DOGeye: Controlling your Home with Eye Interaction

DOGeye: Controlling your Home with Eye Interaction

Dario Bonino[a], Emiliano Castellina[a], Fulvio Corno[a], Luigi De Russis[a,*]

[a]*Politecnico di Torino, Dipartimento di Automatica ed Informatica, Corso Duca degli Abruzzi 24, 10129 - Torino, Italy*

Abstract

Nowadays home automation, with its increased availability, reliability and with its ever reducing costs is gaining momentum and is starting to become a viable solution for enabling people with disabilities to autonomously interact with their homes and to better communicate with other people. However, especially for people with severe mobility impairments, there is still a lack of tools and interfaces for effective control and interaction with home automation systems, and general-purpose solutions are seldom applicable due to the complexity, asynchronicity, time dependent behavior, and safety concerns typical of the home environment. This paper focuses on user-environment interfaces based on the eye tracking technology, which often is the only viable interaction modality for users as such. We propose an eye-based interface tackling the specific requirements of smart environments, already outlined in a public Recommendation issued by the COGAIN European Network of Excellence. The proposed interface has been implemented as a software prototype based on the ETU universal driver, thus being potentially able to run on a variety of eye trackers, and it is compatible with a wide set of smart home technologies, handled by the Domotic OSGi Gateway. A first interface evaluation, with user testing sessions, has been carried and results show that the interface is quite effective and usable without discomfort by people with almost regular eye movement control.

Keywords: Human-Home Interaction, Smart Homes, Domotics, Usability, User Interface, User Study

1. Introduction

In the last 5 years, (smart) home automation gained a new momentum, thanks to an increased availability of commercial solutions (e.g., X10 or Z-Wave) and to steadly reducing costs. The evergreen appeal of automated, intelligent homes together with a raising technology maturity has fostered new research challenges and opportunities in

*Corresponding author, Tel.: +39-011-5647191, Fax.: +39-011-5647099

Email addresses: dario.bonino@polito.it (Dario Bonino), emiliano.castellina@polito.it (Emiliano Castellina), fulvio.corno@polito.it (Fulvio Corno), luigi.derussis@polito.it (Luigi De Russis)

the field of "intelligent" or "smart" environments. According to the Mark Weiser definition, a Smart Home system, that in this paper we decline as domotic or environmental control system[1], is "a physical world that is richly and invisibly interwoven with sensors, actuators, displays and computational elements, embedded seamlessly in the everyday object of our lives, and connected through a continuous network" [1], providing ways for controlling, interacting and monitoring the house. The idea behind this vision is that homes of tomorrow would be smart enough to control themselves, understand contexts in which they operate and perform suitable actions under inhabitants' supervision [2]. Although smart and autonomous homes might raise controversial opinions on how smart are they or should they be, currently available commercial solutions can start playing a relevant role as enabling technology for improving the care of the elderly [3, 4] and of people with disabilities [5, 6], reducing their daily workload in the house, and enabling them to live more autonomously and with a better quality of life. Even if such systems are far from cutting-edge research solutions, they are still really complex to master since they handle and coordinate several devices and appliances with different functionalities and with different control granularities.

In particular, among other disabilities, people who have severely impaired motor abilities can take great advantages from eye tracking systems to control their homes, since they generally retain normal control of their eyes, that become therefore their preferential stream of interaction [7]. Eye tracking can transforms such a limited ability into both a communication channel and an interaction medium, opening possibilities for computer-based communication and control solutions [8]. Even if eye tracking is often used for registering eye movements in usability studies, it can be successfully exploited as alternative input modality to control user interfaces. Home automation can then bridge the gap between software and tangible objects, enabling people with motor disabilities to effectively and physically engage with their surroundings [9]. Several house control interfaces have been proposed in the literature, i.e., applications to allows users to control different types of devices in their homes, to handle triggered alarms, etc. Such interfaces, either based on conventional unimodal [10] or multimodal interactions [11] (e.g., mouse, remote controller, etc.), are too often uncomfortable and/or useless for people with severe impaired motor abilities, and only few of them have been specifically designed and developed to be controlled with eye movements.

In 2004, applications based on gaze interaction have been analyzed by a European Network of Excellence, named COGAIN (*Communication by Gaze Interaction*)[2], to evaluate the state-of-the-art and to identify potential weaknesses and future developments. According to the report "D2.4 A survey of Existing 'de-facto' Standards and Systems of Environmental Control" [12], the COGAIN Network identified different problems in eye-based house control applications, such as the lack of advanced functionalities for controlling some appliances of the house, the absence of interoperability between different smart house systems or the difficulty to use an eye tracker for realizing some actions. In a subsequent report [13], COGAIN members proposed solutions

[1] We only consider systems currently available on the market such as X10, Konnex, MyHome, Z-Wave and ZigBee HA.

[2] `http://www.cogain.org`

to overcome the discovered problems. In particular, they proposed 21 guidelines to promote safety and accessibility in eye tracking based environmental control applications.

This paper describes the design and development of DOGeye, one of the first home control applications designed for gaze-based interaction and by explicitly accounting for the COGAIN guidelines. DOGEye is a multimodal eye-based application for home management and control, based on state-of-the-art technologies in both tracking and home control. It enables people to control their domotic homes through different input devices, possibly combined, so that it does not limit itself to eye tracking only. The presence of various input modalities allows application use by other people present in the house and offers different alternatives to the persons affected by possibly evolving impairments such as the ALS (Amyothrophic Lateral Sclerosis).

The remainder of the paper is organized as follows: Section 2 presents the basic features of eye tracking technology and the characteristics of eye-based user interfaces while section 3 presents the work accomplished by the members of the COGAIN Network and describes COGAIN guidelines for eye tracking based environmental control applications. Section 4 reports relevant related works and findings. DOGeye design and architecture is described in Section 5, while Sections 6 and 7 report the set-up and results of a user test involving people in a controlled environment, thus building the basis for further considerations and research. Section 8 concludes the paper and outlines future works.

2. Eye Tracking Basics

To better understand the principles and implementation of eye controlled interface, this section defines some terms and features pertaining to eye movements and eye tracking.

The eye does not generally move smoothly over the visual field; instead, it makes a series of quick jumps, called *saccades*, along with other specialized movements [14]. A saccade lasts 30 to 120 ms, and typically covers 15 to 20 degrees of visual angle [15]. Between saccades, the *gazepoint*, i.e., the point in a scene where a person is looking, stays at the same location (with a slighty tremor) for a *fixation* that lasts from 100 to 400 ms; a longer fixation is called *dwell* [7].

Eye positions and their movement relative to the head can be measured by using different methods, e.g., computer vision techniques. One of these techniques is the so-called *Corneal Reflection* technique that consists in sending a small infrared beam toward the center of the pupil and estimating the changes in its reflexion (*eye tracking*). Eye tracking has several distinguishing features [15]:

- it is *faster* than other input media, as Ware and Mikaelian [16] observed; before the user operates any mechanical pointing device, she usually looks at the destination to which she wishes to move;
- it is *easy* to operate, since no training or particular coordination is required to look at an object;

- it shows where the *focus of attention* of the user is located; an eye tracker input could be interpreted as an indication of what the user points at, but it can also be interpreted as an indication of what the user is currently paying attention to, without any explicit input action on her part;
- it suffers from *Midas Touch* problem: the user expects to be able to look at an item without having the look cause an action to occur. This problem is overcome by using techniques such as dwell time or blink selection;
- it is *always on*; in fact, there is no natural way to indicate when to engage the input device, as there is with grasping or releasing the mouse;
- it is *noninvasive*, since the observed point is found without physical contact;
- it reduces *fatigue*; if the user uses an eye tracker input instead of other manual pointing devices, movements of arms and hands will be reduced and will cause less fatigue;
- it is *less accurate* than other pointing devices, such as a mouse.

Because of these features, an eye tracking based interface has some specific peculiarities: for example, graphical widgets and objects are bigger than in traditional user interfaces, due to eye tracking lower accuracy; and the pointer is often absent, since its presence could divert users' attention [17], replaced by other forms of visual feedback.

