FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

FACULTY OF ENGINEERING

Intent detection and somatosensory feedback

#07: Signal processing, feature extraction

Claudio CASTELLINI, Sabine THÜRAUF

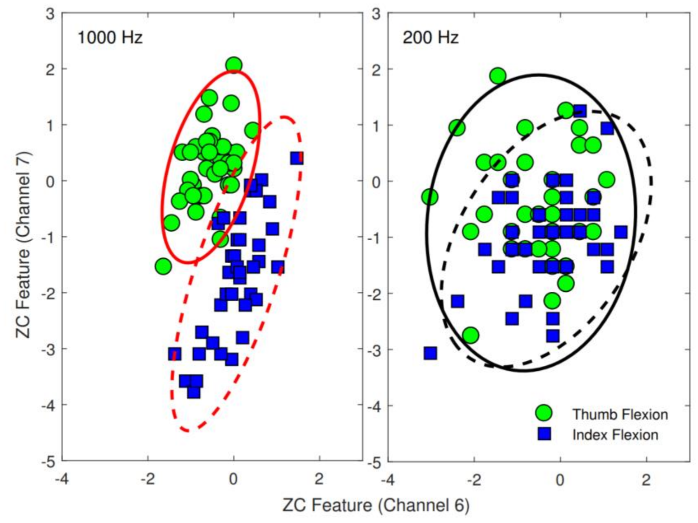


Figure 2. Differences in EMG patterns between using: (**left**) a 1000 Hz sampling rate; and (**right**) a 200 Hz sampling rate. ZC features are extracted from two different EMG channels (6 and 7) during thumb flexion (green circle markers and solid lines) and index flexion (blue square markers and dashed lines). Samples are from Subject 1 of Database 3.

EMG patterns related to two actions. Reproduced from Angkoon Phinyomark, Rami N. Khushaba and Erik Scheme, *Feature Extraction and Selection for Myoelectric Control Based on Wearable EMG Sensors*, MDPI Sensors 2018, 18, 1615

The *rubber hand illusion.* See Botvinick M, Cohen J., *Rubber hands 'feel' touch that eyes see.* Nature. 1998 Feb 19;391(6669):756. doi: 10.1038/35784. PMID: 9486643.

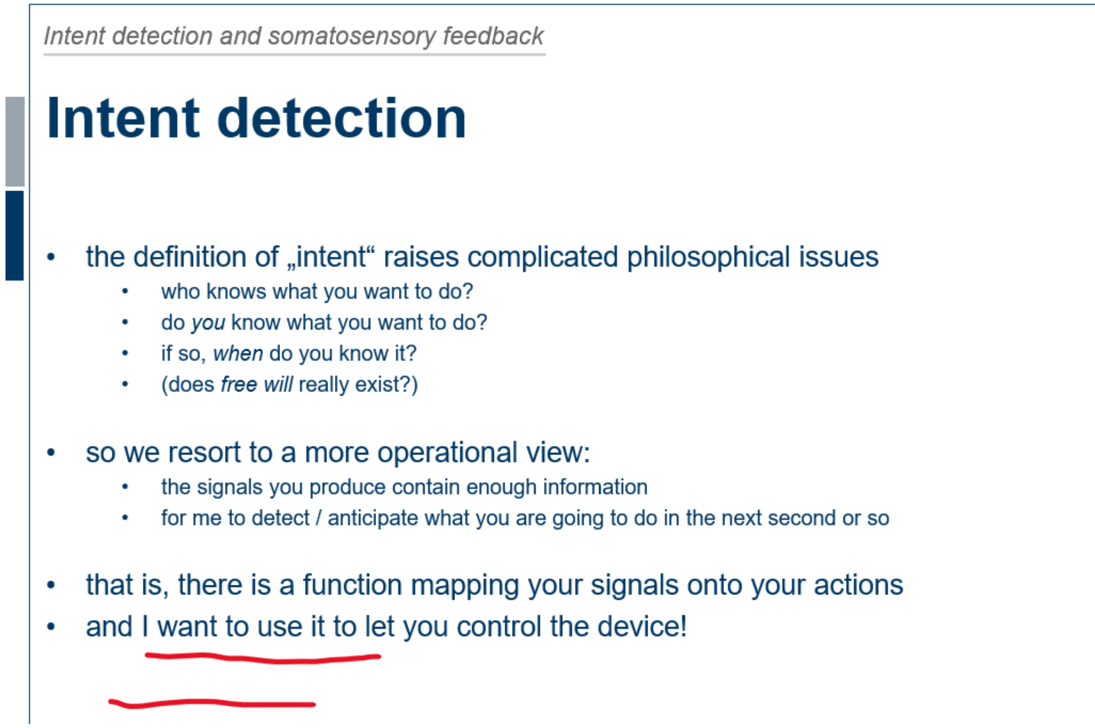
Lecture #07:

Signal processing, feature extraction

- the intent detection pipeline + a less standard one
- filtering, feature extraction
- the TAC test

Machine learning and I.D.

- recall Lecture #2:
- we think there is a *function* out there
- we want to build an approximant
- starting from data collected from a participant / user



