousHook, nCode, wParam, lParam);
        }

        //main starts here
        public static void Main(string[] args)
        {
            _host = new Host();

            previousHook = SetWindowsHookEx(WH_KEYBOARD_LL, handler,
IntPtr.Zero, 0);

            Console.WriteLine("Hook installed...");

            while (!endPressed)
            {
                NativeMessage m;
                PeekMessage(out m, IntPtr.Zero, 0, 0, PM_REMOVE);
```

```
            }
        }

        private static void JumpToPosition()
        {
            GazePointDataStream gazePointDataStream =
_host.Streams.CreateGazePointDataStream(GazePointDataMode.LightlyFiltered);
            gazePointDataStream.IsEnabled = true;
            gazePointDataStream.Next += (s, eye) =>
EyeLocation(gazePointDataStream, eye);
        }

        public static void EyeLocation(GazePointDataStream stream,
StreamData<GazePointData> eye)
        {
            stream.IsEnabled = false;
            uint X = (uint)eye.Data.X;
            uint Y = (uint)eye.Data.Y;
            Console.WriteLine("CurrentX: {0} CurrentY: {1}", X, Y);
            POINT lpPoint;
            GetCursorPos(out lpPoint);
            int oldX = lpPoint.X;
            int oldY = lpPoint.Y;
            Console.WriteLine("oldX: {0} oldY: {1}", oldX, oldY);
            SetCursorPos((int)(eye.Data.X/1.5), (int)(eye.Data.Y/1.5));
            mouse_event(MOUSEEVENTF_LEFTDOWN | MOUSEEVENTF_LEFTUP, X, Y, 0,
0);
            //SetCursorPos(oldX, oldY);
        }


    }
}
```

5.2 Circular Navigation Code

```
//var for the principal wheel
var wheel;

//used for adding the other menu
var wheel2;

// var to store if the circular navigation is open
var open=false;

//array where store the sub menus/links
var linkTo=[];
var otherItems=[];
var otherItemsHome=[];

//num max of elements in a circular navigation
const numMaxElements=9;
var other=false;

var itemsParentList=[];
var itemsGrandParentList=[];
var itemsHomeList = [];

var height_wheel = $('#wheelDiv').height();
$('#wheelDiv').width(height_wheel);


selectMenu();
// take the html and put the elements in an array of array
itemsHome=items = retriveMenuItems($("#circularNavigation"),true);


wheel = new wheelnav("wheelDiv");


//itemsHome=items = ['title-0', 'title-1', 'title-2', 'title-3', 'title-4',
'5','6','7','8','9'];
createCircularNav(items);
var position = $('#wheelDiv').offset();
var top_wheel = position.top;

window.onload = function () {
  attachSubMenus();
```

```
};


$(window).keydown(function (e) {
  if (e.keyCode == 27) {
    open=false;
    $('#wheelDiv').addClass("notVisible");
    $('#home_btn').addClass("notVisible");
    $('#wrapperOpacity').addClass("notVisible");
  }
});

$(window).click(function() {
  open=false;
  $('#wheelDiv').addClass("notVisible");
  $('#home_btn').addClass("notVisible");
  $('#wrapperOpacity').addClass("notVisible");
});

$('#wheelDiv').click(function(event){
    event.stopPropagation();
});
$('#foo').click(function(event){
    event.stopPropagation();
});
$('#home_btn').click(function(event){
    event.stopPropagation();
});



$.ctrl = function(key, callback, args) {
    var isCtrl = false;
    $(document).keydown(function(e) {
        if(!args) args=[]; // IE barks when args is null

        if(e.ctrlKey) isCtrl = true;
        if(e.keyCode == key.charCodeAt(0) && isCtrl) {
            callback.apply(this, args);
            return false;
        }
    }).keyup(function(e) {
        if(e.ctrlKey) isCtrl = false;
    });
};


$.ctrl('M', function() {
  if(!open){
```

```
    openWheel();
  }
  else {
    open=false;

    $('#wheelDiv').addClass("notVisible");
    $('#home_btn').addClass("notVisible");
    $('#wrapperOpacity').addClass("notVisible");
  }
});

$('#foo').bind('click', function() {
  if(!open){
    openWheel();
  }
  else {
    open=false;

    $('#wheelDiv').addClass("notVisible");
    $('#home_btn').addClass("notVisible");
    $('#wrapperOpacity').addClass("notVisible");
  }

});

$('#home_btn').bind('click', function() {
  wheel.removeWheel();
  wheel = new wheelnav("wheelDiv");
  if(itemsHome.length>numMaxElements){

    itemsHome.splice(numMaxElements,1);

    itemsHome=itemsHome.concat(otherItems);