To overcome the "Midas Touch" problem, many interfaces use the *dwell time* technique. By using such a technique, the user can select a widget of a user interface only if she continues to look at it for a sufficiently long time. The amount of time is, generally, customizable by the user itself.

Moreover, interaction with eye-based interfaces can be improved by exploiting the *Selection-Action strategy* (SA), already used in the iAble application[3] and whose basic principle was proposed by Razzak at Al. [18]. This strategy permits to separate the selection of an object from the activation of its associated actions. The *selection* is the process of choosing an object and displaying its related options, while *action* permits to perform some task on the selected object. The selection-action strategy is generally implemented by showing two separate areas to interact with: one is used only for selection, with a really short dwell time; the other is used for actions, with a longer dwell time, controllable by users. Two interaction patterns lie at the basis of SA: the *non-command based interaction*, used for selection, and the *command based interaction*, used for actions. In the *non-command based interaction* pattern, the computer observes and interprets user actions instead of waiting for explicit commands. By using this pattern, interactions become more natural and easier to use, as indicated by the work of Tanriverdi and Jacob [19]. In *command based interactions*, instead, the user explicitly directs the computer to perform some operations.

[3] a SRLabs commercial software - `http://www.srlabs.it/en/iable.html`

3. The COGAIN guidelines

The COGAIN (*Communication by Gaze Interaction*) project was launched in September 2004, as a Network of Excellence supported by the European Commission's Information Society Technology under the 6th framework programme, with the goal of "integrating cutting-edge expertise on gaze-based interface technologies for the benefit of users with disabilities." The project gathered over 100 researchers belonging to the world's cutting-edge research groups and companies with leading expertise in eye tracking integration with computers and in assistive technologies for people with motor impairments. COGAIN also involved the advice of people coming from hospitals and hospices, working daily with persons with motor impairments. Thanks to the integration of research activities, the network developed new technologies and systems, improved existing gaze-based interaction techniques, and facilitated the implementation of systems for everyday communication (for more information see the COGAIN web site[4]).

COGAIN considered home automation as an opportunity for eye tracking users to live in an autonomous way. For this reason, in 2007, the COGAIN project published a Draft Recommendations for Gaze Based Environmental Control [13]. This document proposes a set of guidelines for developing domotic control interfaces based on eye interaction. The guidelines originated from a set of realistic use case examples, describing typical actions that a user with impairments can do in her domotic environment, and underwent an evaluation and validation process in the COGAIN project.

Gaze Based Environmental Control guidelines (see Table 1) are grouped in 4 main categories:

1. *Control applications safety*: guidelines concerning the behavior of the application in critical conditions, such as alarms and emergencies.
2. *Input methods for control application*: guidelines about input methods that the control applications should support.
3. *Control applications significant features*: guidelines impacting the management of commands and events within the house.
4. *Control applications usability*: guidelines concerning the graphical user interface and the interaction patterns of the control applications.

Each guideline is associated to a priority level (PL), following the typical W3C style:

- Priority Level 1: the guideline MUST be implemented by the applications, since it relates to safety and basic features;
- Priority Level 2: the guideline SHOULD be implemented by the applications.

Control interfaces for domotic environments must face three main issues that lie at the basis of most guidelines:

[4] `http://www.cogain.org/`

Guideline	Content	PL
1.1	Provide a fast, easy to understand and multi-modal alarm notification	1
1.2	Provide the user only few clear options to handle alarm events	2
1.3	Provide a default safety action to overcome an alarm event	1
1.4	Provide a confirmation request for critical & possibly dangerous operations	1
1.5	Provide a STOP Functionality that interrupts any operation	1
2.1	Provide a connection with the COGAIN ETU-Driver	1
2.2	Support several input methods	2
2.3	Provide re-configurable layouts	2
2.4	Support more input methods at the same time	2
2.5	Manage the loss of input control by providing automated default actions	2
3.1	Respond to environment control events and commands at the right time.	1
3.2	Manage events with different time critical priority	1
3.3	Execute commands with different priority	1
3.4	Provide feedback when automated operations and commands are executing	2
3.5	Manage Scenarios	2
3.6	Communicate the current status of any device and appliance	2
4.1	Provide a clear visualization of what is happening in the house	1
4.2	Provide a graceful and intelligible interface	2
4.3	Provide a visualization of status and location of the house devices	2
4.4	Use colors, icons and text to highlight a change of status.	2
4.5	Provide an easy-to-learn selection method.	2

Table 1: COGAIN Guidelines summary [13]

Asynchronous control sources The control interface is not the sole source of commands: other house occupants may choose to operate on wall-mounted switches, some external events may change the status of some sensors, etc. The control interface needs therefore to continuously update the status of the house, i.e., the icons, menus and labels must timely change according to the home status evolution, to provide a coherent view of the environment *(Guidelines 3.6, 4.1)*.

Time-sensitive behavior In an alarm condition the user is normally put in a stressful condition: she has limited time to take important decisions, which may pose threats to her safety. In such stressful conditions, eye control may become unreliable or, in some cases, not functional. In this case the control interface must offer simple and clear options, easy to select, and must be able to take the safest action in case the user cannot answer in time *(Guidelines 1.1, 1.2, 1.3, 1.4, 3.2, 3.3)*. Time-sensitive behaviors include automated actions, initiated by rules (e.g., closing the windows when it is raining). In this case the user should be allowed to interrupt any automatic action or to override it at any time. The control interface should make the user aware that an automatic action has been initiated, and offer ways to interrupt it *(Guidelines 1.5, 3.5)*.

Structural and functional views Control interfaces can organize the home information according to two main logics: structural and functional. Most environmental control applications apply the structural logic and display information mimick-

ing the physical organization of devices in the home. This choice, however, cannot address global or "not-localized" actions as switching the anti-theft system on, or set the temperature of the house. Functional logic, instead, is best suited for tackling not-localized options and to support type-driven interaction with interface elements, i.e., interaction involving actions having the same nature. Effective interfaces should find a good trade-off between the two logics *(Guidelines 4.1, 4.2)*.

4. Related Works

While home automation technology is maturing and evolving, a sensible lack of User Interfaces for controlling such environmental systems using eye movements can be easily spotted. Overall, different studies and surveys about HCI perspective on automated homes are present in literature, such as the work of Hee-Cheol Kim et Al. [20]. In the same way, a variety of User Interfaces are proposed for controlling a smart environment using different traditional input modalities, such as mouse, keyboard and remote control [21, 22].

The *Home Operating System* (Home OS) [11] (Figure 1) is a multimodal interface proposed by the Technical University of Berlin. This UI allows the user to control her home using touch, speech and gesture interactions. However, according to COGAIN Guidelines (Table 2) this interface is not yet complete and eye tracking interaction was not considered.

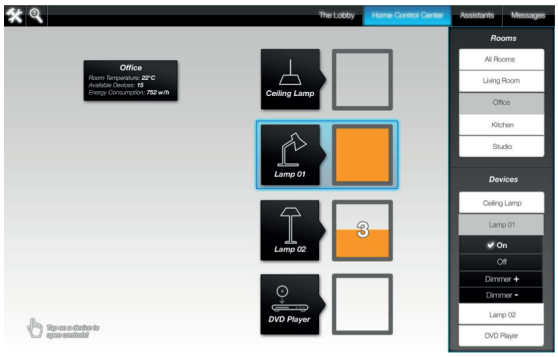


Figure 1: The Home Operating System

Eye tracking systems, typically, include commercial interfaces for environmental control, obviously based on gaze interaction. *LC Technologies* provides a basic eye-controlled "Light and Appliances" interface (Figure 2), bundled with some electrical switching equipments. This system provides basic control of light and appliances located anywhere in the home. The user can turn appliances on and off by looking at a bank of switches displayed on the screen, with commands sent to home sockets and lights via the home mains electricity wiring, exploiting the X10 protocol.

