AUGMENTED USER INTERACTION

R. Bacher, DESY, Hamburg, Germany

Abstract

The advent of advanced mobile, gaming and augmented reality devices provides users with novel interaction modalities. Speech, finger and hand gesture recognition or even gaze detection are commonly used technologies, often enriched with data from embedded gyroscope-like motion sensors. This paper discusses potential use cases of those technologies in the field of accelerator controls and maintenance. It describes the conceptual design of an intuitive, single-user, multi-modal human-machine interface which seamlessly incorporates actions based on various modalities in a single API. It discusses the present implementation status of this interface (Web2cHMI) within the Web2cToolkit framework. Finally, an outlook to future developments and ideas is presented.

MOTIVATION

Zooming applications by performing a pinch gesture at touch-sensitive displays, talking with the personal assistant to retrieve information from the internet, or controlling video games through a gaming console recognizing arm and body motions are all popular and intuitive interface features currently in common use. These technologies, well known in the consumer market, have extremely enriched the way in which people interact with their devices. Even in the rather conservative market of industrial applications, novel interaction technologies are gaining in importance, e.g. to simplify quality assurance of manufacturing processes in the car industry or to improve the efficiency of warehouse logistics.

In addition, a novel concept known as “App” and optimized for these unique technological features has been introduced which has revolutionised the conceptual design, look-and-feel and handiness of graphical user applications.

Hardware commissioning and maintenance use cases might profit from such novel interaction capabilities (modalities). For instance the alignment of mirrors mounted on an optical table to adjust a laser beam spot often requires a “third hand”. Interacting with control applications via spoken commands could be an appropriate alternative. Likewise wearing rough and dirty working gloves during cooling water maintenance work is not adequate for touch sensitive devices. Interacting via hand or arm gestures might be a better choice. Accessing on-line documentation is often indispensable for efficient inspection work. Wearing see-through augmented reality glasses controlled by head movements displaying routing schemes alongside with control applications could substantially improve measurement operations in the field.

Even control room work provides use cases for novel interaction modalities. Remote-controlling an overhead mounted screen showing some overview or control application panels might be considerably simplified by recognizing spatial gestures such as clenching a fist or snapping fingers. Beam steering requires uninterrupted eye contact with trend charts or other display updates. Controlling a virtual knob by recognizing hand rotation could eliminate the risk of losing device focus which often happens with mouse-based operations.

Today’s youth are more than familiar with these novel interaction capabilities and app-like user applications. Providing up-to-date tools for future control room operators and maintenance technicians appears to be a must.

A PARADIGM CHANGE

Today’s users of accelerator control applications have developed intuitions based on click-like interactions. In an accelerator control room the mouse is still the standard user input device to interact with graphical applications. Being well accepted by the operators it provides a very accurate pointing capability and standardized user actions normally associated with graphical widgets. Mouse-based interactions are highly reliable unambiguous single-user actions. They are best suited for complex applications containing a wealth of graphical widgets.

Thus the introduction of any new interaction capabilities will be accompanied by a serious paradigm change regarding how software programmers design graphical operations and maintenance applications and how operators or maintenance personnel interact with them.

Gesture-Based Interaction

Spatial hand- and arm gestures provide only a rough pointing capability, and experience shows that the user’s arm tends to fatigue quickly, a phenomenon known as “gorilla arm”. In addition head gestures such as turning or nodding might also be considered.

In practice only a very limited number of gesture types are available which are partially standardized and not a priori associated with graphical widgets. Consequently the design and look-and-feel of applications must accommodate these restrictions. A multi-page application design consistent with the app concept, where each page provides a well-confined and standardized functionality with an unambiguous gesture-to-action mapping, appears to be best suited.

In general hand- and arm gestures are less reliable and require a specific arming / disarming procedure to prevent the user from unwanted interaction.

Spatial gestures are not limited to single-user interaction only. It depends on the technology of the gesture recognition device used how many gestures from different persons can be tracked individually. Devices with embedded infrared stereo cameras, multi-axis gyro sensors or muscle activity sensors are commercially available.

Speech-Based Interaction

In contrast to other interaction modalities the recognition of spoken commands does not provide any pointing capability at all.

On the one hand the huge word pool of human languages is a clear plus factor. On the other hand the context-dependent ambiguity and the language- or dialect-dependence of the vocabulary pose a big challenge for the recognition algorithms involved.

From the audio technical point of view the recognition of spoken commands might suffer from ambient noise and the interference of multi-user inputs.