Intent detection and somatosensory feedback

Intent detection

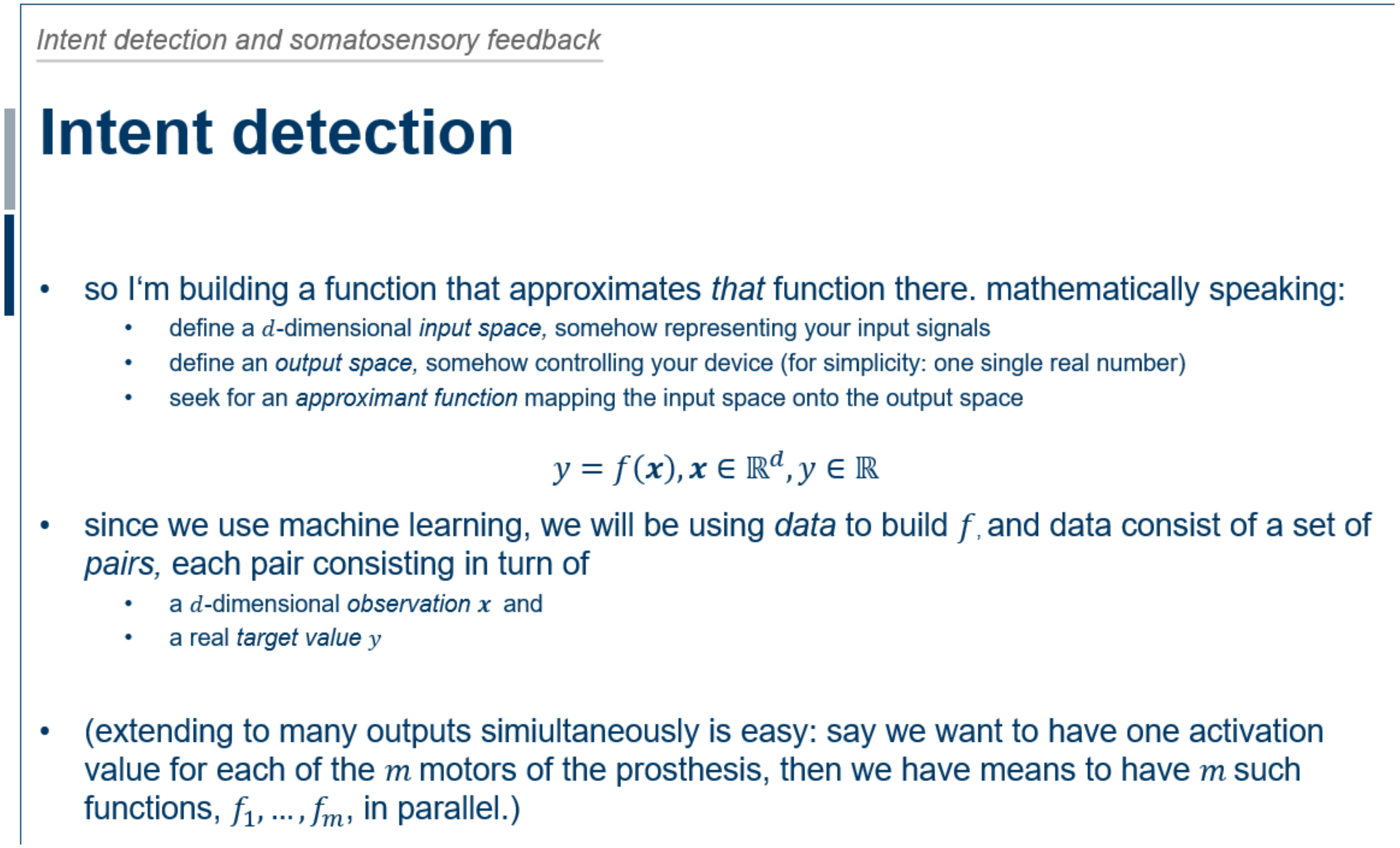
- the definition of „intent“ raises complicated philosophical issues
  - who knows what you want to do?
  - do *you* know what you want to do?
  - if so, *when* do you know it?
  - (does *free will* really exist?)
- so we resort to a more operational view:
  - the signals you produce contain enough information
  - for me to detect / anticipate what you are going to do in the next second or so
- that is, there is a function mapping your signals onto your actions
- and I want to use it to let you control the device!

Machine learning and I.D.

- so we use ML to that aim.
- notice that in some cases this is *not necessary*
- but for advanced prostheses and reha device, it is…
- …alas.

Machine learning and I.D.

- recall Lecture #2:



Intent detection

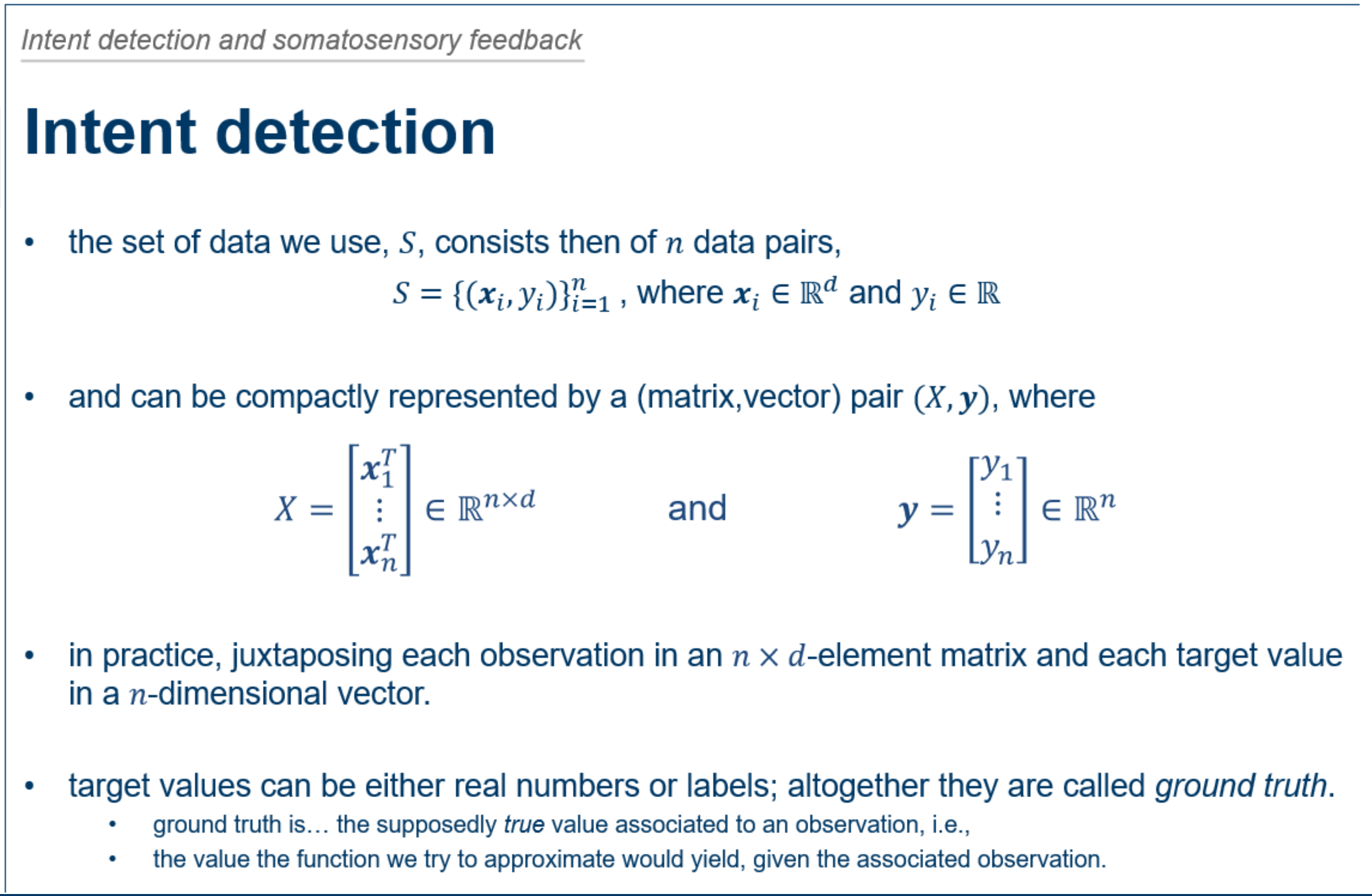
- so I'm building a function that approximates *that* function there. mathematically speaking:
  - define a d-dimensional *input space*, somehow representing your input signals
  - define an *output space*, somehow controlling your device (for simplicity: one single real number)
  - seek for an *approximant function* mapping the input space onto the output space

$$y = f(\boldsymbol{x}), \boldsymbol{x} \in \mathbb{R}^d, y \in \mathbb{R}$$

- since we use machine learning, we will be using *data* to build f, and data consist of a set of *pairs*, each pair consisting in turn of
  - a d-dimensional *observation* $\boldsymbol{x}$ and
  - a real *target value* y
- (extending to many outputs simiultaneously is easy: say we want to have one activation value for each of the m motors of the prosthesis, then we have means to have m such functions, $f_1, \dots, f_m$, in parallel.)

