ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-THCB3

IMPROVING Web2cHMI GESTURE RECOGNITION USING MACHINE LEARNING

R. Bacher, Deutsches Elektronen-Synchrotron DESY, 22607 Hamburg, Germany

Abstract

Web2cHMI is multi-modal human-machine interface which seamlessly incorporates actions based on various interface modalities in a single API, including finger, hand and head gestures as well as spoken commands. The set of native gestures provided by off-the-shelf 2D- or 3D-interface devices such as the Myo gesture control armband can be enriched or extended by additional custom gestures. This paper discusses a particular method and its implementation in recognizing different finger, hand and head movements using supervised machine learning algorithms including a non-linear regression for feature extraction of the movement and a k-nearest neighbour method for movement classification using memorized training data. The method is capable of distinguishing between fast and slow, short and long, up and down, or right and left linear as well as clockwise and counter clockwise circular movements, which can then be associated with specific user interactions.

INTRODUCTION

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.

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. Therefore, any new interaction capability such as gesture or spoken command recognition will only be accepted by the users if they provide comparable or even better handiness, ease-of-use and reliability to click-like interactions.

This paper discusses a particular method and its implementation aiming at improving the quality and reliability in recognizing gestures based on different finger, hand and head movements using supervised machine learning algorithms. This includes a non-linear regression for extracting the features of the movement and a k-nearest neighbour method for movement classification using memorized training data.

Web2cTOOLKIT

The Web2cToolkit [1] is a collection of Web services, i.e. servlet applications and the corresponding Web browser applications, including

- Web2cViewer: Interactive synoptic live display to visualize and control accelerator or beam line equipment,
- Web2cViewerWizard: Full-featured graphical editor to generate and configure synoptic live displays,
- Web2cArchiveWizard: Web form to request data from a control system archive storage and to display the retrieved data as a chart or table,
- Web2cArchiveWizardWizard: Full-featured graphical editor to generate and configure archive viewer displays,
- Web2cToGo: Interactive display especially designed for mobile devices embedding instances of all kinds of Web2cToolkit Web client applications,
- Web2cGateway: Application programmer interface (HTTP-gateway) to all implemented control system interfaces,
- Web2cManager: Administrators interface to configure and manage the Web2cToolkit.

The Web2cToolkit provides interfaces to major accelerator and beam line control systems including TINE [2], DOOCS [3], EPICS [4], TANGO [5] and STARS [6].

Web2cHMI

The Web2cHMI [1] is a platform-neutral Web-based human-machine-interface implementation for accelerator operation and maintenance applications in the context of the Web2cToolkit Web service collection. It 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 [7]), hand- or arm-gestures (Myo gesture control armband [8]) and 3-axis (yaw, pitch roll) head movements (smart glasses including Epson Moverio BT-200 [9] and Vuzix M100 [10])
- English spoken commands (Sphinx speech recognition [11]).

