Evaluation of Graphical User Interfaces by the Search Time for Information Objects

Y Sineschuk¹, S Terekhin², I Saenko⁳, I Kotenko⁴

¹St. Petersburg University of the Ministry of Internal Affairs of the Russian Federation, 1, Letchika Piljutova street, St. Petersburg, 198206, Russia
²St. Petersburg University of State Fire Service of Emercom of Russia, Department, Organization, 149, Moskovskiy prospect, St. Petersburg, 196105, Russia
³St. Petersburg Federal Research Center of the Russian Academy of Sciences, 39, 14th Line V.O., St. Petersburg, 199178, Russia

E-mail: ibsaen@comsec.spb.ru

Abstract. The paper discusses the problem of poor quality of graphical user interfaces used for various software products. The provided analysis shows that modern software has poor-quality interfaces, and there are no flexible programs for evaluating such interfaces. The paper highlights and describes the characteristics that are used to assess the quality of interfaces, such as operator speed, error rates, skill retention, and subjective satisfaction. A formalized approach to assess the speed of searching for information and functional objects is proposed. It is based on such characteristics as the time of fixation of the gaze, movement of the eye, the volume of a person's operative memory, the zone of clear vision, the path of the user's gaze information search. An algorithm for estimating the speed of searching for information and functional elements is proposed. This algorithm can be implemented and used for further design of a software application based on it. The algorithm makes it possible to automatically evaluate both a particular interface and to compare the interfaces of different programs.

1. Introduction

Today, in the world of rapidly developing information technologies, the number of new software products is constantly growing every day, trying to cover all possible information and functional needs of ordinary users [1-4]. The constantly expanding functionality and new approaches implemented by modern algorithms inspire users with the illusion of guaranteed obtaining the necessary results in the shortest possible time and with minimal physical and psychological stress. However, as the practice of using programs and surveys of target audiences shows, the true motives and goals of users are very far from complete satisfaction [2,5,6].

The reasons for negative reviews and internal cognitive dissonance arising when working with various software products often lie in the poor quality of graphical user interfaces (GUI) [4], which are intermediaries between algorithms and mental models of users. At the same time, one of the main functions of the GUI is overcoming the conceptual barrier by users and simplifying their understanding of the mechanisms of interaction with programs.

Existing software products, both specialized and general-purpose office applications, have a number of serious drawbacks, such as the difficulty of learning to work with software tools, low
speed in the environment, and constant crashes and errors in systems. Also, the indicator of subjective user satisfaction remains at a low level [4,6]. This leads to an increase in cognitive and psychological stress on the operator and the emergence of internal cognitive resistance [4,6-8].

2. Theoretical background

To eliminate these shortcomings and improve the quality of the graphical interface, it is necessary, first of all, to be able to compare them both with each other and with a certain standard [4,9,10]. One of these author's methods is presented in this paper and is based on an estimate of the time spent by the user to find information and functional objects.

The paper discusses the concept of information retrieval from the point of view of the process of finding an object on a screen form with given information characteristics, such as: a special shape, colour, or the required functionality: indicator, button, checkbox, input field. The user's task is to find such an object, and it is characterized by the time spent on this search. The total search time for information is determined as follows:

\[ T_{is} = \sum_{i=1}^{n} \left( t_{g,i} + t_{f,i} \right) \]  

(1)

where \( t_{g,i} \) is a time of the \( i-th \) eye's movement, \( t_{f,i} \) is a time of the \( i-th \) eye's fixation, \( n \) is the number of search steps (the number of commits spent on finding the desired object).

The user's gaze movement is determined by the angle of the gaze jump, and the fixation time is a complex concept and depends on such factors as the properties of information fields, the way of operation and the degree of complexity of the characteristics of the required elements of the graphical interface [1].

In conditions of uniformity of elements of a specific information field of the GUI and a specific task, the value \( t_{f,i} \) is relatively constant and is a characteristic of these operating conditions. It can be taken from the following experimental results [1]: fixing the GUI object - 370 ms; reading letters or numbers - 310 ms; fixed characters - 300 ms; fixation of simple geometric shapes - 200 ms; fixation of the indicator - 280 ms.

In turn, \( t_{g,i} \) under these conditions is equalled to 1. Then (1) will take the following form:

\[ T_{is} = \bar{n} \cdot t_{f,i} \]  

(2)

where \( \bar{n} \) is the mathematical expectation of the number of search steps (visual fixations required to find an object with the specified features in the GUI):

\[ \bar{n} = \frac{N}{M + 1} + 1 \]  

(3)

where \( N \) is the total volume (number of elements) of the information field, \( M \) is the number of objects that have the specified search feature, \( a \) is the volume of visual perception.

Substituting \( \bar{n} \) from formula (3) into formula (2), we get

\[ T_{is} = \frac{N}{M + 1} \cdot t_{f,i} \]  

(4)

The volume of visual perception (\( a \)) has some limitations associated with the cognitive features of human nature. So, the amount of a person's operative memory is 4-6 components [6], and the size of the clear vision zone during the search is about 10° in the horizontal plane and 6°-7° in the vertical plane [11].
The above parameters and formulas were used to create an algorithm for evaluating the graphical user interface in terms of the speed of information retrieval, which consists of the following steps.