Machine learning and I.D.

- recall Lecture #2:



Intent detection

- the set of data we use, S, consists then of n data pairs,

  $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$

- and can be compactly represented by a (matrix,vector) pair $(X, \boldsymbol{y})$, where

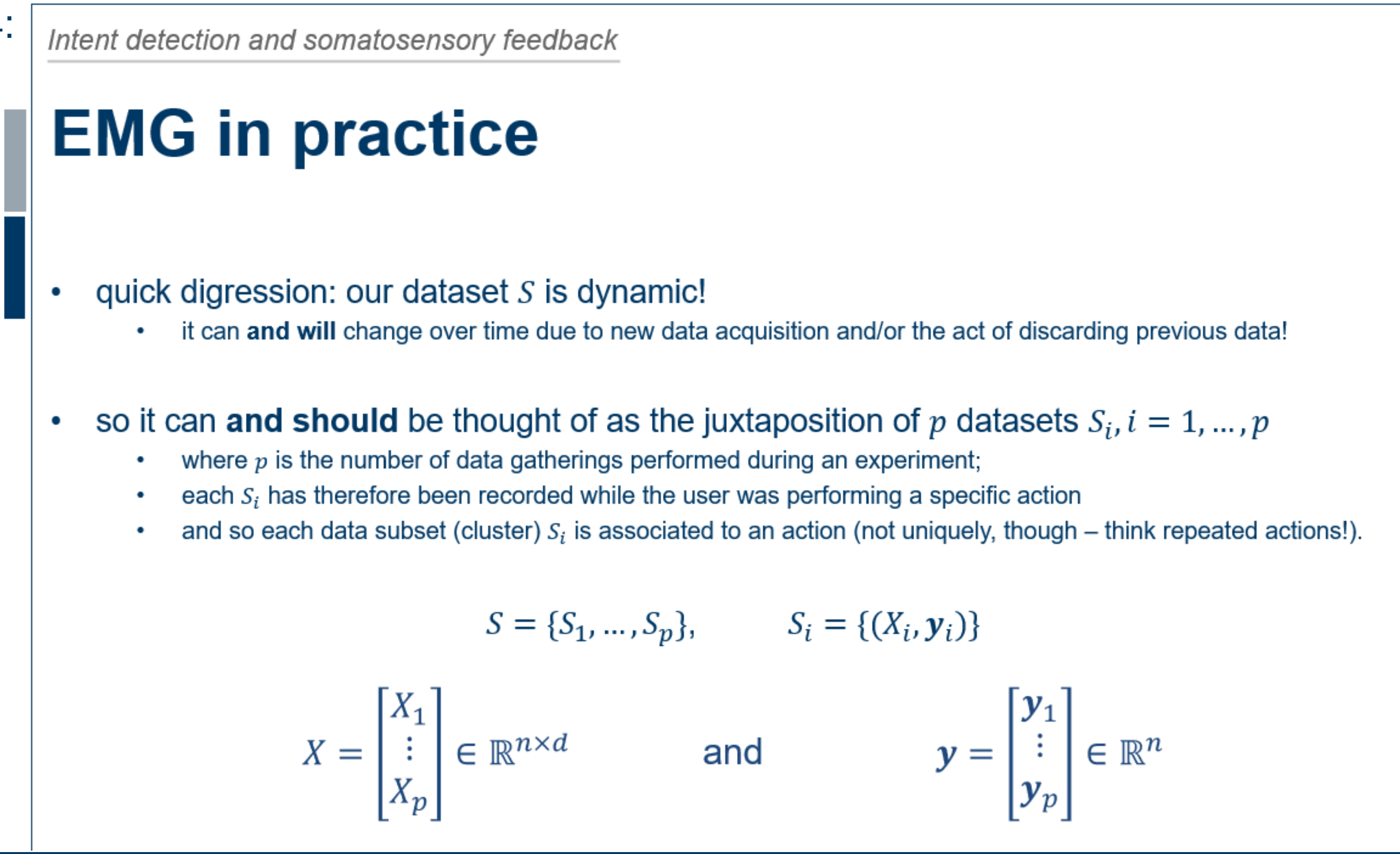
  $X = \begin{bmatrix} \boldsymbol{x}_1^T \\ \vdots \\ \boldsymbol{x}_n^T \end{bmatrix} \in \mathbb{R}^{n \times d}$ and $\boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^n$

- in practice, juxtaposing each observation in an $n \times d$-element matrix and each target value in a n-dimensional vector.

- target values can be either real numbers or labels; altogether they are called *ground truth*.
  - ground truth is… the supposedly *true* value associated to an observation, i.e.,
  - the value the function we try to approximate would yield, given the associated observation.

Machine learning and I.D.

- recall Lecture #4:



EMG in practice

- quick digression: our dataset S is dynamic!
  - it can **and will** change over time due to new data acquisition and/or the act of discarding previous data!
- so it can **and should** be thought of as the juxtaposition of p datasets $S_i, i = 1, \dots, p$
  - where p is the number of data gatherings performed during an experiment;
  - each S_i has therefore been recorded while the user was performing a specific action
  - and so each data subset (cluster) S_i is associated to an action (not uniquely, though – think repeated actions!).

$$S = \{S_1, \dots, S_p\}, \qquad S_i = \{(X_i, \boldsymbol{y}_i)\}$$

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_p \end{bmatrix} \in \mathbb{R}^{n \times d} \qquad \text{and} \qquad \boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_p \end{bmatrix} \in \mathbb{R}^n$$

Ground truth

- difficult to gather it, but we already covered the issue:
  - rather than using sensors to recisely determine it,
  - we induce the participant to „try and perform an action“
  - we assume „synthetic“ ground truth value(s) and artificially associate it to the current signals.
- so let us suppose that, for each X_i we know a suitable y_i
- y_i is an „action“, but *what is it really?*
- …it depends on „what your devices wants to receive“

Ground truth

- it can be a label (output of a *classifier*), e.g.,
  - for a prosthetic hand+wrist, $y_i \in \{$*power_grasp*, *flex_wrist*, *abduct_thumb*$\}$
  - for a prosthetic leg, $y_i \in \{$*walk, ascend_stair, descend_stair, ascend_slope, descend_slope, sit, stand*$\}$
  - for an upper-limb rehabilitation exoskeleton, $y_i \in \{$*reach_in_front, extend_left, extend_right, go_to_rest*$\}$
- it can be a set of motor positions or currents (output of a *regressor*), e.g.,
  - for a prosthetic hand with 6 motors, $\boldsymbol{y}_i \in \mathbb{R}^6$
  - for a lower-limb rehabilitation exoskeleton with 4 motors, $\boldsymbol{y}_i \in \mathbb{R}^4$