To achieve a reliable speech recognition ability limiting the allowed vocabulary and ensuring an unambiguous word-to-action mapping is preferable. Similar to gesture recognition a specific arming / disarming procedure is capable to prevent the user from unwanted interaction.

A PRACTICAL EXAMPLE

This paper reports ongoing R&D work and describes a common single-user human-machine interface which seamlessly combines actions based on various modalities provided by input devices commonly available from the consumer market. It presents and discusses a platform-neutral Web-based interface implementation (Web2cHMI [1]) for accelerator operation and maintenance applications in the context of the Web2cToolkit Web service collection [2].

Web2cHMI defines a set of common user interactions comprising all actions needed to control Web2cToolkit-compliant Web applications such as application browsing, display zooming or executing commands associated with interactive graphical widgets. In general it can be considered as a prototype implementation exploring the advantages and disadvantages of these novel interaction modalities for accelerator control and maintenance applications.

In particular the Web2cToGo Web service [3] implements Web2cHMI and provides a test environment for identifying intuitive and handy user actions as well as investigating the proper structure, design and operability of multi-modal accelerator operation and maintenance applications. Web2cToGo embeds both Web2cViewer and Web2cArchiveViewer Web application which are also members of Web2cToolkit Web service collection.

Supported Modalities

Web2cHMI supports various modalities which can be used simultaneously including

- 1D/2D flat gestures including single-finger actions (mouse) and single- or multi-finger gestures (touch-sensitive display)
- 2D/3D spatial gestures including hand-gestures (LEAP Motion controller (Figure 1) [4]), hand- or arm-gestures (Myo gesture control armband (Figure 2) [5]) and 3-axis (yaw, pitch roll) head movements (smart glasses)
- English spoken commands (Sphinx speech recognition [6]).

The LEAP Motion controller and the Myo gesture control armband are connected locally with their corresponding host device (desktop, notebook or tablet computers) through USB and Bluetooth, respectively. Currently supported smart glasses with gyroscope-based head tracking capability include Epson Moverio BT-200 [7] and Vuzix M100 (Figure 3) [8].

Supported Gesture Types

Web2cHMI recognizes various primitive, i.e. native or input device-specific gestures including

- Mouse: *Click, Move*
- Touch-sensitive display: *Tap, Move / Swipe, Pinch* (two fingers)
- LEAP Motion controller: *Key-Tap, Swipe, Open-Hand, Closed-Hand, Circle*
- Myo gesture control armband: *Double-Tap, Wave-Out / Wave-In, Fingers-Spread, Fist*
- Smart glass: *Move-Fast / Move-Slow, Roll*

In addition, enriched gestures formed by primitive gestures followed by moves or rotations etc. are supported.

All gesture-capable devices provide orientation data being used to position a virtual cursor label at an application window. Unlike mice or touch-sensitive displays gesture recognition devices such as LEAP, Myo or smart glasses with head tracking capability do not allow an accurate positioning of the cursor label.

Figure 1: LEAP Motion sensor.

Figure 2: Myo gesture control armband.

Figure 3: VUZIX M100 with gyroscope-based head tracking capability.