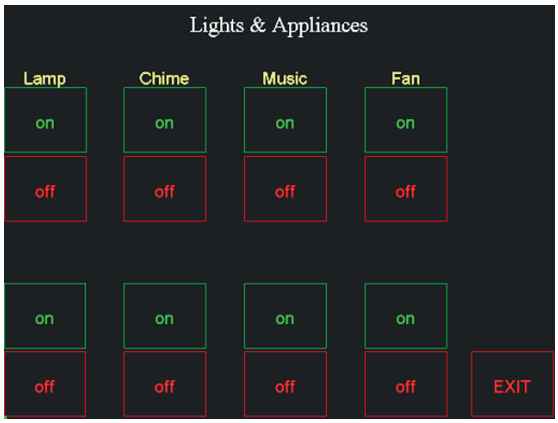


Figure 2: LC Technologies domotic interface

The same basic functionalities are offered by ERICA *EnviroMate*[5] (Figure 3) and iAble *DOMOTICS*[6] (Figure 4). ERICA presents a grid of buttons related to environmental objects and can use X10, Z-Wave and general purpose IR for interface-to-device communications. The iAble system takes a pretty similar approach: it displays a grid of buttons and communicates with home devices through an infrared transmitter, supplied with the system. Both interfaces are quite incomplete in their functionalities.

Figure 3: ERICA domotic interface

We evaluated HomeOS, ERICA EnviroMate and iAble DOMOTICS against the COGAIN guidelines by both analyzing nominal features and runtime execution of the 3 applications. Every guideline category has been considered separately and, for each category, we measured the number of satisfied guidelines versus the total number of

[5] http://www.dynavoxtech.com/default.aspx
[6] http://www.srlabs.it/en/iable-modules.html

Figure 4: iAble domotic interface

category guidelines. If an application does not respect any guideline in a category, its COGAIN support level is labeled as "Absent" for the category; if part of the guidelines (but not all) are respected, for a given category, the application is labeled as having "Partial" support for the COGAIN guidelines in the considered category. Finally, applications respecting all guidelines in a given category are labeled as providing "Full" compliance. Table 2 summarizes the evaluation results showing that none of the considered alternatives have full support to the COGAIN guidelines and, moreover, that partial support is actually very limited, being less than 50% of the guidelines on average.

Guidelines category	**Home OS**	**EnviroMate**	**iAble DOMOTICS**
1. Safety	Absent (0/5)	Absent (0/5)	Absent (0/5)
2. Input methods	Partial (3/5)	Absent (0/5)	Absent (0/5)
3. Operative features	Partial (3/6)	Partial (2/6)	Partial (2/6)
4. Usability	Partial (2/5)	Absent (0/5)	Absent (0/5)

Table 2: An evaluation of the interfaces presented in this Section. The numbers in parenthesis represent the quantity of fulfilled guidelines per category.

5. DOGeye

To overcome the shortcomings of currently available solutions and to provide a first reference home control application designed explicitly for supporting COGAIN guidelines through multi-modal interaction, with a strong focus on eye tracking technologies, we designed, implemented and evaluated DOGEye. Following subsections

describe in detail the logic architecture of gaze-based home interaction supported by DOGEye and provide useful insights on the DOGEye design and functionalities.

5.1. Logic Architecture

DOGeye has been designed according to *user-centered design* [23] to be a COGAIN-compliant, multimodal eye-based application for controlling, interacting and monitoring a house. Interaction with automated (smart) homes is provided by *Dog* (Domotic OSGi Gateway) [24], an ontology-based domotic gateway able to integrate and abstract functionalities of heterogeneous domotic systems, thus offering a uniform, high-level access to home technologies.

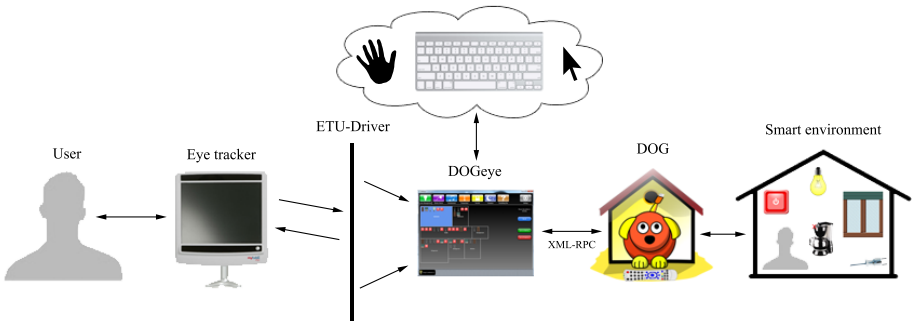


Figure 5: The context where DOGeye is inserted

Figure 5 shows the logic architecture of home control through gaze. DOGeye communicates with the smart environment exploiting Dog (on the right), thanks to a XML-RPC connection that allows exchanging all needed information about the home and directly controlling the available devices. The DOGeye connection with Dog allows to respond almost instantly (≈100ms) to environmental control events and commands, thus complying to the COGAIN Guideline 3.1.

DOGeye, can either be controlled by gaze (main interaction channel, focus of this paper), by communicating with the eye tracker through a universal driver named ETU-Driver [25] (Guideline 2.1), or can be used exploiting other interaction mediums such as touch screens or traditional keyboards and mouses (not reported in the paper), thus fulfilling Guideline 2.2: *Support several inputs methods*. These input methods, moreover, are usable at the same time (according to *Guideline 2.4*), allowing the user to manage a possible loss of eye tracking input by using other interaction methods and giving a preliminary implementation to Guideline 2.5, i.e., manage the loss of input control by providing automated default actions.

5.2. Design

The DOGeye interface has been designed following an incremental specification paradigm where a first layout skeleton (see Figure 6) has been incrementally refined to explicitly comply with most of COGAIN guidelines about environment control.

5.2.1. Interface skeleton

According to the COGAIN guideline 2.3, the draft specification accounts for reconfigurable layouts appropriate for different eye tracking resolutions and precisions, even if the current DOGeye implementation is not yet complete in this regard. As suggested by guideline 4.2 on "graceful and intelligible interfaces", we adopted a color policy shared by all the interface elements, in a consistent manner, as shown in Table 3.

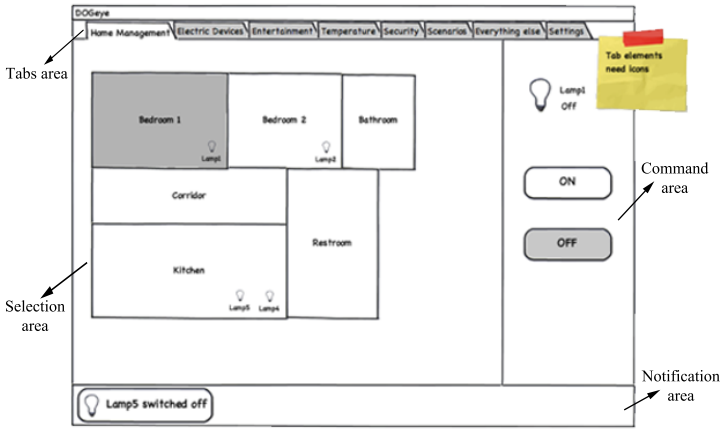


Figure 6: DOGeye abstract layout

Function	Description	Color
Positive behavior	Commands with a positive meaning, such as "open"	Green
Negative behavior	Commands with a negative meaning, such as "close" or "stop"	Red
Neutral behaviour	Commands with neither a negative or positive meaning, such as "set to..."	Gray
Selection	Buttons for enabling/disabling selection functionalities	Black
House navigation	Commands for navigating within the house, such as "enter this room"	Blue

Table 3: Descriptions of the colors used for DOGeye buttons

The interface visualization is divided in four main logical areas:

- **Tabbed area** - in the upper part of the interface, it represents the *functional view* of the house. It contains tabs showing different views of the ambient, according to the type of the device to show. This area behaves according to the "selection" part of the Selection-Action pattern.
- **Selection area** - in the left part of the interface, this large area represents the *structural view* of the house. It contains the house rooms and its devices and it allows selecting a room, device or a group thereof. This area behaves according to the "selection" part of the Selection-Action pattern.

- **Command area** - in the right part of the interface, it shows the commands supported by the object selected in the selection area. This area behaves according to the "action" part of the Selection-Action pattern.
- **Notification area** - located at the bottom of the interface, it shows notifications and alarms to the user.