  (recall: an m-dimensonal output is just like m independent machines running in parallel)

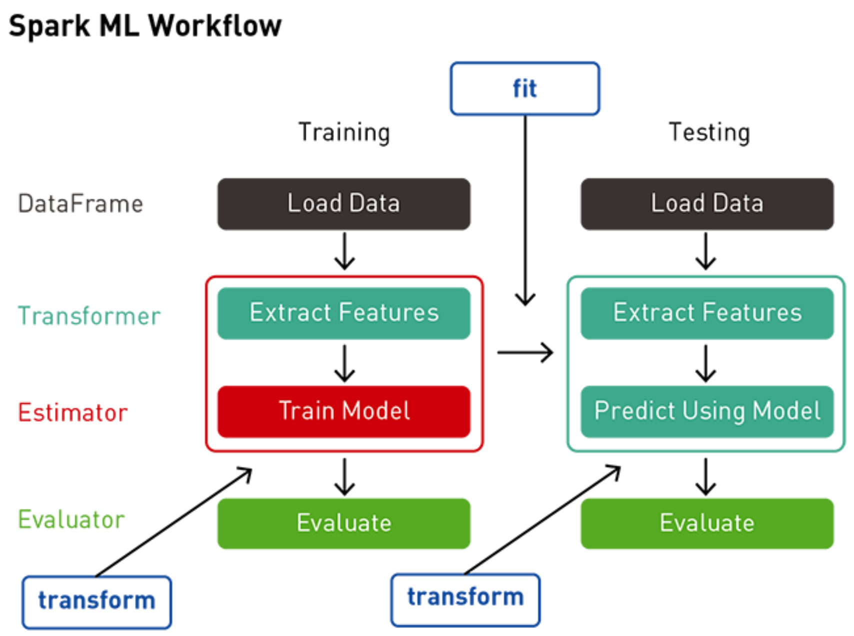
  The output at each time point is independent and represents the state of all motors at that specific moment.
- it can be a set of trajectories, e.g., sequences of positions, e.g.,
  - for a prosthetic hand with 6 motors, $\boldsymbol{y}_i(t) \in \mathbb{R}^{6\times T}$
  - The output is a continuous sequence of positions over time, showing how each motor's position changes.
- it can be a position in Cartesian space, e.g.,
  - for an upper-limb rehabilitation exoskeleton, $\boldsymbol{y}_i \in \mathbb{R}^3$
  - „reach that specific position up there on the left-hand side“

Ground truth

- difficult to gather it, but we already covered the issue:
    - rather than using sensors to precisely determine it,
    - we induce the participant to „try and perform an action“
    - we assume „synthetic“ ground truth value(s) and artificially associate it to the current signals.
- so let us suppose that, for each X_i we know a suitable y_i
- $\boldsymbol{y}_i$ is an „action“, but *what is it really?*
- **…it depends on „what your devices wants to receive“**
- **and one needs to take care how the device enforces it.**

The typical *machine learning pipeline*

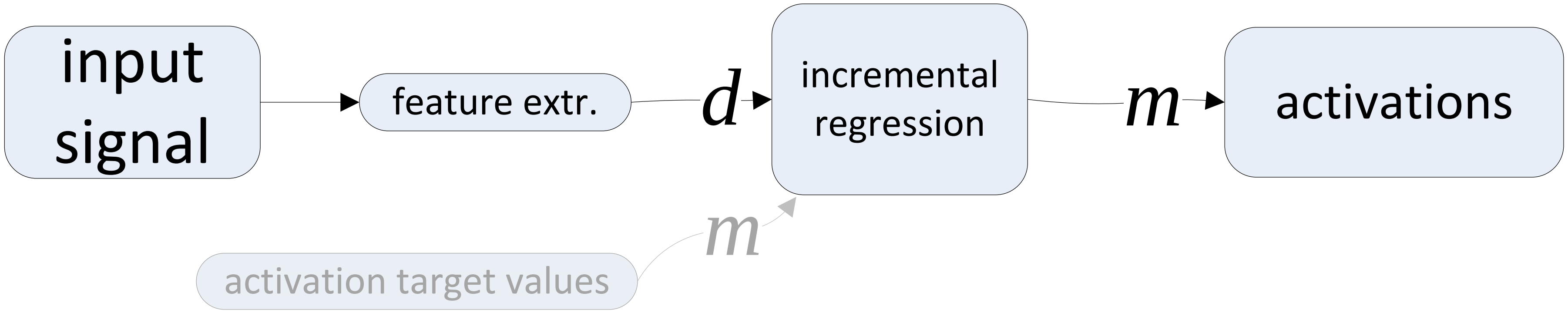
- „training“ phase: load data, process it, build model, evaluate
- testing phase: load data, process it, predict, evaluate



(by *model* we simply mean the *function* f.)

Our *non*-typical *machine learning pipeline*

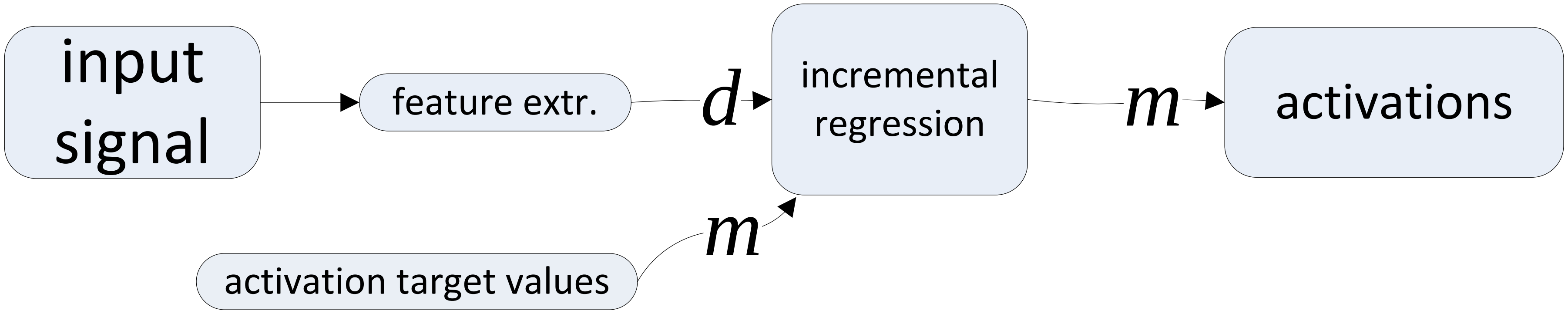
```
10.    start with an empty model
20.    load new data, process it, update model
30.    load new data, process it, predict, evaluate
40.    goto 20
```