**Step 1.** Segmentation of the analyzed area of the GUI into rectangular zones with a size not exceeding the clear view zone, which is 400 pixels horizontally and 240 pixels vertically. It is assumed that the user is at a distance of 650 mm from the monitor \([6,12]\) (figure 1).

**Figure 1.** Segmentation of the analyzed area of the GUI into rectangular zones.

**Step 2.** Counting the total number of objects \(N\) separately in each selected segment (figure 2, objects are highlighted in green). If an object is located in several segments, it belongs to the segment in which its large area is located.

**Figure 2.** Counting the total number of objects \(N\) separately in each selected segment.

**Step 3.** Calculation of the number of objects for which the attribute \(M\) is set to search separately in each segment (figure 3, objects are highlighted in orange). If an object is located in several segments, it refers to the segment in which its large area is located.

**Figure 3.** Calculation of the number of objects for which the search function is set.
Step 4. Selecting $t_{ij}$ from reference materials [4] depending on the properties of the sought object.

Step 5. Selecting $a$. The minimum value of $a$ is 4, since experimentally any user can save such a number of elements in his operative memory [6,13]. This indicator can be changed in the range from 4 to 6, depending on the developed custom models.

Step 6. Calculation of the complexity of information search for each segment according to (4) and the properties of the sought object. The results obtained allow us to compare the segments with each other.

3. Results

Using the developed algorithm, we will analyze two segments of the Microsoft Office Word program for the complexity of finding information using the text formatting buttons (figure 4).

![Graphical representation of calculation results](image)

**Figure 4.** Graphical representation of calculation results.

Let us form table 1 with the values of the parameters required to calculate the complexity of information retrieval, with the following restrictions:

1. the calculations do not take into account the nesting of elements;
2. potential objects for analysis are highlighted with green rectangles;
3. parameter $a$ is set to 4;
4. since the desired object is a button, we accept an experimental value of 200mc [1].

| Table 1. Parameters for calculating the complexity of information search. |
|-----------------|-----------------|-----------------|-----------------|
|                 | Number of elements | Volume of visual perception | Time of the eye's fixation (ms) | Number of searched objects |
| **Area 1**      | 25               | 4               | 200             | 9 |
| **Area 2**      | 22               | 4               | 200             | 17 |

Substituting the values from table 1 into (4), we get the following values for the information search time: for Area 1, the search time is 145 ms, for Area 2 – 76.47 ms. Thus, the search for information on the indicated objects in the second area will be 2 times faster than in the first.

When assessing the complexity of finding information about a specific object throughout the entire area of the graphical interface, the developed algorithm is complemented by the following two steps.

Step 7. After completing Step 6, the search for the desired object is performed in one of the formed segments. The sequence in which each segment is traversed determines the F-shaped path. According to studies [14-17], the sequence of the passage of the segments can be as shown in figure 1.

Step 8. Finally, the complexity of information retrieval is determined by the sum of information retrieval for each traversed segment along the F-shaped path:
The developed algorithm is the basis for a software product that automatically calculates the complexity of information retrieval both for a group of objects and for a specific object.

The algorithm of the software product is as follows:
1) Opening the graphical interface on the computer desktop.
2) Segmentation of the graphical interface into 400x240px zones.
3) Selection of operator search objects (button, checkbox, radial button, drop-down list, input field) indicating the parameter a (in accordance with the target user model).
4) Presentation of the results of calculations according to (5) to the operator in numerical form.
5) Presentation of the calculation results to the operator in the graphical form, where each segment is assigned one of three colours (green, blue, orange) in accordance with the limitations of the volume of human visual perception. At \( M \leq 4 \), the segment is coloured green; at \( 4 < M \leq 6 \), it is blue; at \( M > 6 \), it is orange (as shown in figure 4).

\[
T_{i,j} = \left( \frac{N_i + 1}{M_i + 1} \cdot t_{r,i,j} \right) + \left( \frac{N_j + 1}{M_j + 1} \cdot t_{r,j,i} \right) + \ldots + \left( \frac{N_j + 1}{M_j + 1} \cdot t_{r,j,i} \right),
\]

where \( J \) is the total number of traversed segment.

4. Discussion
The developed algorithm can be implemented in a software product in two ways. In the first way, we can use the capabilities of machine vision. In the second way, we can access the elements of the program through the code.

The implementation in the first way seems to be simpler, since there are currently out-of-the-box object recognition libraries such as OpenCV. However, there is a problem with the difficulty of recognizing the shape of objects, since modern interfaces no longer use clearly marked buttons. More often, there are symbols or icons whose borders appear only after you place the cursor over them. For this reason, the second way will be more practical and accurate due to direct access to information and functional objects of the GUI.