ISBN 978-3-95450-189-2

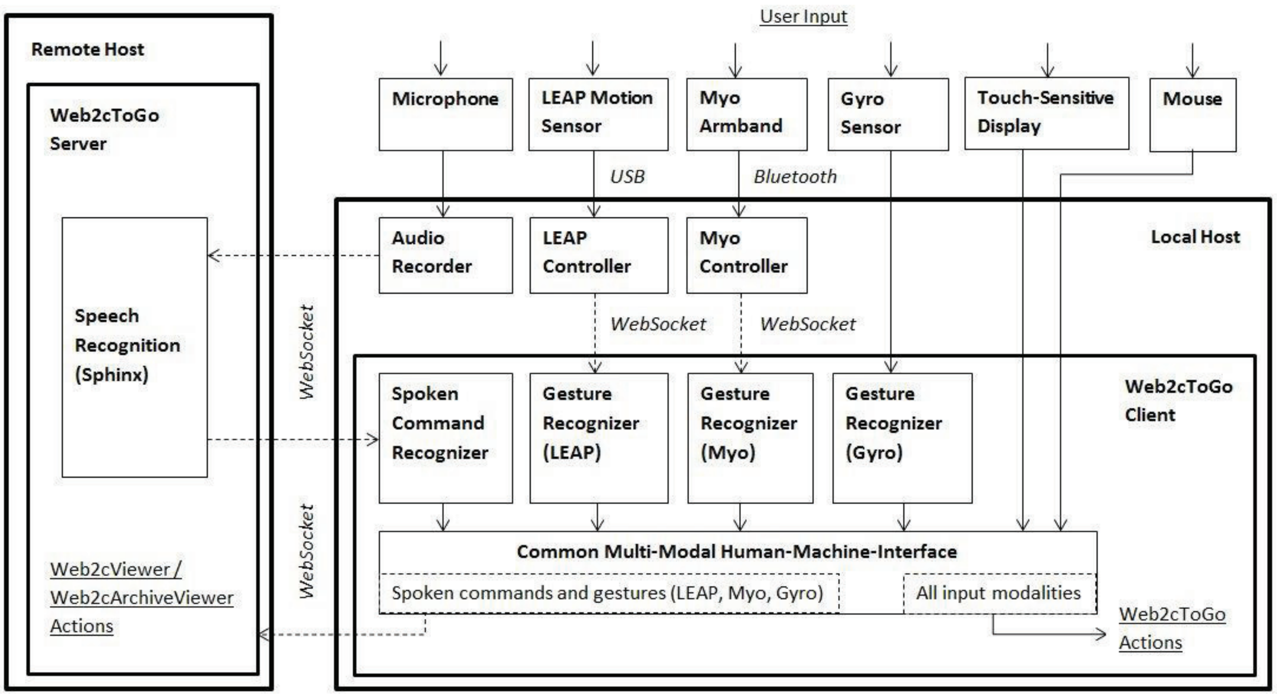

Figure 4: Web2cToGo / Web2cHMI user input data flow.

If applicable or required by ergonomics principles, different gestures may be applied by right or left handed individuals. If a gesture has been successfully recognized the next gesture recognition is momentarily inhibited while the cursor label is fixed to the center of the application window to notify the user.

In addition to avoid unwanted responses to unintentional gestures recognition ability must be armed / disarmed explicitly by the user:

- LEAP Motion controller: *Key-Tap / Key-Tap*
- Myo gesture control armband: *Double-Tap / Double-Tap*
- Smart glass: *Clockwise-Roll - Counter-Clockwise-Roll / Counter-Clockwise-Roll - Clockwise-Roll*

Common Human-Machine-Interface

Web2cHMI analyses user actions recorded by any modality-specific input device attached and maps the recognized gestures, spoken commands, head movements or even mouse clicks etc. to unambiguous commands. The recognized commands are used to control both the Web2cToGo framework application itself and the embedded Web2cToolkit-compliant Web applications. Figure 4 sketches the user input data-flow within Web2cToGo. Besides speech recognition which is performed by the Web2cToGo servlet (Java) at the Web server all recognition algorithms are implemented as client-side JavaScript modules being executed by an HTML5-compliant Web browser. Commands dedicated to an embedded application are redirected to the corresponding Web application.

Web2cHMI recognizes, for instance, among other user input the following corresponding Web2cViewer actions to increase a set value of an attached controls device in small steps using the slider widget (Figure 5):

- Mouse: *Click* ">"-button and *Click* "Set Value"-button of the slider widget
- Touch-sensitive display: *Tap* ">"-button and *Tap* "Set Value"-button of the slider widget (right or left hand)
- LEAP Motion Controller: *Clockwise Circle* (right or left hand)
- Myo gesture control armband: *Fist & Clockwise Rotation* (right or left arm)
- Gyro Sensor: *Upward Left-Tilted Move-Slow*
- Speech Recognition: "*More*"

Similarly, in order to choose an item above the currently chosen item in a list box of the Web2cArchiveViewer application the following actions can be performed (Figure 6):

- Mouse: *Click* on item in list box
- Touch-sensitive display: *Tap* on item in list box (right or left hand)
- LEAP Motion Controller: *Upward Long Swipe* (right or left hand)
- Myo gesture control armband: *Wave-Out & Upward Move* (right or left arm)
- Gyro Sensor: *Upward Move-Slow*
- Speech Recognition: "*Browse Up*"

A compilation of all commands implemented by Web2cHMI is given in the Appendix.