Our *non*-typical *machine learning pipeline*

Starts with an empty model and updates it iteratively with new data

```
10.    start with an empty model
20.    load new data, process it, update model
30.    load new data, process it, predict, evaluate
40.    goto 20
```

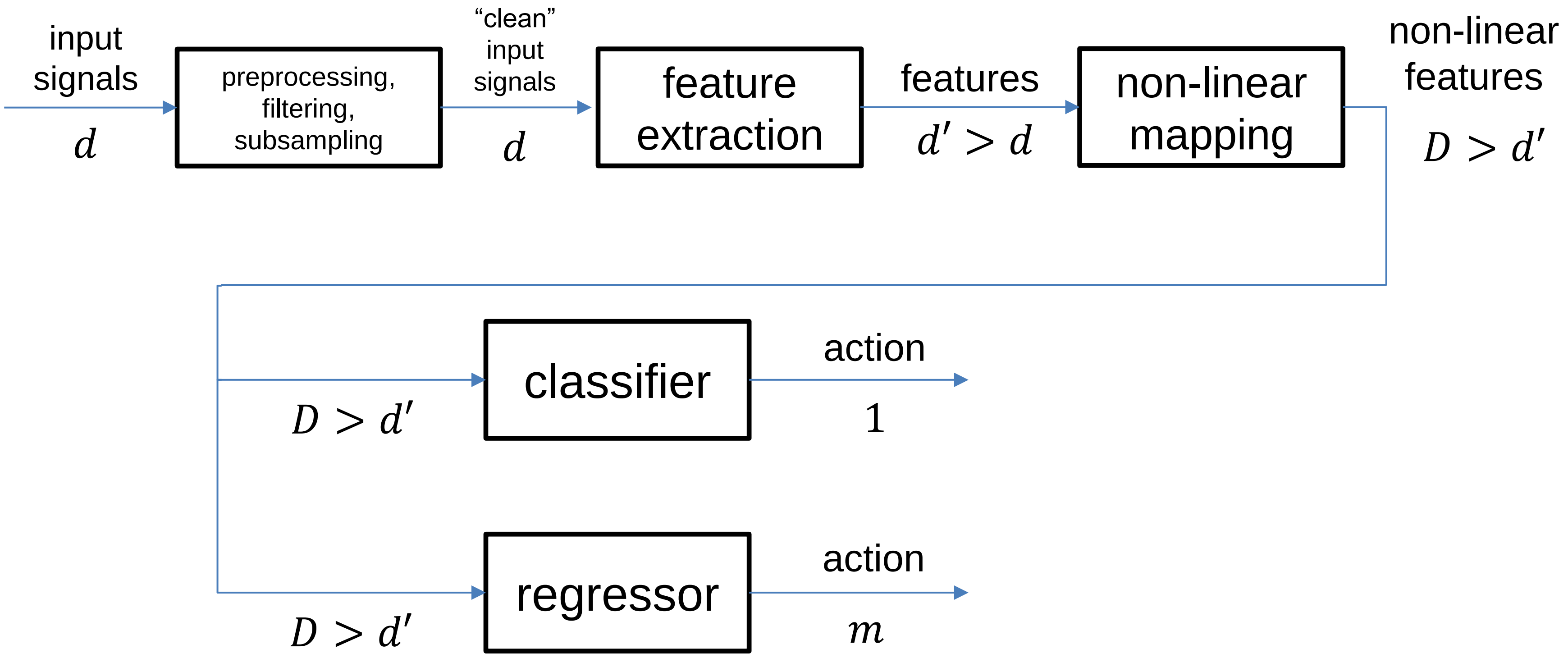


Activation target value is the training part

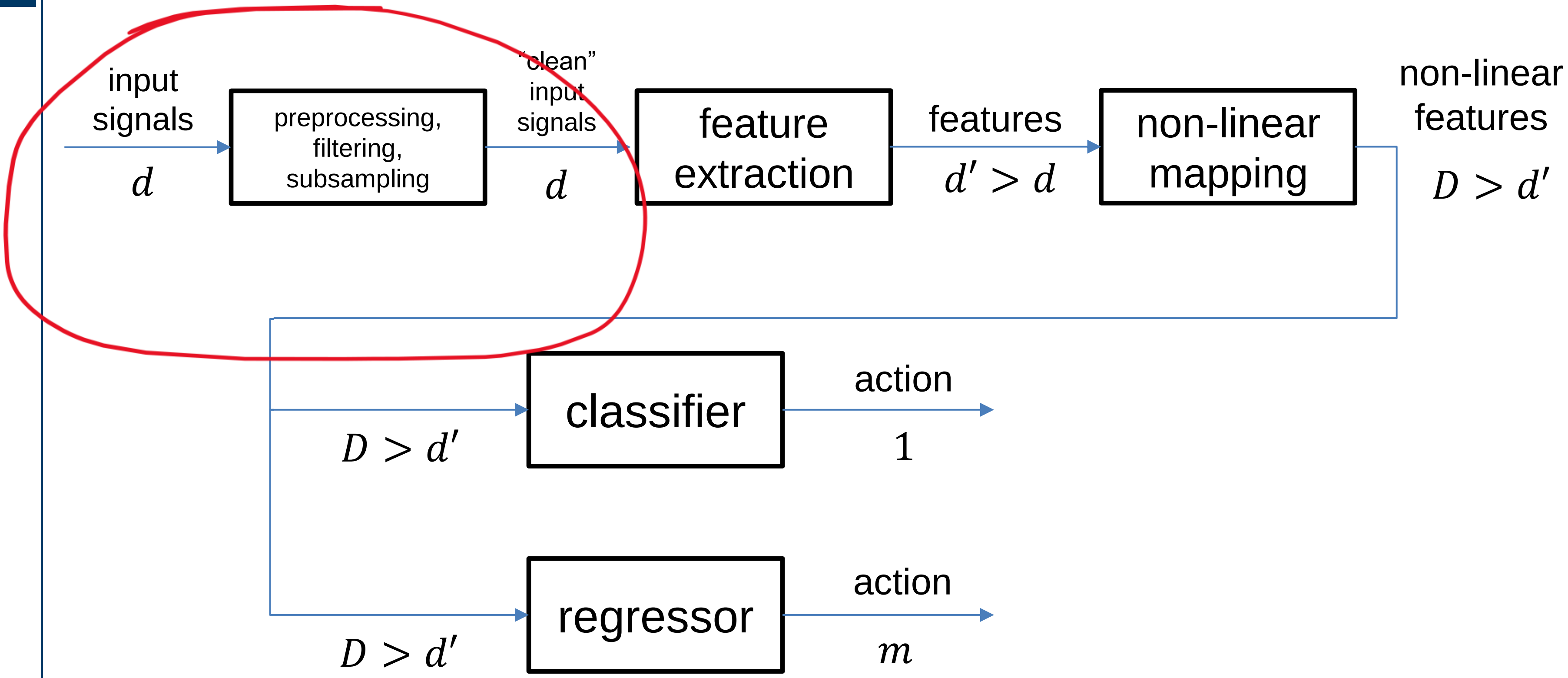
Intent Detection, again

- now, what is f supposed to be, really?
- literally anything turning signals into actions.
- traditionally, in practice, a chain of
  - a preprocessing / filtering / subsampling stage - Filtering is used to remove noise, and subsampling reduces the data size without losing significant information.
  - a feature extraction stage
  - a non-linear mapping
  - a classifier or a regression machine

The Intent Detection pipeline



The Intent Detection pipeline



John L. Semmlow, Benjamin Griffel, *Biosignal and Medical Image Processing (3rd Edition),* ISBN 9781466567368, CRC Press, 2014.