The first two areas are designed for specifically addressing COGAIN requirements asking for a proper balance between functional and structural home views. In the functional view, devices are grouped according to their nature and functionality, rather than by location. The structural view, instead, follows the physical organization of the devices in the house, i.e., each device is located inside its containing room. The joint adoption of the two views defines two distinct navigation hierarchies: a functional hierarchy that lets users choose the device type or the kind of operation to accomplish, and a structural hierarchy, which allows users to choose specific devices inside the home.

The *Command area* fulfills Guideline 1.4: *Provide a confirmation request for critical and possibly dangerous operations*. DOGeye never acts on the basis of just one fixation, but always requires at least two: the first with a short dwell time and the second with a longer one. In fact, by using a short dwell time the user can easily select an object, but errors are possible: no harms, since the selection has been quick and the operation is simply reversible.

The last area - the *Notification area* - provides a feedback each time an operation or a command is executed in the house. It communicates also the current state of any device and appliance, thus implementing Guidelines 3.4, 3.6 and 4.1. Every status change of the house devices is notified in different ways, both visually and phonetically. For example, when the lamp in the kitchen is switched on, its icon will represent a lighted bulb, a notification with the lamp image and carrying the label "*The lamp in the kitchen is switched on*" will show up in the *Notification area* and the speech system, built in the Windows OS, speaks the same sentence reported in the notification. Moreover, the speech system gives the user a feedback for every actions she does, e.g., it speaks "You are in the kitchen" when the user "enters" the kitchen using the application. In this way, DOGeye actuates the Guideline 4.4: *Use colors, icons and text to highlight a change of status*.

5.2.2. Actual DOGeye Interface

The final appearance of DOGeye is presented in Figure 7; it is possible to notice eight tabs in the *Tabbed area*, each with a different function. Every tab has a different icon with a different color and an explanatory text: associating an icon with a text label significantly reduces the possibility of misinterpretation and error by a user, compared with the use of only an icon or just a short text [26].

The eight tabs with their functions are:

- **Home Management** contains the basic devices present in a house, i.e., devices belonging to the mains electricity wiring, such as shutters, doors, lamps, etc.
- **Electric Device** contains the electrical appliances not belonging to the entertainment system, such as a coffee maker.

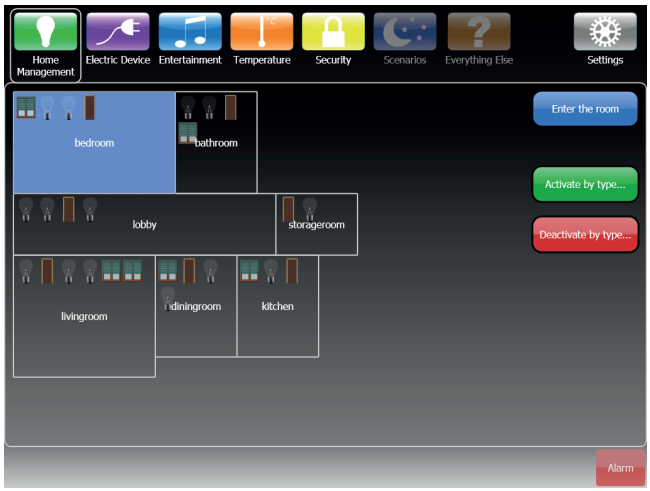


Figure 7: DOGeye User Interface

- **Entertainment** contains the devices for entertainment, such as media centers and TVs.
- **Temperature** allows handling the heating and cooling system of the house.
- **Security** contains the alarm systems, anti-theft systems, etc.
- **Scenarios** handles the set of activities and rules defined for groups of devices, as suggested by Guideline 3.5.
- **Everything Else** contains devices not directly falling in the previous tabs, e.g., a weather station.
- **Settings** shows controls for starting, stopping and configuring the ETU-Driver.

All the tabs report the home plan (i.e., the map) of the house and the current state of devices, represented as a changing icon located in the room in which the device is positioned. This architectural view of home devices enables DOGeye to satisfy COGAIN Guidelines 4.1 and 4.3 regarding visualization of what happens in the house and where. Given a tab selection, e.g., "Home Management", users can gather an overview of the current state of the house devices and may decide to turn on/off some of them. To actuate a specific device, users are required to "enter" the room containing the device and to command it through the Selection-Action interaction pattern.

Sample scenario. Imagine that Sam, a man with severe mobility impairments wants to turn on the ceiling lamp in the kitchen. In the typical DOGeye interaction, he first choses the "Home Management" tab by briefly looking at it (first level of selection), then he looks for a moment at the kitchen on the map (second level of selection), and then he fixates at the "Enter the room" button for a longer dwell time (action). After entering the room, Sam can see the list of all devices present in that room; he briefly

looks at the ceiling lamp to select it. Then he gazes at the "On" button present in the Command area, to turn the lamp on. If, afterwords, Sam wants to prepare a warm coffee, he needs to switch to the "Electric Device" tab, using the same procedure.

Tabs are designed to act as isolated points of access to the home. This means that selections made in each tab are independent from each other and that the state of each tab representation is independent from all the others. With reference to Sam's case, if Sam returns to the "Home Management" tab after switching on the coffee maker he will find the kitchen still selected and DOGeye will still be showing the devices in the kitchen and not the plan of the house, since he didn't "leave" the kitchen before changing tab.

This feature is called *tab isolation*: each tab is independent from the others, so that a selection or an action made in one of them is always preserved. Next paragraphs better detail the most relevant tab functionalities.

5.2.3. Home Management and Electric Devices

These two tabs include the most common devices present in a house, e.g., plugs, lamps, door actuators, window and shutter actuators, etc. These devices are controllable individually or in group. According to Guideline 4.5, i.e., *Provide an easy-to-learn selection method*, we implemented *single and multiple selection*, as summarized in Table 4.

Selection modality		**Activated**		
		where	*by looking at...*	*what happens?*
Single	normal	house level	a room	the chosen room is selected
		room level	a device	the chosen device is selected
	implicit	house level	a room with only one device	the only device present inside the room is selected
Multiple	normal	room level	"Multiple selection" button	it is possible to select more than one device
			"Select by type..." button	all the devices of a chosen type are selected
	implicit	house level	"Multiple selection" button	it is possible to implicitly select the devices present in more than one room
			"Activate/Deactivate by type..." buttons	all the devices of a chosen type, present in a room, are activated/deactivated

Table 4: Summary of the different selection modalities present in DOGeye

Single selection is divided in *normal* and *implicit* selection. The simplest modality is *normal single selection*: by looking at an icon, the user selects the correspondent object. This selection acts both for selecting a room from the house map and for selecting a device or an appliance once inside a room.

The *implicit single selection* occurs at the "house map level", when a selected room has only one device that is automatically (i.e., implicitly) selected by selecting the

containing room: in this case, the Command area shows directly the commands for that device, as seen in Figure 8.

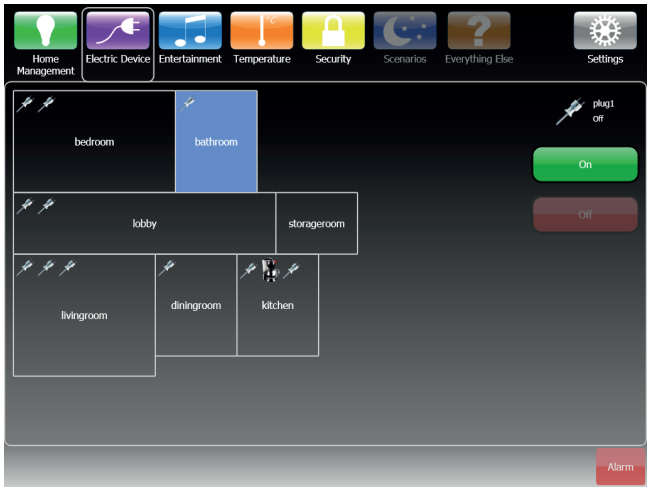


Figure 8: What happens when a room contains only one device

Multiple selection is also divided in *normal* and *implicit*. The *normal multiple selection* involves multiple devices present in a room. As shown in Figure 9, by looking at the multiple selection button, it is possible to select a subset of devices of the same type and then control them using one of the associated commands.