  }
  createCircularNav(itemsHome);
  itemsParentList=itemsHomeList;
  attachSubMenus();
});




function openWheel(){
  open=true;
  $('#wheelDiv').removeClass("notVisible");
  $('#home_btn').removeClass("notVisible");
```

```
  $('#wrapperOpacity').removeClass("notVisible");
  positionHomeBtn(other);
}

function setWheelItems(subMenu=false){
  courrentItem=0;
  if(subMenu){
    courrentItem=numMaxElements+wheel2.currentClick;
  }
  else{
    courrentItem=wheel.currentClick;
  }


  wheel.removeWheel();
  wheel = new wheelnav("wheelDiv");
  items=linkTo[courrentItem];

  for(l=0;l<courrentItem;l++){

itemsBeforeCourrent=itemsGrandParentList[l].children('ul').children('li').leng
th;
    for(r=0;r<itemsBeforeCourrent;r++){
      itemsParentList.shift();
    }
  }
  createCircularNav(items);
  attachSubMenus();
}

function goToLink(subMenu=false){
  courrentItem=0;
  if(subMenu){
    courrentItem=numMaxElements+wheel2.currentClick;
  }
  else{
    courrentItem=wheel.currentClick;
  }
  window.location.href = linkTo[courrentItem];
}



function createCircularNav(items){
  wheel.selectedNavItemIndex = null;
  wheel.clickModeRotate = false;
  wheel.titleRotateAngle = 180;
  wheel.hoverPercent = 1.1;
  wheel.animatetime = 1;
```

```
wheel.animateeffect = 'linear';
wheel.colors = colorpalette.gamebookers;

// if there are more items add the "other" item
if(items.length>numMaxElements){
  other=true;
  otherItems=[];
  wheel.wheelRadius=wheel.wheelRadius/1.2;
  while (items.length>numMaxElements) {
    otherItems.push(items.pop());
  }
  otherItems.reverse();

  if(otherItemsHome.length==0){
    otherItemsHome=otherItems;
  }

  items.push(". . .");
  wheel.initWheel(items);
  setTitleAngle(items);
  wheel.createWheel();




  wheel2 = new wheelnav('wheel2', wheel.raphael);
  wheel2.slicePathFunction = slicePath().DonutSlice;
  wheel2.slicePathCustom = slicePath().DonutSliceCustomization();
  wheel2.minRadius = wheel.wheelRadius;
  wheel2.slicePathCustom.minRadiusPercent = 0.75;
  wheel2.slicePathCustom.maxRadiusPercent = 1.1;
  wheel2.sliceSelectedPathCustom = wheel2.slicePathCustom;
  wheel2.sliceInitPathCustom = wheel2.slicePathCustom;
  wheel2.spreaderRadius = 85;
  wheel2.clickModeRotate = false;
  wheel2.clockwise=false;

  wheel2.initWheel(otherItems);

  for(i=0;i<wheel2.navItems.length;i++){
    wheel2.navItems[i].sliceAngle=20;
  }
  wheel2.navItemsContinuous = true;

  wheel2.createWheel();
}
else{
```

```
    wheel.initWheel(items);
    setTitleAngle(items);
    wheel.createWheel(items);

  }


}



function retriveMenuItems(node,firstLevel=false){
  var listOfElements=[];
  var listOfElementsObject=[];

  console.log(node);


  node.children('li').each(function() {
        listOfElementsObject.push($(this));
        listOfElements.push($(this).html());
        itemsParentList.push($(this));
        if(firstLevel){
          itemsHomeList.push($(this));
        }
      });




  var listOfTrimmedElements=[];
  for(s=0;s<listOfElements.length;s++){

    if($(listOfElementsObject[s]).children('a').length==1){
      listOfElements[s]=$(listOfElementsObject[s]).children('a').html();
    }

    trimmedElement=$.trim(listOfElements[s]);
    itemText="";
    counterSpaces=0;
    j=0;
    while(counterSpaces<2&&j<trimmedElement.length){
      if(trimmedElement.charAt(j)!=" "){
        itemText+=trimmedElement.charAt(j);
        counterSpaces=0;
      }
      else{
        itemText+=trimmedElement.charAt(j);
        counterSpaces++;
```

```
      }
      j++;

    }

    listOfTrimmedElements[s]=$.trim(itemText);

  }

  return listOfTrimmedElements;