Filtering and subsampling

- aim: to keep **all** and **only** the meaningful part of the signal
  - filter out whatever carries no useful information (noise)
  - subsample as much as possible - Subsampling reduces the number of samples in the signal to decrease the amount of data for processing.

- sampling your signals:
  - need to consider Nyquist‘s theorem given the signal bandwidth

- take care that the bandwidth does not exceed
  - your computational power
  - your wireless bandwidth

- filtering: eliminating frequency components which carry no information (noise)
  - bandpass filtering - Allows frequencies within a certain range to pass through and attenuates frequencies outside this range.
  - lowpass filtering - Allows frequencies below a certain cutoff to pass through and attenuates higher frequencies.

- subsampling: only retain 1 in N samples, N given by the bandwidth *after* filtering
  - *never subsample before filtering – risk of aliasing!*

Intent detection and somatosensory feedback

1. When a signal is sampled below the Nyquist rate, different frequency components of the signal become indistinguishable from one another in the sampled data. This phenomenon is called aliasing, where high-frequency components are misrepresented as lower frequencies, leading to distortion.
2. **Computational Power**:
   1. Ensure that the signal processing, including filtering and any subsequent operations, can be handled by the available computational resources. High sample rates and complex filters require more processing power.
3. **Wireless Bandwidth**:
   1. When transmitting data wirelessly, ensure the bandwidth requirements of the signal do not exceed the capacity of the wireless link. Subsampling can help reduce the data rate, making wireless transmission more feasible.

John L. Semmlow, Benjamin Griffel, *Biosignal and Medical Image Processing (3rd Edition),* ISBN 9781466567368, CRC Press, 2014.

Filtering and subsampling

- example: surface EMG from 32 monopolar sensors, resolution: 16bits
  - signal bandwidth: 15-450Hz

- want to extract the ARV signal

1. sample each channel at, say, 1kHz – better 2kHz to stay on the safe side
   *(produces 32*2B*2000s^{-1} = 128kB/s of data)*
2. (digitally) bandpass filter 15-450Hz
   (+possibly notch at 50Hz)
3. take absolute value (rectification)
4. lowpass filter 5Hz
5. subsample at 10Hz, that is, take one sample in 200
   *(ends up with 32*2B*10s^{-1} = 640B/s of data)*

John L. Semmlow, Benjamin Griffel, *Biosignal and Medical Image Processing (3rd Edition),* ISBN 9781466567368, CRC Press, 2014.

Filtering and subsampling

- beware of the delays you introduce in the pipeline
  - moving averages
  - buffering
  - low-pass filtering
  - …
- beware of irregularities in sampling rate, quantization, etc.
  - how often do you read/store your data?
  - Windows is no RT system! (and there‘s very few RT OSs around, actually)
  - the SDK that comes with the device needs be *fully understood…*
  - *…as well as the transmission protocol!*

John L. Semmlow, Benjamin Griffel, *Biosignal and Medical Image Processing (3rd Edition),* ISBN 9781466567368, CRC Press, 2014.

Filtering and subsampling

- fundamental issue: how do you synchronise signals coming from multiple sources?

- we need to do that!, e.g.,
  - to harmonise the input and output space,- Ensuring that the data streams from different sensors can be used together meaningfully.
  - to determine the adherence to the experimental protocol, - Making sure that the data collected adheres to the timing and sequence required by the experiment.
  - to correctly enforce data fusion, - Correctly combining data from different sources into a single, unified dataset
  - …

- example:
  - sEMG sampled at 2kHz,
  - IMU data smapled at 75Hz,
  - visual stimulus sampled at 25Hz,
  - no synchronization among threads,
  - no way of enforcing RT loops.

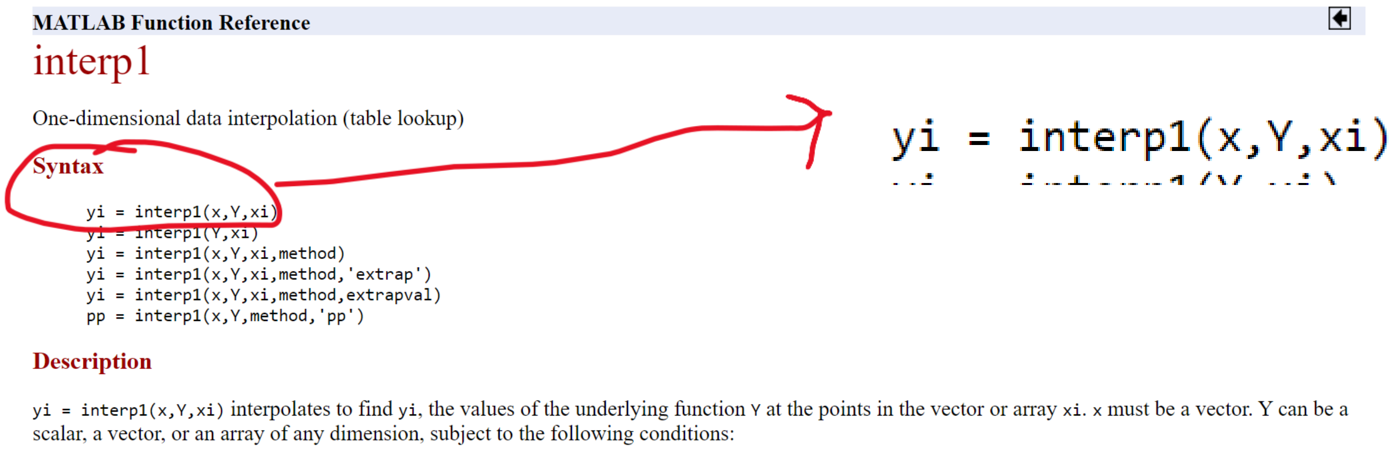
- how do you get to a coherent and usable flow of data?

John L. Semmlow, Benjamin Griffel, *Biosignal and Medical Image Processing (3rd Edition),* ISBN 9781466567368, CRC Press, 2014.