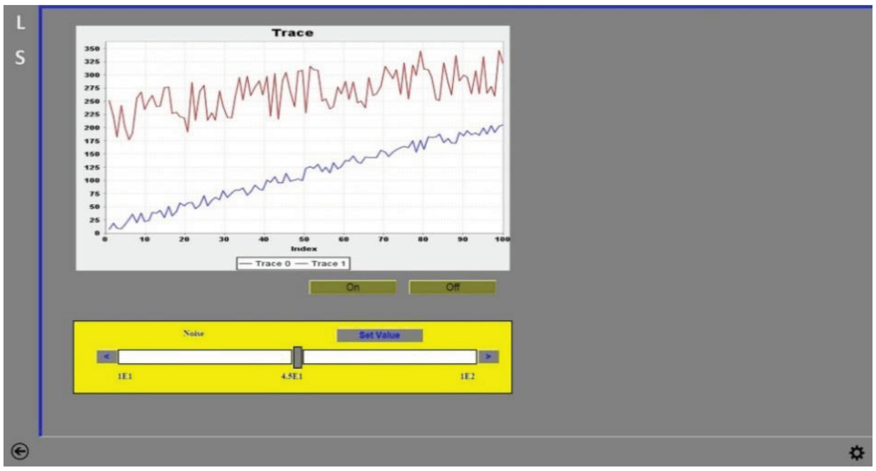

Figure 5: Web2cViewer embedded in Web2cToGo application (Operation View).

Similarly, in order to choose an item above the currently chosen item in a list box of the Web2cArchiveViewer application the following actions can be performed (Figure 6):

- Mouse: *Click* on item in list box
- Touch-sensitive display: *Tap* on item in list box (right or left hand)
- LEAP Motion Controller: *Upward Long Swipe* (right or left hand)
- Myo gesture control armband: *Wave-Out & Upward Move* (right or left arm)
- Gyro Sensor: *Upward Move-Slow*
- Speech Recognition: "*Browse Up*"

A compilation of all commands implemented by Web2cHMI is given in the Appendix.

Standardized Multi-Page Application Design

Due to the limited number of available gestures embedded accelerator controls and maintenance applications have to be split into individual pages which provide well-known, standardized functionality in order to preserve an unambiguous gesture-to-command mapping.

Following this concept each Web2cViewer application page might contain a single widget instance of each of the following interactive widget types (Figure 5). According to their type, the interactive widgets are capable of performing a specific, predefined user action such as opening a vacuum valve or changing a set value of a power supply:

- On-type Button (user action = "On")
- Off-type Button (user action = "Off")
- Slider (user action = "Set Value")
- Chart (user action = "Zoom Data")

In addition a page might contain an unlimited number of passive Web2cViewer widgets such as labels or value fields.

Similarly, sets of interactive widgets have been defined for Web2cArchiveViewer application pages.

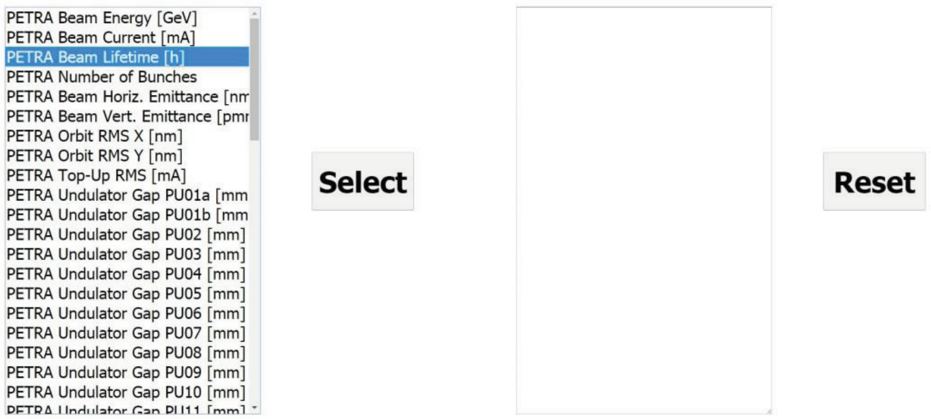
PETRA Beam Energy [GeV]
PETRA Beam Current [mA]
PETRA Beam Lifetime [h]
PETRA Number of Bunches
PETRA Beam Horiz. Emittance [nm
PETRA Beam Vert. Emittance [pmr
PETRA Orbit RMS X [nm]
PETRA Orbit RMS Y [nm]
PETRA Top-Up RMS [mA]
PETRA Undulator Gap PU01a [mm
PETRA Undulator Gap PU01b [mm
PETRA Undulator Gap PU02 [mm]
PETRA Undulator Gap PU03 [mm]
PETRA Undulator Gap PU04 [mm]
PETRA Undulator Gap PU05 [mm]
PETRA Undulator Gap PU06 [mm]
PETRA Undulator Gap PU07 [mm]
PETRA Undulator Gap PU08 [mm]
PETRA Undulator Gap PU09 [mm]
PETRA Undulator Gap PU10 [mm]
PETRA Undulator Gap PU11 [mm]

Figure 6: Web2cArchiveViewer embedded in Web2cToGo application (Operation View).