}


function attachSubMenus(){



  itemsGrandParentList=itemsParentListTemp=itemsParentList;
  itemsParentList=[];

  for(v=0;v<itemsParentListTemp.length;v++){

    if($(itemsParentListTemp[v]).children('ul').length>=1){
      linkTo[v]=retriveMenuItems($(itemsParentListTemp[v]).children('ul'))
    }
    else if ($(itemsParentListTemp[v]).children('a').length==1) {
      linkTo[v]=[$(itemsParentListTemp[v]).children('a').attr("href")];
    }
    else {
      linkTo[v]=[];
    }
  }


  //linkTo=[['http:/www.google.ch'],[],['title-1', 'title-1'],['title-2',
'title-2','title-2','title-2','title-2'],['title-3', 'title-3','title-3']];
  for(var i=0;i<wheel.navItems.length;i++){
    if(linkTo[i]&&i<numMaxElements){
      if(linkTo[i].length==1){
        wheel.navItems[i].navigateFunction = function(){goToLink();};
      }
      else if(linkTo[i].length==0){
      }
      else{
        wheel.navItems[i].navigateFunction =function(){setWheelItems();};
      }

    }
```

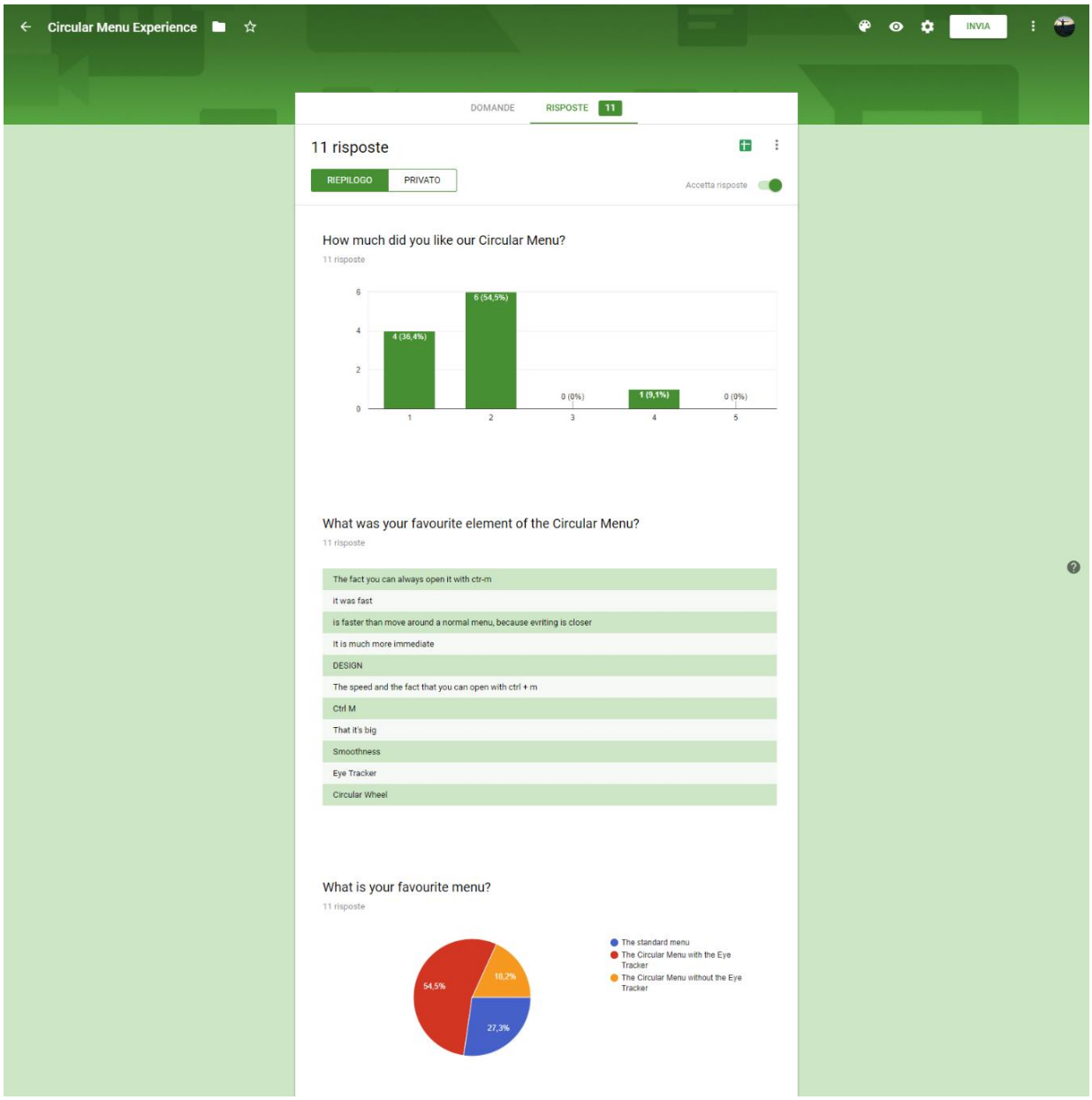
```
    else if (i==numMaxElements) {
      for(y=0;y<wheel2.navItems.length;y++){
        wheel2.navItems[y].navItem.hide();
      }