Filtering and subsampling

- fundamental issue: how do you synchronise signals coming from multiple sources?
- simple procedure:
    - obtain a precise timestamp for each sample, - Each sample from each sensor should have an associated timestamp
    - check that the sampling rates make sense,- Verify that the sampling rates are as expected and consistent throughout the data collection process.
    - *interpolate* signals over the fastest signal‘s timestamps.-Choose the highest sampling rate as the reference. Here, it’s the sEMG signal sampled at 2 kHz. Interpolate the IMU and visual stimulus data to match the timestamps of the sEMG data.

- consider, e.g., Matlab‘s `interp1` function:



MATLAB Function Reference

interp1

One-dimensional data interpolation (table lookup)

Syntax

```
yi = interp1(x,Y,xi)
yi = interp1(Y,xi)
yi = interp1(x,Y,xi,method)
yi = interp1(x,Y,xi,method,'extrap')
yi = interp1(x,Y,xi,method,extrapval)
pp = interp1(x,Y,method,'pp')
```

```
yi = interp1(x,Y,xi)
```

Description

`yi = interp1(x,Y,xi)` interpolates to find `yi`, the values of the underlying function `Y` at the points in the vector or array `xi`. `x` must be a vector. Y can be a scalar, a vector, or an array of any dimension, subject to the following conditions:

Intent detection and somatosensory feedback

1. Interpolation is often used in signal processing and data synchronization, especially when integrating data from multiple sources sampled at different rates. Interpolation is a method used to estimate unknown values that fall between known values in a dataset. It is commonly used in data analysis to predict missing data points or to convert data sampled at different rates to a common timeline.
2. Synchronization in the context of data processing refers to aligning datasets or signals in time or space to ensure that they are comparable or can be used together effectively. When working with multiple datasets that were recorded at different times or with different sampling rates, it is crucial to synchronize them so that the data points from each set correspond accurately to the same moments in time or positions. in space.
3. If you have datasets recorded at different times or intervals, you may need to interpolate one dataset so that its time points match another dataset.

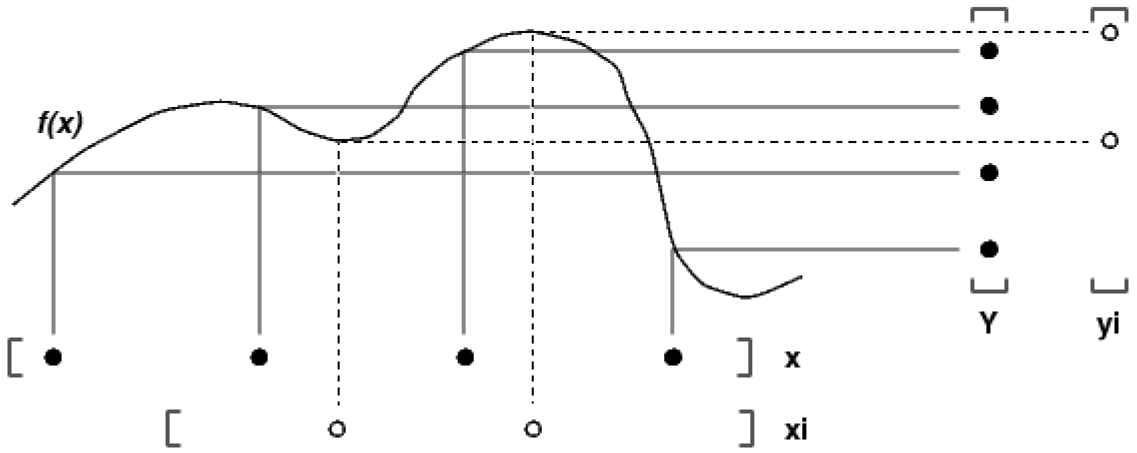
John L. Semmlow, Benjamin Griffel, *Biosignal and Medical Image Processing (3rd Edition),* ISBN 9781466567368, CRC Press, 2014.

Filtering and subsampling

- fundamental issue: how do you synchronise signals coming from multiple sources?
- simple procedure:
  - obtain a precise timestamp for each sample,
  - check that the sampling rates make sense,
  - *interpolate* signals over the fastest signal‘s timestamps.

```
yi = interp1(x,Y,xi)
```

- consider, e.g., matlab‘s `interp1` function:



John L. Semmlow, Benjamin Griffel, *Biosignal and Medical Image Processing (3rd Edition),* ISBN 9781466567368, CRC Press, 2014.

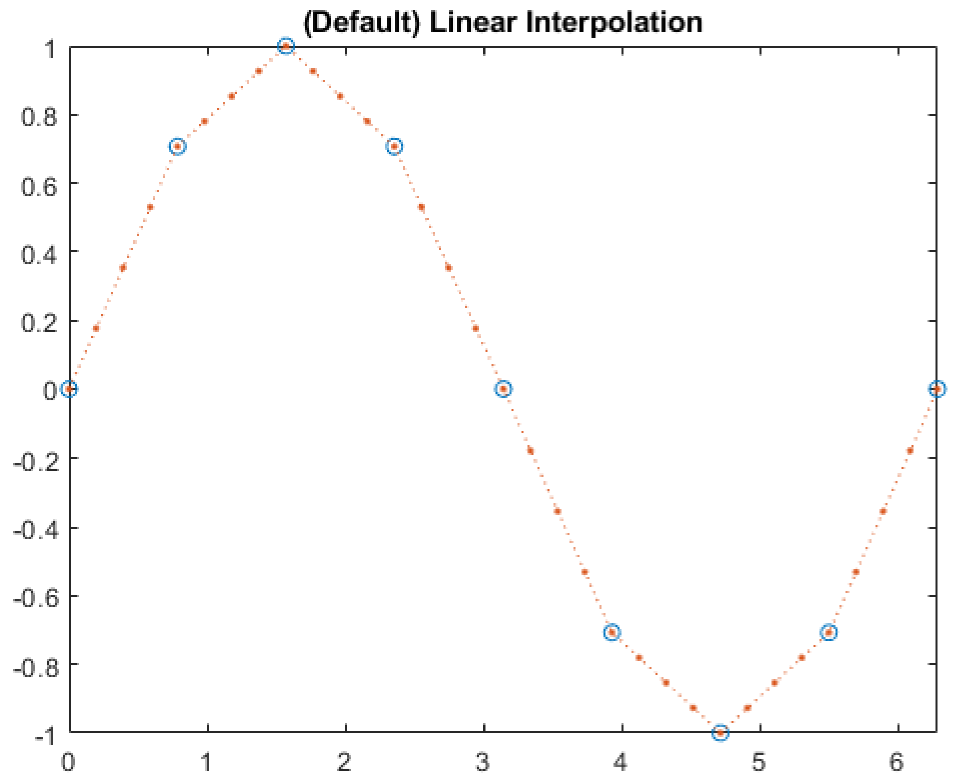
Filtering and subsampling

- fundamental issue: how do you synchronise signals coming from multiple sources?