The *implicit multiple selection* occurs at the "house map level" and it is realized with two buttons, located in the Command area when a room is selected: "Activate by type..." and "Deactivate by type...". When the user looks at one of these buttons, a popup window appears (Figure 10) and it is possible to give a basic command to all the devices of the chosen type in that room, without "entering" it. For example, by looking at "Activate by type..." and then selecting Dimmer Lamp, it is possible to turn on all the dimmer lamps present in the selected room, but not to set their luminosity.

As a subcase of multiple selection we provided the interface of a "select all" functionality. In these tabs, by looking at the "Select by type..." button present inside the room, it is possible to select all the devices of the type chosen through a popup window, and then control them using one of the associated commands.

5.2.4. Temperature

The "Temperature" tab allows to control the temperature of a room driving the heaters/coolers present in that room. When a room is selected, it is possible to turn on/off the heating/cooling system inside the room and to set the room temperature in Celsius degrees.

This tab only implements implicit *single* and *multiple selection*, due to the presence of only one device for each room. In this way, we allow to set a uniform temperature on different ambients. An example of the implicit multiple selection is shown in Figure 11.

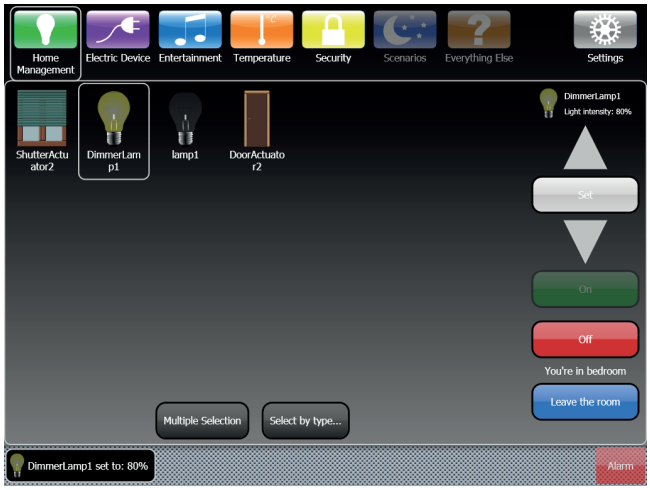


Figure 9: Example of devices inside a room in the Home Management tab

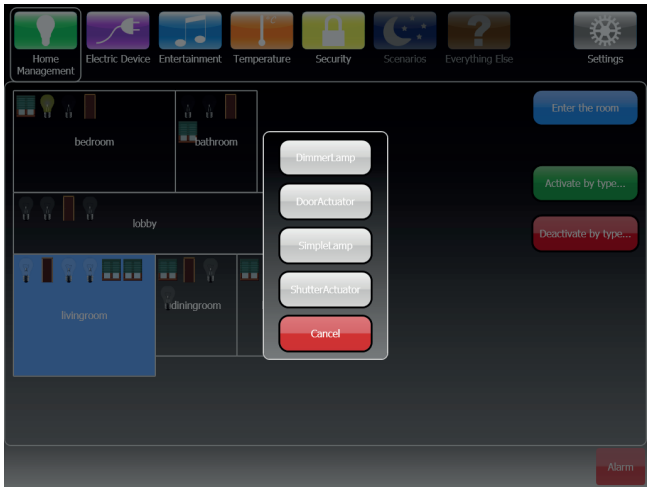


Figure 10: Example of implicit multiple selection in the Home Management tab

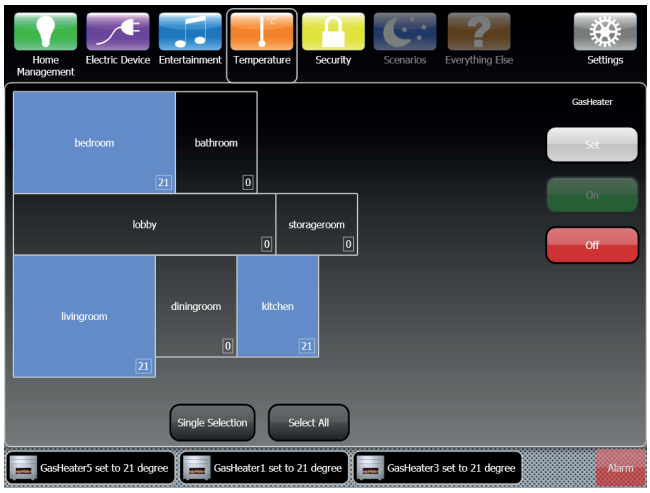


Figure 11: Example of multiple selection in the Temperature tab

In this tab, the “select all” functionality occurs at the “house map level” and it is realized with the “Select all” button: by looking at it, it is possible to select the whole house and so, implicitly, the whole heating/cooling system.

5.2.5. Security

The security tab allows to see what happens in any room provided with one or more cameras. Live videos or pictures can be viewed by entering the room containing the camera; for example, Figure 12 shows a room with one camera whose video can be accessed by looking at the upper part of the Command area.

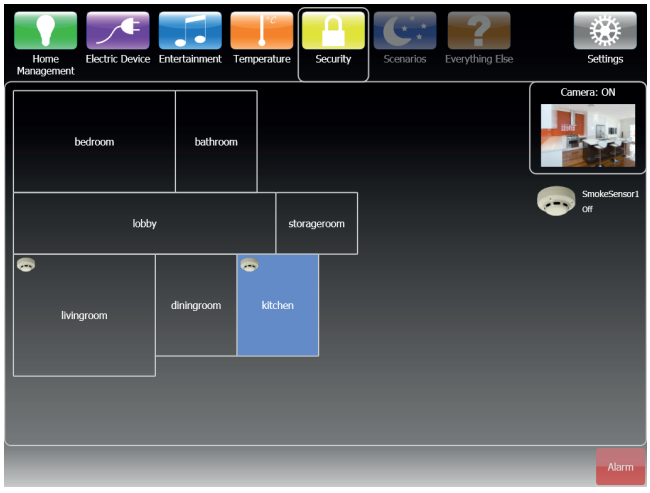


Figure 12: The Security tab

The user may expand the video to full screen by simply looking at its icon. A button at the bottom (Figure 13) closes the video and returns to the previous view.

Figure 13: Full screen video in the Security tab

5.2.6. Asynchronous Alarm Events

To support handling of alarm events, as required by the COGAIN Guidelines 1.1 (*Provide a fast, easy to understand and multi-modal alarm notification*) and 1.2 (*Provide the user only few clear options to handle alarm events*) we designed two alarm types: a *general alarm* and an *environmental alarm*. The *general alarm* functionality consists of a button, placed in the bottom right corner of the interface, that the user may use to draw attention and request help. When activated, this alarm generates an acoustic alert to attract attention, until the user stops it, by looking at a dedicated "STOP" button, thus complying with Guideline 1.5 that suggests such functionality.

An *environmental alarm* is an event triggered if an alarm notification is received from Dog; it may occur in an asynchronous way respect to user actions. On the event activation, an acoustic alert is played with the purpose of attracting user attention.

As reported in Figure 14, the alarm event is managed by showing an overlay window containing a label which identifies the *device* that triggered the alarm and the *room* in which it is located, a *video* stream of what happens inside that room (if available) and *two buttons*, one for canceling the alarm and the other for handling the alarm in a safe way, for example by dialing 911 (or other emergency).

If the user does not look at any button, DOGeye chooses for her the safest option after a pre-defined timeout (currently set to 20 seconds), e.g., "Call 911" (as required from Guideline 1.3). This could happen, for example, when the user is not able to look at any buttons due to a loss of the eye tracker calibration.

Alarm events are examples of activities with critical priority, that interrupt other user actions on her home. This behavior fulfills Guideline 3.2: *Manage events with*

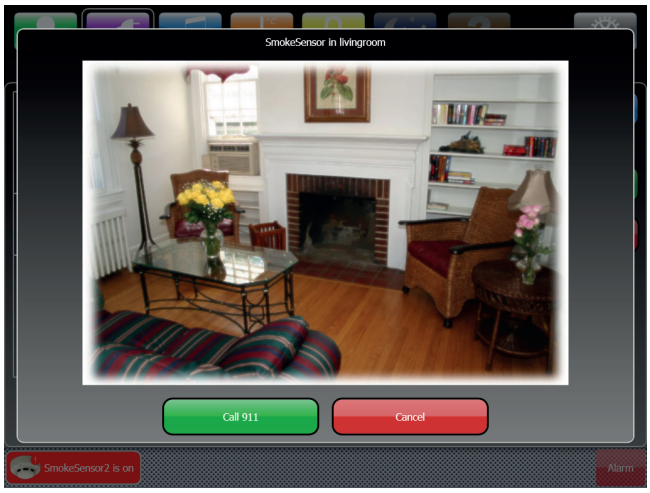


Figure 14: An alarm generated by a smoke sensor

different time critical priority and Guideline 3.3: *Execute commands with different priority*.