      var otherSelected = false;
      wheel.navItems[numMaxElements].navigateFunction = function () {

          if (otherSelected) {
            for(y=0;y<wheel2.navItems.length;y++){
              wheel2.navItems[y].navItem.hide();
            }
          }
          else {
            for(y=0;y<wheel2.navItems.length;y++){
              wheel2.navItems[y].navItem.show();
            }
          }
          otherSelected = !otherSelected;

      };
      for(var e=0;e<wheel2.navItems.length;e++){
        if(linkTo[numMaxElements+e]){
          if(linkTo[numMaxElements+e].length==1){
            wheel2.navItems[e].navigateFunction = function(){goToLink(true);};
          }
          else if(linkTo[e].length==0){
          }
          else{
            wheel2.navItems[e].navigateFunction
=function(){setWheelItems(true);};
          }

        }
      }
    }
  }
}

function positionHomeBtn(other=false){


    var height_wheel = $('#wheelDiv').height();

    var width_wheel = $('#wheelDiv').width();


    var viewWidth = $( window ).width();
    var left_wheel = (viewWidth-width_wheel)/2;
```

```
    $('#wheelDiv').css({"left":left_wheel});



    //var left_wheel = position.left;


    $('#home_btn').css({"left":left_wheel+(width_wheel/2)-
35,"top":top_wheel+(height_wheel/2)-35});
    //$('#home_btn').css({"left":viewWidth/2,"top":viewHeight/2});
}



function selectMenu(){
  var circularNavContainer="";
  //Gizmodo
  if($( ".section-nav > .js_sections").length>0){
    $( ".section-nav > .js_sections").attr('id', 'circularNavigation');
  }
  //Treccani
  else if ($( ".menu-first_level").length>0) {
    $( ".menu-first_level").attr('id', 'circularNavigation');
  }
  else if ($( ".level").children('ul').length>0) {
    $( ".level").children('ul').attr('id', 'circularNavigation');
  }


}

function setTitleAngle(items){
  if(items.length==2){
    wheel.navItems[0].titleRotateAngle = 0;
    wheel.navItems[1].titleRotateAngle = 180;
  }
  else if(items.length==3){
    wheel.navItems[0].titleRotateAngle = 0;
    wheel.navItems[1].titleRotateAngle = 180;
    wheel.navItems[2].titleRotateAngle = 180;
  }
  else if(items.length==4){
    wheel.navItems[0].titleRotateAngle = 0;
    wheel.navItems[1].titleRotateAngle = 180;
    wheel.navItems[2].titleRotateAngle = 180;
    wheel.navItems[3].titleRotateAngle = 180;
  }
  else if(items.length==5){
    wheel.navItems[0].titleRotateAngle = 0;
```