ISBN 978-3-95450-189-2

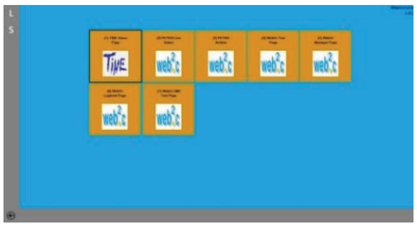
Figure 7: Web2cToGo (Explorer View).

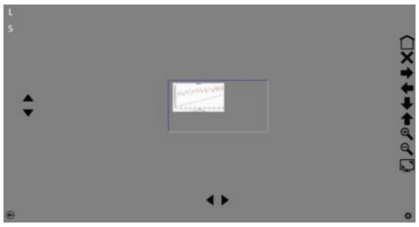
Figure 8: Web2cToGo (Navigation View).

OPEN ISSUES

The work described in this paper is still an R&D project. It is inspired and abetted by a growing number of consumer and industrial use cases. It is expected that the acceptance level of modern user interaction technologies will steadily increase in the future.

To compete successfully with standard mouse-based interactions the quality and reliability of gesture and speech recognition procedures has to be improved. The applicability for accelerator controls and maintenance has to be studied in detail. The most intuitive, most common and best matching sets of gestures and spoken commands have to be defined and the usefulness and potential predominance of this approach have to be explored and proved in real field tests.

In addition, it is an open issue how a future control room preserving the unambiguity of operator's interactions might look. Will it be a camera supervised collective multi-user attentive environment or rather an environment for operators wearing individual single-user augmented reality glasses?

Finally, the next steps to introduce these novel interaction technologies for accelerator operations and maintenance have to be discussed. Is an intermediate step preferable providing, for instance, user applications adapted for touch pads or interactive tables yet keeping the traditional application look-and-feel? Or should a larger step be taken, where mouse-click interactions are omitted entirely, thereby skipping the familiar mouse-centric application design pattern?

APPENDIX

Implemented Web2cHMI commands include:

- *Web2cToGo (Explorer View)*: launch / display selected application, select application icon above below / right / left (Figure 7)
- *Web2cToGo (Navigation View)*: switch to operation view, switch to explorer view, browse to previous / next application, browse to previous / next application page, close current application, zoom in / zoom out current application page, fit current application page to window size, scroll down / scroll up / scroll right / scroll left current application page (Figure 8)
- *Web2cToGo (Operation View)*: Switch to navigation view (Figure 5)
- *Web2cViewer*: activate on-button / off-button, change set-value (any value / small positive step / small negative step, big positive step, big negative step), zoom in chart (left data area / center data area / right data area), reset chart zoom,
- *Web2cArchiveViewer (Single Entry / Composite Entry Data Selector)*: choose channel above / below chosen channel, select chosen channel, clear all entries in list of selected channels (Figure 6)
- *Web2cArchiveViewer (Date and Time Span Selector)*: choose next / previous month of year, select day, select day above / below / right of / left of selected day, select time span above / below selected time span, select default time span
- *Web2cArchiveViewer (Single Data Chart)*: retrieve data, retrieve data from next / previous time interval, zoom in chart (left data area / center data area / right data area), reset chart zoom
- *Web2cArchiveViewer (Multiple Data Chart)*: retrieve data, retrieve data from next / previous time interval, zoom in chart (left data area / center data area / right data area), reset chart zoom, scroll down / scroll up visible pane
- *Web2cArchiveViewer (Data Table)*: retrieve data, retrieve data from next / previous time interval

REFERENCES

[1] R. Bacher, "A Multi-Modal Human-Machine-Interface for Accelerator Operation and Maintenance Applications", in *Proc. ICALEPCS'15*, Melbourne, Australia, Oct. 2015, paper WEM308, pp 677.

[2] Web2cToolkit, `http://web2ctoolkit.desy.de`

[3] R. Bacher, "Web2cToGo: Bringing the Web2cToolkit to Mobile Devices", in *Proc. PCaPAC'12*, Kolkata, India, Dec. 2012, paper WEIC01, pp. 4.

[4] LEAP, `https://www.LEAPmotion.com`

[5] Myo, `https://www.myo.com`

[6] Sphinx-4, `http://cmusphinx.sourceforge.net/sphinx`

[7] Epson Moverio BT-200, `https://www.vuzix.com`

[8] Vuzix M100, `https://www.epson.com`

ISBN 978-3-95450-189-2