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-THCB3

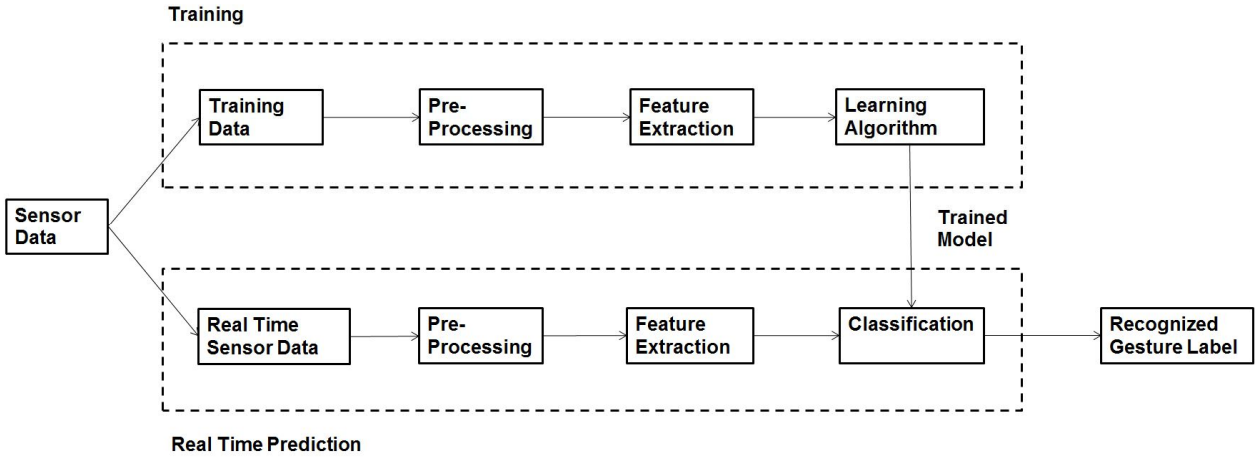

Figure 1: Gesture recognition workflow using supervised machine learning (classification).

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.

The Sphinx speech recognition system knows a Web2cHMI-specific vocabulary listed in a dictionary such as "*Browse Up*" or "*Lot More*".

GESTURE RECOGNITION WORKFLOW USING MACHINE LEARNING

Web2cHMI gesture recognition using supervised machine learning consists of two distinct phases [Fig. 1], training phase and real time prediction phase. The input for both phases is continuously recorded position data of the user's input device in both Cartesian (X, Y) and polar (R, PHI) coordinates resulting from finger, hand, arm or head movements. The continuous stream of sensor data is pre-processed to reduce the noise floor of the measurement or to remove obvious outliers. Next, the cleansed data are fed into a mathematical algorithm to extract a set of parameters (features) representative for a particular gesture.

During the training phase the extracted feature values are used by a supervised learning algorithm to train a model resulting in sets or clusters of learned data representing valid gesture types. The Web2cHMI learning algorithms provides four distinct learned data sets representing *Long-Slow*, *Long-Fast*, *Short-Slow* or *Short-Fast* movements. During real time prediction phase the extracted feature values are classified by comparing with the memorized learned data sets. The result of this classification analysis is the proper gesture label.

It is important that both training and real time prediction phase are using the same pre-processing and feature extraction algorithms.

The Web2cHMI gesture recognition algorithm is entirely coded in JavaScript capable of being processed locally by a standard Web-browser and does not rely on any server-side logic.

Feature Extraction

The feature extraction algorithm implements a non-linear regression (Nelder-Mead method) to fit consecutive 333 ms long time sequences of sensor data to a predefined mathematical model [Eq. 1]:

$$\begin{aligned} &\mathbf{t} \le \mathbf{t_{start}}: && \mathbf{f(t) = s_{start}} \\ &\mathbf{t_{start} < t < t_{end}}: && \mathbf{f(t) = s_{start} + \left(\left(\frac{s_{end} - s_{start}}{t_{end} - t_{start}} \right) * (t - t_{start}) \right)} \quad (1) \\ &\mathbf{t \ge t_{end}}: && \mathbf{f(t) = s_{end}} \end{aligned}$$

Here, the parameters t_{start} and t_{end} mark the start and end time of the recognized gesture within the analysed user's finger, arm or head movement, respectively. Similar, s_{start} and s_{end} are indicating the start and end position of the gesture.

The regression reduces the dimension of the gesture recognition task by two orders of magnitude and is performed for each Cartesian coordinate orientation (linear horizontal movement, linear vertical movement) separately. Fitting the polar angle PHI allows the identification of circular movements. If the regression algorithm converges and the parameters are confined within reasonable limits, the duration ($t_{end} - t_{start}$) and the length ($s_{end} - s_{start}$) of the gesture are calculated.

Figure 2 is exemplary and shows a recognized linear diagonal gesture (horizontal (red): rightward movement, vertical (blue): upward movement) and the corresponding values for t_{start}, t_{end}, s_{start}, s_{end} (horizontal (red) and vertical (blue) marker lines).

ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-THCB3

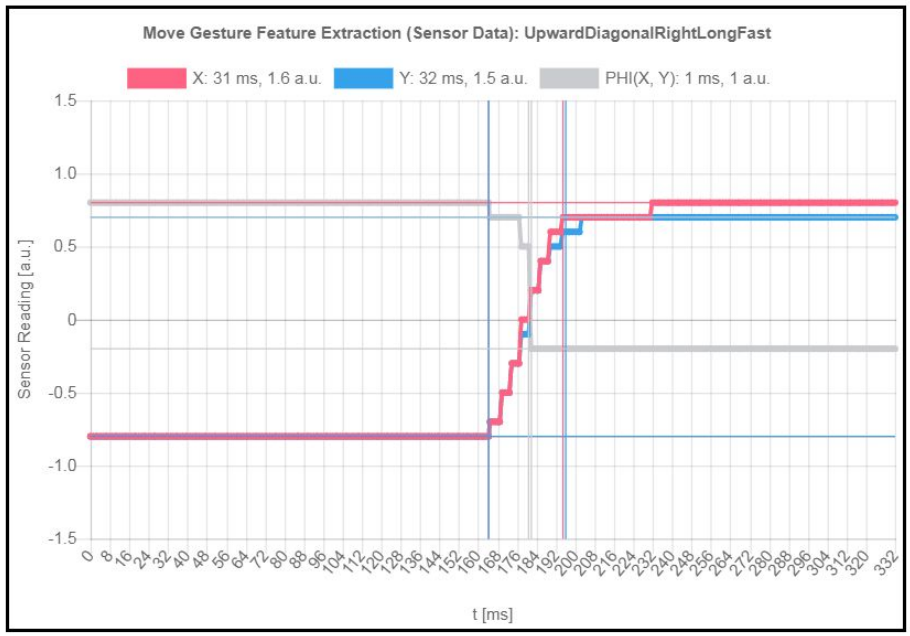

Figure 2: Regression identifying a linear diagonal gesture (horizontal (red): left to right, vertical (blue): bottom to top) and the corresponding values for t_{start}, t_{end}, s_{start}, s_{end} (horizontal (red) and vertical (blue) marker lines).

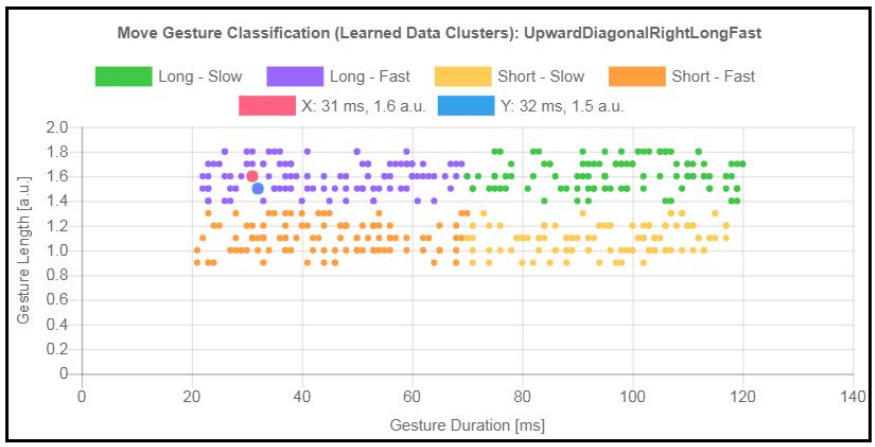

Figure 3: Predicted gesture duration / length (horizontal: red dot, vertical: blue dot) and learned data clusters. The recognized gesture is classified as a linear diagonal (left to right / bottom to top movement) *Long-Fast* gesture.

Classification

Finally, the classification algorithm delivers the orientation and the type of the recognized gesture.

Comparing s_{start} and s_{end} allows distinguishing between left / right linear, up / down linear and clockwise / counter clockwise circular movements.

The duration and length of the gesture are used as input for a k-nearest neighbour (k = 3) analysis. The algorithm calculates the Euclidean distance between the predicted data point (duration / length) and each memorized learned data point. It searches the k nearest neighbours to identify the learned data cluster the predicted data point is belonging to (Fig. 3).

REFERENCES

[1] Web2cToolkit; http://web2ctoolkit.desy.de
[2] TINE; http://tine.desy.de
[3] DOOCS; http://doocs.desy.de
[4] EPICS; http://www.aps.anl.gov/epics
[5] TANGO; http://www.tango-controls.org
[6] STARS; http://stars.kek.jp
[7] LEAP; https://www.LEAPmotion.com
[8] Myo; https://www.myo.com
[9] Sphinx-4; http://cmusphinx.sourceforge.net/sphinx
[10] Epson Moverio BT-200; https://www.vuzix.com/
[11] Vuzix M100; https://www.epson.com/