5.3. Interface Implementation

DOGeye is written in C#, using the *Windows Presentation Foundation* technology [27]. This solution allows to easily interface DOGeye with the ETU-Driver that is realized in C++ using COM Objects. It has a modular architecture shown in Figure 15 that also specifies the technologies adopted for the various modules and their communications. The main module is connected to external applications, i.e., Dog and the ETU-Driver.

The modular organization of DOGeye allows developers to easily edit the various parts of the program and possibly expand it to include new features.

The application includes a main window linked to *eight objects* representing the eight different tabs analyzed before. This main window also uses some configuration file and it is connected to *DOGleash* and an *eye tracking wrapper*. DOGleash is the library used for the connection with Dog, while the eye tracking wrapper is the library managing the interoperation with the ETU-Driver. These two libraries are obviously linked with Dog and with the eye tracking driver, respectively.

DOGeye was tested on *myTobii P10*[7], a remote and portable (but not wearable) eye tracker, running Windows XP and including a 15" single touch screen. Dog was locally installed on the eye tracker.

[7] `http://www.tobii.com/assistive_technology/products/mytobii_p10.aspx`

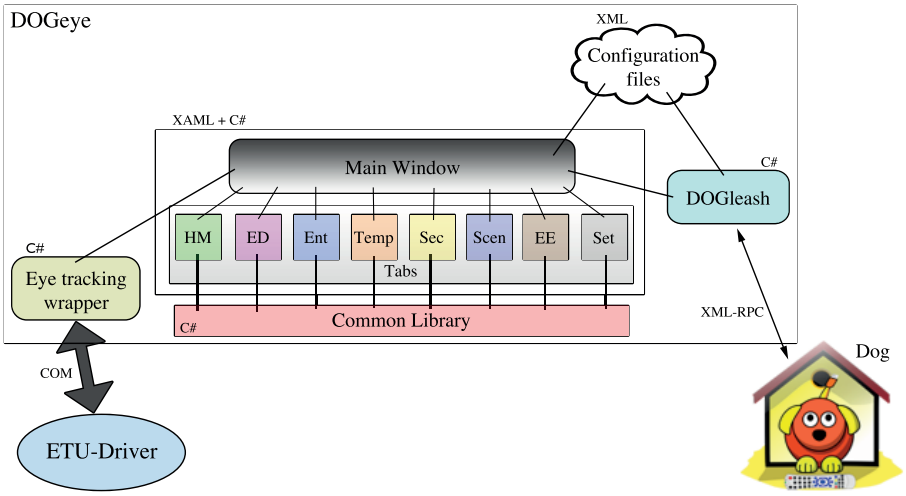


Figure 15: The general architecture of DOGeye

6. User Evaluation

The goal of the user evaluation is to identify the relative strengths and weaknesses of DOGeye, to roughly estimate the ability to use the interface without external hints and to check whether its advanced functionalities, such as multiple selection, are easy to discover and to use.

Eight participants used DOGeye in a controlled environment (described in 6.1) performing nine tasks (see Table 5) each, with Dog simulating the behavior of a realistic home through the DogSim capabilites [28]. Test tasks have been extrapolated from the COGAIN reference use cases and they have been tailored to the synthetic environment simulated by Dog; they reflect typical household tasks as also emerge from our design experience: we currently collaborate with the management staff of two domotic building: the Maison Equipée in Val d'Aosta and the Don Gnocchi Foundation in Italy. Participants never met during the evaluation. Their observations were used to allow qualitative analysis, to help identifying strong and weak points of the interface, and to identify future directions. Our analysis focuses on four basic questions to verify the usability of DOGeye:

1. How easily do users understand the tabbed organization? Are any icons hard to understand?
2. How easily and successfully do users find the options and tools they need?
3. How easily can the user control the smart home? What are the problems?
4. How easily can the user learn using the interface?

Eye control quality was not taken into account to study the usability of DOGeye, since we considered eye tracking simply as an input modality.

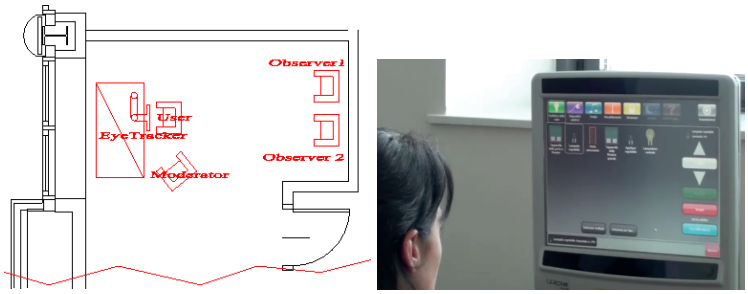
6.1. Methodology

We recruited 8 participants for our user study: 5 female and 3 male, aged 21 to 45 (with an average age of 31.3). Participants were selected for diverse skill level, especially about eye tracker usage. All except two worked in non-technology related fields, even if they use a computer more than four hours a day. Two groups of testers were selected: experienced (50%) and not experienced (50%) users. This design choice allows to gather some qualitative hint on the different needs DOGeye should be able to fulfill and on the degree to which such goal is reached. The study was held in Italian and the interface was localized accordingly.

A within-subject design was employed for both groups, where each subject in a group performed each task in counterbalanced order, to reduce order effects. We recorded a back video of the participant (Figure 16b), also capturing the screen content during the experiments.

6.1.1. Controlled Environment setup

Experiments were conducted into a controlled environment composed of a light-controlled room, inside which we positioned a desk carrying the eye tracker holding arm and the tracker itself. The eye tracker used in the study is the *myTobii P10* system and the adjustable holding arm allowed to set the tracker position to reach reasonable comfort of use for every study participant. The room used for the study hosted a moderator, seated near to the user, and two observers located in the background, not interfering with the test execution (typical *Simple Single-Room Setup*, see Figure 16). In general, methods followed recommendations for typical user studies [29].



(a) Environment layout (b) Environment setup

Figure 16: The controlled environment setup.

6.1.2. Test Deployment

After a short introduction to the study and the collection of demographic data, a static DOGeye screenshot was shown to the participant to collect a first impression on the interface, by querying her agreement level (from “Strongly disagree” to “Strongly agree”) about these sentences:

1. I like the appearance of the program.

2. I think that the program is intuitive.

3. I think that the program layout is understandable.

It must be noticed that the previous sentences are translated from the original Italian formulation, therefore they just provide a mean for the reader to understand the posed questions while they do not retain the meaning nuances and the carefully formulated wording we used.

Warm-Up. Afterwards the eye tracker was calibrated to the user's eye and the participant was introduced to a simple game named "*Lines*" (Figure 17), already available on our eye tracker, to get her used to the eye tracking interaction. The goal of the game is simple: move some colored balls in order to make a line of five equal elements.

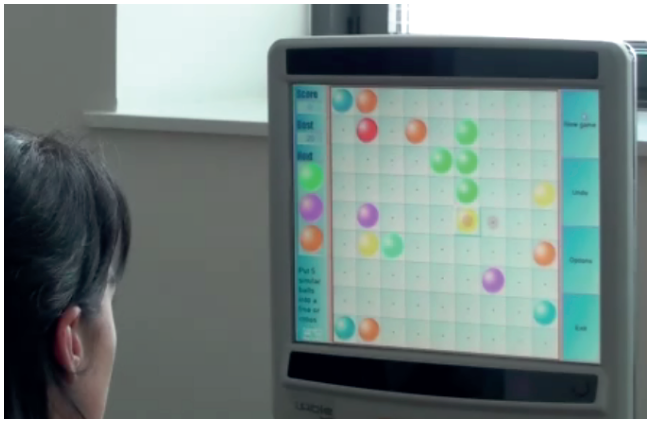


Figure 17: One participant uses Lines, during the study.