However, currently, among the services with similar functionality, only services using heat map algorithms are available. These programs are used to evaluate websites [8,11,18]. They have limited capabilities to find problem areas of the GUI [19,20]. They do not take custom models into account, as in the proposed algorithm.

It should also be noted that developing custom models is an integral part of modern interaction design.

The proposed algorithm allows you to change the input data depending on the characteristics of the user, such as the amount of the operative memory, the size of the area of clear vision, as well as the sequence of gaze analysis.

5. Conclusions
The paper proposes an approach to assessing the GUI based on calculating the search time for interface elements and taking into account various user parameters. The developed algorithm for calculating the search time allows us to compare various options for constructing a GUI in terms of the complexity of finding information and make assumptions about the quality of various aspects of the interface: the complexity of visual perception, subjective satisfaction, the complexity of mastering the skills of working with the software, the speed of the program.

The proposed algorithm is planned to be implemented in software products through direct access to information and functional objects, since this approach has a higher accuracy than using the capabilities of machine vision. The development and research of such software products is considered as a further research direction.
6. References

[1] Jura J, Cejnek M, Trnka P and Cahyna M 2021 Engineering Psychology as a Humanities for Engineers Proc. Int. Conf. on Process Control (Strbske Pleso) (New York: IEEE) pp 119–24
[2] Wills C E 1994 User Interface Design for the Engineer Proc. ELECTRO '94 (Boston) (New York: IEEE) p 415
[3] Nielsen J 1999 Designing Web Usability: The Practice of Simplicity (Indianapolis: New Riders Publishing)
[4] Bai Y-W, Lin H-E and Jau W-Ch 2008 Design and Implementation of an Integrated Man-Machine Interface by Touch Panel for an Embedded Electronic Measurement System Proc. Conf. virtual Environments, Human-Computer Interfaces and Measurement Systems (Istanbul) (New ork: IEEE) pp 31–6
[5] Raskin J 2000 The humane Interface: New Directions for Designing Interactive Systems (New York: ACM Press/Addison-Wesley Publishing)
[6] Goldenberg P, Mark J, Harvey B, Cuoco A and Fries M 2020 Design Principles behind Beauty and Joy of Computing Proc. Tech. Symp. Computer Science Education (New York: ACM) pp 220–6
[7] Kong X, Ma K, Hou Sh, Wang D and Xia F 2019 Human Interactive Behavior: A Bibliographic Review IEEE Access 7 4611–28
[8] Dubey S K, Ghosh S and Rana A 2012 Comparison of Software Quality Models: An Analytical Approach Int. J. of Emerging Technology and Advanced Engineering 2 111–9
[9] Rubinov N S, Rudakov I V and Stroganov I V 2020 Development of Specialized Software for Biomedical Research Proc. Conf. Russian Young Researchers in Electrical and Electronic Engineering (Moscow: IEEE) pp 486–8
[10] Dobslaw F, Feldt R, Michaëlsson D, Haar P, Neto F G de O and Torkar R 2019 Estimating Return on Investment for GUI Test Automation Frameworks Proc. Int. Symp. Software Reliability Engineering (Berlin) (New York: IEEE) pp 271–82
[11] Levin I P, Schneider S L and Gaeth G J 1998 All Frames Are Not Created Equal: A Typology and Critical Analysis of Framing Effects Organizational Behavior and Human Decision Processes 76 2 149–88
[12] Talabi U O, Amunga B and Rad B B 2016 Usability and Security in User Interface Design: A Systematic Literature Review Int. J. of Inf. Technology and Comp. Science 8 5 72–80
[13] Gonzalez R C and Woods R E 2018 Digital Image Processing (Delhi: Pearson India)
[14] Lo E C and Marchand M 2004 Security audit: a case study [information systems] Proc. Canadian Conf. Electrical and Computer Engineering (Niagara Falls) vol 1 (New York: IEEE) pp 193–6
[15] Cooper A, Reiman R, Cronin D and Noessel Ch 2014 About Face: The Essentials of Interaction Design (New York: Wiley)
[16] Kay J 2009 A Test-First View of Usability Interacting with Computers 21 5–6 347–9
[17] Shneiderman B, Plaisant C, Cohen M, Jacobs S, Elmqvist N and Diakopoulos N 2021 Designing the User Interface: Strategies for Effective Human-Computer Interaction (New York: Pearson)
[18] Viola P and Jones M 2011 Rapid Object Detection Using a Boosted Cascade of Simple Features Proc. Conf. Computer Vision and Pattern Recognition (Kauaii) vol 1 (New York: IEEE) pp 511–8
[19] Blackledge J M 2005 Digital image processing: Mathematical and Computational Methods (Cambridge: Woodhead Publishing)
[20] Bernet Y and McCarthy J M 1987 Report: The 630 Multitasking Graphics Terminal AT&T Tech. J. 66 6 3–14

Acknowledgments

This research is being supported by the grant of RSF No. 21-71-20078 in SPC RAS.