```
    wheel.navItems[1].titleRotateAngle = 0;
    wheel.navItems[2].titleRotateAngle = 180;
    wheel.navItems[3].titleRotateAngle = 180;
    wheel.navItems[4].titleRotateAngle = 0;
  }
  else if(items.length==6){
    wheel.navItems[0].titleRotateAngle = 0;
    wheel.navItems[1].titleRotateAngle = 0;
    wheel.navItems[2].titleRotateAngle = 180;
    wheel.navItems[3].titleRotateAngle = 180;
    wheel.navItems[4].titleRotateAngle = 180;
    wheel.navItems[5].titleRotateAngle = 0;
  }
  else if(items.length==7){
    wheel.navItems[0].titleRotateAngle = 0;
    wheel.navItems[1].titleRotateAngle = 0;
    wheel.navItems[2].titleRotateAngle = 180;
    wheel.navItems[3].titleRotateAngle = 180;
    wheel.navItems[4].titleRotateAngle = 180;
    wheel.navItems[5].titleRotateAngle = 180;
    wheel.navItems[6].titleRotateAngle = 0;
  }
  else if(items.length==8){
    wheel.navItems[0].titleRotateAngle = 0;
    wheel.navItems[1].titleRotateAngle = 0;
    wheel.navItems[2].titleRotateAngle = 180;
    wheel.navItems[3].titleRotateAngle = 180;
    wheel.navItems[4].titleRotateAngle = 180;
    wheel.navItems[5].titleRotateAngle = 180;
    wheel.navItems[6].titleRotateAngle = 0;
    wheel.navItems[7].titleRotateAngle = 0;
  }
  else if(items.length==9){
    wheel.navItems[0].titleRotateAngle = 0;
    wheel.navItems[1].titleRotateAngle = 0;
    wheel.navItems[2].titleRotateAngle = 180;
    wheel.navItems[3].titleRotateAngle = 180;
    wheel.navItems[4].titleRotateAngle = 180;
    wheel.navItems[5].titleRotateAngle = 180;
    wheel.navItems[6].titleRotateAngle = 180;
    wheel.navItems[7].titleRotateAngle = 0;
    wheel.navItems[8].titleRotateAngle = 0;
  }
  else if(items.length==10){
    wheel.navItems[0].titleRotateAngle = 0;
    wheel.navItems[1].titleRotateAngle = 0;
    wheel.navItems[2].titleRotateAngle = 180;
    wheel.navItems[3].titleRotateAngle = 180;
```

```
    wheel.navItems[4].titleRotateAngle = 180;
    wheel.navItems[5].titleRotateAngle = 180;
    wheel.navItems[6].titleRotateAngle = 180;
    wheel.navItems[7].titleRotateAngle = 180;
    wheel.navItems[8].titleRotateAngle = 0;
    wheel.navItems[9].titleRotateAngle = 0;
  }
}
```

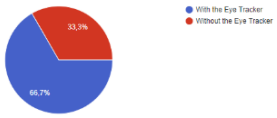
5.3 Survey answers



Circular Menu Preference

Of the two Circular Menus, which was your favourite?

3 risposte



Why?

2 risposte

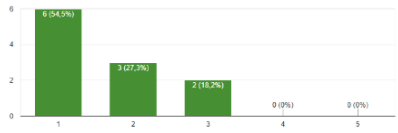
because it s faster

Too confusing

Circular Menu Preference (2)

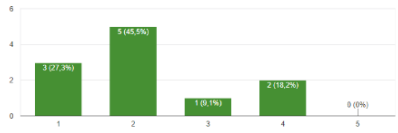
How would you rate the time efficiency of the Circular Menu? (with the Eye Tracker)

11 risposte



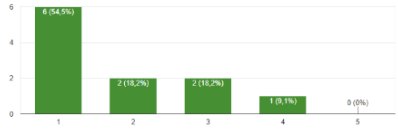
How would you rate the time efficiency of the Circular Menu? (without the Eye Tracker)

11 risposte



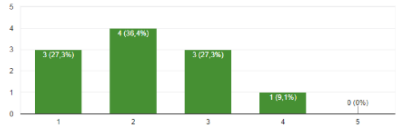
How would you rate the ease of use of the Circular Menu? (with the Eye Tracker)

11 risposte



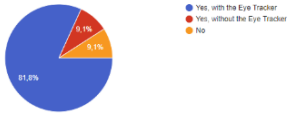
How would you rate the ease of use of the Circular Menu? (without the Eye Tracker)

11 risposte



Would you use the Circular Menu if it was made easily accessible?

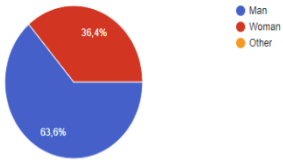
11 risposte



Statistical Data

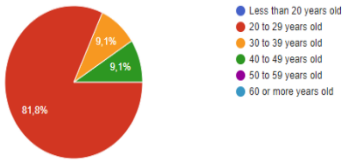
Are you a man or a woman?

11 risposte



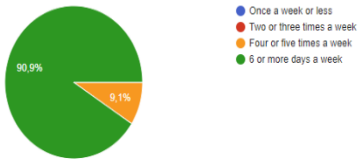
How old are you?

11 risposte



How often do you visit websites on a computer?

11 risposte



Thanks for answering!

5.4 Speed measurements

Speed test n°1		
Number	Standard	Wheel & Eye Tracker
1	3,610	2,473
2	3,260	3,020
3	4,207	3,113
4	3,982	3,171
5	2,667	2,220
6	4,300	3,399
7	2,573	2,630
8	4,078	3,554
9	3,247	3,260
10	4,319	3,823
11	3,359	2,652
12	4,071	3,287
13	3,217	3,044
Average	3,607	3,050
Diff. (sec)	0,557	