Task execution. After two or three completed lines, DOGeye was started. We found that a reasonable value to set the *dwell time* of the eye tracker was 1.5 seconds, so we used that setting during all the study. This dwell time is longer than usual (i.e., 500 ms) and it explicitly aims at reducing as much as possible the eye tracking access gap far un-experienced users. Each user was told to complete a set of nine task (see Table 5), one at a time. For two of them, particularly simple, the participant was asked to use the *think-aloud* protocol, to verify her actions. Examples of proposed tasks include: "Turn on the lamp in the living room" and "If the heating system in the bedroom and in the kitchen is off, turn it on".

Test conclusion. At the end, participants were given a questionnaire and asked to rate DOGeye in general, and to rate their agreement with the same sentences proposed earlier, just after seeing the screenshot of DOGeye. Users open comments or explanations were collected (e.g., problems found or explanation about something done during a task) through debriefing interviews. The duration of the entire experiment was dependent on eye tracker calibration problems and on how quickly participants answered the questions, but it ranged between 20 and 30 minutes.

Task	Description
T1	Turn on the lamp in the living room
T2	Plug the microwave oven in the kitchen
T3	Find a dimmer lamp, turn it on and set its luminosity to 10%
T4	Cancel the alarm triggered by the smoke detector
T5	Turn on all the lamps in the lobby
T6	If the heating system in the bedroom and in the kitchen is off, turn it on
T7	Set to 21 degree the heating system for the entire house
T8	Send a general alarm to draw attention in the house
T9	Read the smoke detector status and expand the video of the room to full screen

Table 5: The nine tasks used for the study

At the end of the study, we extrapolated from the videos the time (seconds) it took each participant to react to an alarm sent from the house, while the time it took participants to complete each task was not relevant due to the usage of the think-aloud protocol.

7. Results

We present and discuss quantitative as well as qualitative findings of our user study.

7.1. Quantitative results

According to Nielsen's Alertbox[8] we calculate the *success rate* of each participant as the percentage of tasks that users complete correctly, also giving partial credit for partially completed tasks, i.e., those tasks completed with minor errors. We expected the participants with higher eye tracker experience to perform much better than others. Table 6 reports the success rates of the study, using the Nielsen's Alertbox notation, where the first four participants belong to the group of eye tracking "experts" (E) while the others are the "non experts" (NE). In the table, "S" indicates a successful task, "P" a partial success, and "F" a failed task.

As expected, "expert" participants had a mean success rate of 91.67% (standard deviation 8.33%) while non-experts reached a satisfying, yet lower, rate of 86.11% (standard deviation 8.33%), with an overall average of 88.89%. These success rates provide a general picture of how DOGeye supports users and suggest that only minor adjustments are needed to the interface design.

The difference between the two user groups is clearly evident by looking at the time for replying to an alarm: the mean time for "experts" is 3.33 seconds (standard deviation 0.33 s) while the mean time for the others is 6.25 seconds (standard deviation 1.86 s), indicating that the difference of experience with the eye tracker favors the former.

[8] `http://www.useit.com/alertbox/20010218.html`

Table 6: Success rate of the study

User	T1	T2	T3	T4	T5	T6	T7	T8	T9	Success rate
E1	*P*	S	S	S	*P*	S	*P*	S	S	83.33%
E2	S	*P*	S	S	*P*	S	*P*	S	S	83.33%
E3	S	S	S	S	S	S	S	S	S	100.00%
E4	S	S	S	S	S	S	S	S	S	100.00%
NE1	S	S	S	S	S	S	S	S	S	100.00%
NE2	S	S	**F**	S	*P*	S	S	S	S	83.33%
NE3	S	S	S	S	*P*	S	S	*P*	S	88.89%
NE4	**F**	**F**	*P*	S	S	S	S	S	S	72.22%

E: expert-users, NE: non-expert users

7.2. Qualitative results

The final questionnaire asked participants to give an overall grade to DOGeye, in a scale from 1 (the worst) up to 5 (the best). Results showed that they were satisfied of DOGeye performance, with a mean value of 4.25 (standard deviation 0.37).

In the test conclusion (see Table 7 for the results), participants were asked to express their agreement about four sentences of which the first three are equal to the preliminary questions described in Section 6.1:

1. I like the appearance of the program.
2. I think that the program is intuitive.
3. I think that the program layout is understandable.
4. It is easy to learn how to use the program.

Results from participants were satisfying: most users agree or strongly agree with the proposed sentences about DOGeye (see the bottom of Table 7 for details).

By comparing the results of the first three questions, before and after the test, we see that 7 participants indicate an experience with the program better than they expected, as shown in the last column of Table 7. These results confirm our expectations about DOGeye design: it is rather easy to use, learn, and it achieves a satisfying fulfillment of COGAIN Guideline 4.2 about graceful and intelligible interfaces.

During the debriefing interviews, we collected some observations from the participants, about their behavior during the study and about what works in DOGeye.

All the participants observed that the name "Home Management" for indicating the tab with the home basic appliances is not clear; for example, some of them intuitively looked for lamps in "Electric Device". They suggest to divide the "Home Management" tab in two different tabs: "*Doors and Windows*" and "*Lighting*".

Table 7: Qualitative evaluation graded from Strongly disagree (1) to Strongly agree (5).

User	Question 1	Question 2	Question 3	Question 4	Better than expected?
E1	4	4	4	4	Yes
E2	4	4	4	5	Yes
E3	4	2	3	4	No
E4	4	4	4	4	Yes
NE1	4	4	4	4	Yes
NE2	5	4	4	4	Yes
NE3	4	5	4	4	Yes
NE4	4	5	5	5	Yes
Summary					
	7 agree 1 strongly agree	5 agree 2 strongly agree 1 disagree	6 agree 1 strongly agree 1 not agree or disagree	6 agree 2 strongly agree	7 yes 1 no

The *tab isolation* feature is "strange" for 7 out of 8 participants: i.e., they expected that once entered in a room in a defined tab, the program "remains" in that room when they change tab. So, they thought tabs as *different views* of the same house, instead of different "virtual houses" with different set of devices in it.

Only 3 users had difficulties to find the general "Alarm" button, placed in the bottom right corner of DOGeye: they look for it in the "Security" tab.

During the DOGeye study we have deactivated the "Scenarios" and "Everything else" tabs but we have asked participants what they expected to find in them. None understood what "Scenarios" tab includes, thinking about rules, home external views, external lights or music. Three of them, instead, understood the "Everything else" tab and other two subjects found it "useless".

An interesting thing we noticed is that none of the participants used the "Activate/Deactivate by type..." buttons, thus making the *implicit multiple selection*, present in the first two tabs, unnecessary. A good hint from a participant was to make the "Notification area" interactive, i.e., by offering the possibility to click on a notification to implicitly perform a single device selection, allowing the user act on the device.

Comments from users are, in general, very good: some of them appreciate the vocal feedback and the tab divisions, others the presence of the camera in the "Security" tab while some of them just found DOGeye "*very cool*".

7.3. Discussion

Overall, DOGeye evaluation is positive and provides useful insights on home-control applications explicitly designed for eye tracking support (COGAIN). By analyzing the success rate of each task, we noticed that tasks like *task #5* ("Turn on all the lamps in the lobby") are the most difficult for users, since they require quite advanced selection modalities. Referring to task #5, participants tend to turn the lamps on one by one, instead of using some kind of multiple selection.

The tab subdivision needs a refactoring as pointed out by user observations: we plan to split "Home Management" in two different tabs: "*Lighting System*" and "*Doors and Windows*". The tab "Everything else" will be removed, since few participants understand its meaning, and the *tab isolation* feature will be removed, thus offering different views of the same house when a user changes tab.

Since nobody used the *implicit multiple selection* present in the first two tabs, that feature will be removed: we keep only the other selection modalities, i.e., implicit selection, single selection and multiple selection. Implicit multiple selection will obviously remain in the "Temperature" tab, since it is the only viable modality for multiple selection in such a tab.

We are continuing to refine DOGeye, by adding functionalities such as a complete implementation of scenarios, a better visualization of the house map or the support for house on multiple floors. Based on our current results, we intend to implement these design and development changes and to conduct a sounder and deeper evaluation in a real smart home setting. We are working towards increasing the amount and quality of interaction for home inhabitants with or without mobility impairments.

8. Conclusions

We have introduced *DOGeye*, a multimodal eye-based application that may enable people with motor disabilities to control and manage their homes, thus living as autonomously as possible.

The paper describes the basic principle of eye tracking and presents the design and the implementation of DOGeye. We have discussed the various design issues, such as the use of both the structural view and the functional view of a home, by also referring to COGAIN Guidelines for gaze-based environmental control applications. Using the current implementation of DOGeye we conducted a first study with 8 subjects where we identified the relative strengths and weaknesses of DOGeye, roughly estimating the ability to use the interface without external hints and whether its advanced functionalities are easy to use. Results show that we have built a user interface that can successfully be used through eye interaction and that demonstrates only minor weaknesses in its design.

In future studies we plan to fix these weaknesses and to enhance the user interface so that the usability and the flexibility of the system could be improved.

References

[1] M. Weiser, The computer for the 21st century, SIGMOBILE Mob. Comput. Commun. Rev. 3 (3) (1999) 3–11. doi:10.1145/329124.329126.

[2] I. Bierhoff, et. al., Smart Home Environment, Tech. rep., Towards an Inclusive Future, COST219 Project (2007).

[3] A. van Berlo, Smart home technology: Have older people paved the way?, Gerontechnology 2 (1) (2002) 77–87. doi:10.4017/gt.2002.02.01.010.00.

[4] B. Zhang, P.-L. P. Rau, G. Salvendy, Design and evaluation of smart home user interface: effects of age, tasks and intelligence level, Behav. Inf. Technol. 28 (3) (2009) 239–249. doi:10.1080/01449290701573978.

[5] B. Chikhaoui, H. Pigot, Towards analytical evaluation of human machine interfaces developed in the context of smart homes, Interacting with Computers In Press, Corrected Proof (2010) –. doi:10.1016/j.intcom.2010.08.003.

[6] M. Chan, E. Campo, D. Estève, J.-Y. Fourniols, Smart homes – current features and future perspectives, Maturitas 64 (2) (2009) 90–97. doi:10.1016/j.maturitas.2009.07.014.

[7] A. J. Hornof, A. Cavender, Eyedraw: enabling children with severe motor impairments to draw with their eyes, in: CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM, New York, NY, USA, 2005, pp. 161–170. doi:10.1145/1054972.1054995.

[8] M. Donegan, L. Oosthuizen, R. Bates, G. Daunys, J. Hansen, M. Joos, P. Majaranta, I. Signorile, D3.1 User requirements report with observations of difficulties users are experiencing, Tech. rep., COGAIN (2005).

[9] R. Andrich, V. Gower, A. Caracciolo, G. Del Zanna, M. Di Rienzo, The DAT Project: A Smart Home Environment for People with Disabilities, in: K. Miesenberger, J. Klaus, W. Zagler, A. Karshmer (Eds.), Computers Helping People with Special Needs, Vol. 4061 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2006, pp. 492–499. doi:10.1007/11788713_74.

[10] T. Koskela, K. Väänänen-Vainio-Mattila, Evolution towards smart home environments: empirical evaluation of three user interfaces, Personal Ubiquitous Comput. 8 (3-4) (2004) 234–240. doi:10.1007/s00779-004-0283-x.

[11] F. Weingarten, M. Blumendorf, S. Albayrak, Towards multimodal interaction in smart home environments: the home operating system, in: DIS '10: Proceedings of the 8th ACM Conference on Designing Interactive Systems, ACM, New York, NY, USA, 2010, pp. 430–433. doi:10.1145/1858171.1858255.

[12] R. Bates, G. Daunys, A. Villanueva, E. Castellina, G. Hong, H. Istance, A. Gale, Lauruska, O. V. Spakov, P. Majaranta, D2.4 A survey of Existing 'de-facto' Standards and Systems of Environmental Control, Tech. rep., COGAIN (2006).

[13] F. Corno, E. Castellina, R. Bates, P. Majaranta, H. Istance, M. Donegan, D2.5 Draft standards for gaze based environmental control, Tech. rep., COGAIN (2007).

[14] R. Haber, M. Hershenson, Psychology of Visual Perception, Holt, Rinehart and Winston, 1973.

[15] R. J. K. Jacob, Eye Tracking in Advanced Interface Design, in: W. Barfield, T. Furness (Eds.), Advanced Interface Design and Virtual Environments, Oxford University Press, 1995, pp. 258—288.

[16] C. Ware, H. H. Mikaelian, An evaluation of an eye tracker as a device for computer input, in: CHI '87: Proceedings of the SIGCHI/GI conference on Human factors in computing systems and graphics interface, ACM, New York, NY, USA, 1987, pp. 183–188. doi:10.1145/29933.275627.

[17] M. Donegan, L. Oosthuizen, G. Daunys, H. Istance, R. Bates, I. Signorile, F. Corno, A. Garbo, L. Farinetti, E. Holmqvist, M. Buchholz, M. Joos, J. Hansen, D. MacKay, R. Eskillson, P. Majaranta, D3.2 Report on features of the different systems and development needs, Tech. rep., COGAIN (2006).

[18] F. Razzak, E. Castellina, F. Corno, Environmental Control Application compliant with COGAIN Guidelines, in: COGAIN 2009 - Gaze Interaction For Those Who Want It Most, 2009, pp. 31–34.

[19] V. Tanriverdi, R. J. K. Jacob, Interacting With Eye Movements In Virtual Environments, in: PROC. ACM CHI 2000 HUMAN FACTORS IN COMPUTING SYSTEMS CONFERENCE, Addison-Wesley/ACM Press, 2000, pp. 265–272.

[20] T. Saizmaa, H.-C. Kim, A Holistic Understanding of HCI Perspectives on Smart Home, in: NCM '08: Proceedings of the 2008 Fourth International Conference on Networked Computing and Advanced Information Management, IEEE Computer Society, Washington, DC, USA, 2008, pp. 59–65. doi:10.1109/NCM.2008.141.

[21] S. Lee, I. Ko, M. Kil, A user interface for controlling information appliances in smart homes, in: N. Nguyen, A. Grzech, R. Howlett, L. Jain (Eds.), Agent and Multi-Agent Systems: Technologies and Applications, Vol. 4496 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, Berlin, Heidelberg, 2007, Ch. 92, pp. 875–883. doi:10.1007/978-3-540-72830-6_92.

[22] E. Farella, M. Falavigna, B. Riccò, Aware and smart environments: The casattenta project, Microelectron. J. 41 (11) (2010) 697–702. doi:10.1016/j.mejo.2010.01.008.

[23] C. Abras, D. Maloney-Krichmar, J. Preece, User-centered design, in: In Bainbridge, W. Encyclopedia of Human-Computer Interaction, Thousand Oaks: Sage Publications, 2004, pp. 1–14.

[24] D. Bonino, E. Castellina, F. Corno, The DOG gateway: enabling ontology-based intelligent domotic environments, Consumer Electronics, IEEE Transactions on 54 (4) (2008) 1656–1664. doi:10.1109/TCE.2008.4711217.

[25] R. Bates, O. Spakov, D2.3 Implementation of COGAIN Gaze Tracking Standards, Tech. rep., COGAIN (2006).

[26] A. Dix, J. E. Finlay, G. D. Abowd, R. Beale, Human-Computer Interaction (3rd Edition), 3rd Edition, Prentice Hall, 2003.

[27] A. Nathan, Windows Presentation Foundation Unleashed (WPF) (Unleashed), Sams, Indianapolis, IN, USA, 2006.

[28] D. Bonino, F. Corno, DogSim: A state chart simulator for Domotic Environments, in: Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on, 2010, pp. 208–213. doi:10.1109/PERCOMW.2010.5470666.

[29] J. Rubin, D. Chisnell, Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests, 2nd Edition, Wiley, 2008.