- simple procedure:
  - obtain a precise timestamp for each s
  - check that the sampling rates make s
  - *interpolate* signals over the fastest sig

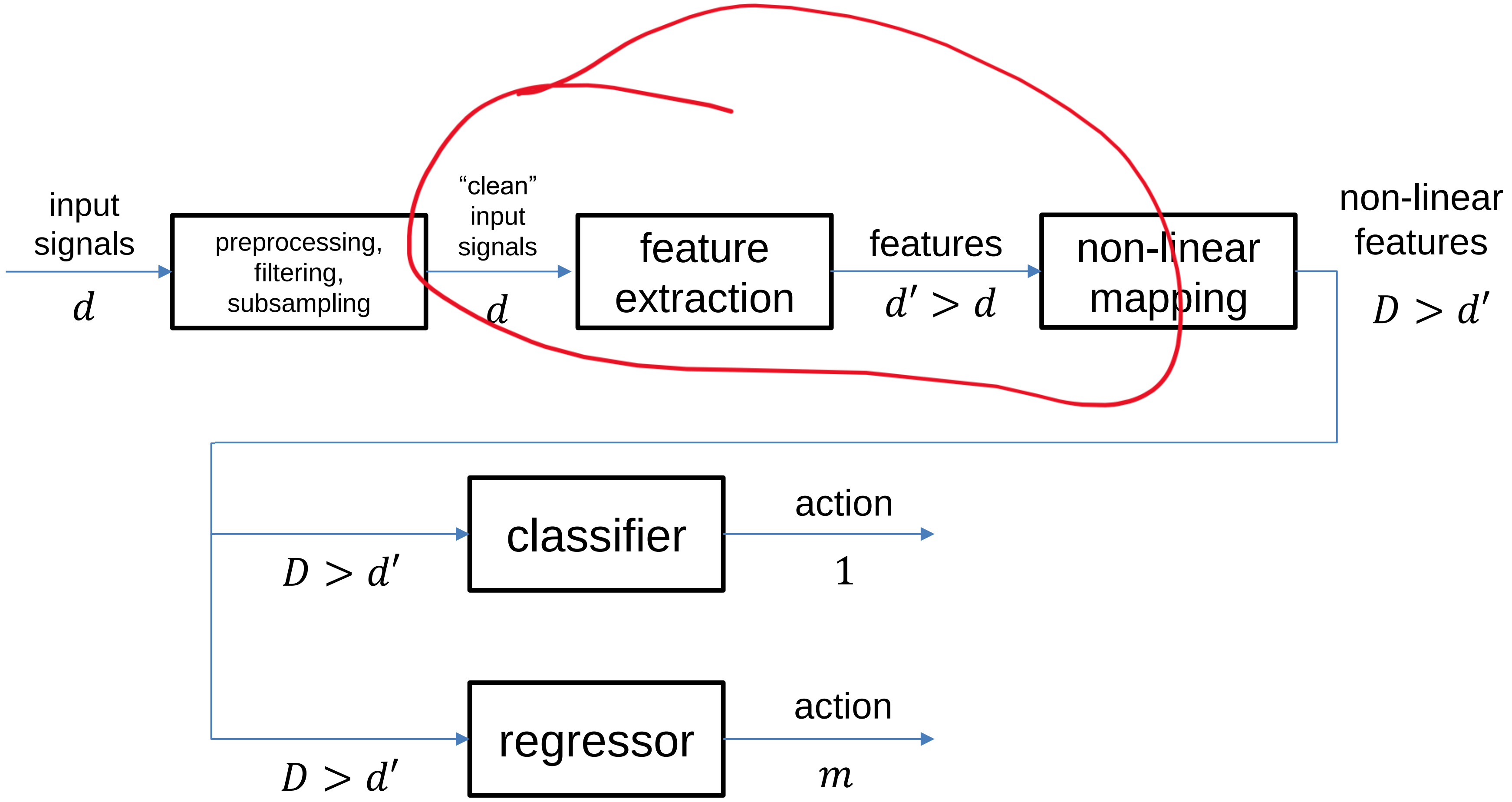
- consider, e.g., matlab‘s `interp1`

```
yi = interp1(x,Y,xi)
```



John L. Semmlow, Benjamin Griffel, *Biosignal and Medical Image Processing (3rd Edition),* ISBN 9781466567368, **CRC Press,** 2014.

The Intent Detection pipeline



Features

- „feature extraction“ is an extremely vague term
  - in general, the „feature space“ will have more dimensions than the signal‘s ($d‘ > d$),
  - but this needs not necessarily be the case.
  - unfortunately „feature space“ in the ML lingo means something else

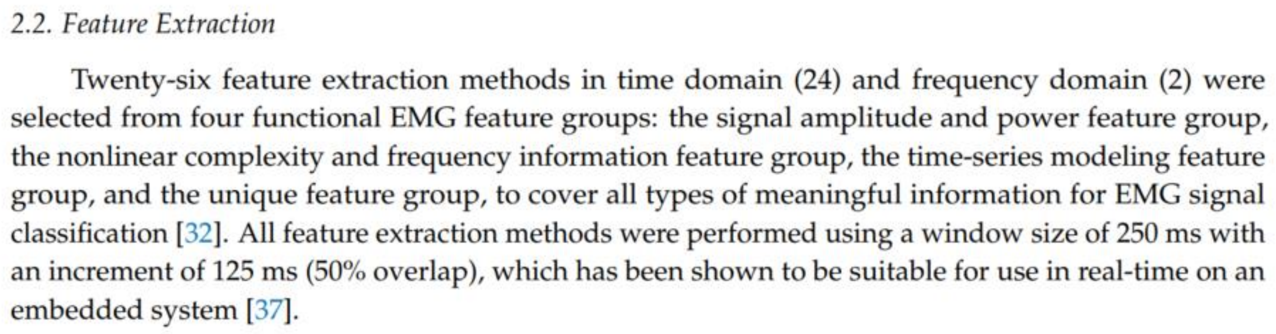
- the features you extract are supposed to
  - represent „meaningful“ characteristics of the signals,
  - typical of the activity of the participant,
  - interesting to monitor and check,
  - useful to deduce something about her/his status (the intent? the certainty of an action?)

- usually:
  - characteristics of the time domain – features of the signal that changes with time – eg amplitude
  - characteristics of the frequency domain eg frequency spectrum
  - algebraic operations, i.e., combining and matrix-multiplying them – will also give new features

Features

- examples:
- a simple one: adding the „effort“ to your signals
    - use the norm of the ARV values: $e = \sqrt{e_1^2 + \cdots + e_d^2}$
    - juxtapose this value to the ARV values: $x = [e_1 \ldots e_d \; e]$
    - The norm of ARV values provides a single scalar representing the overall strength of the signal across multiple dimensions or channels.

- a more complex one, this time starting from the raw EMG signal:



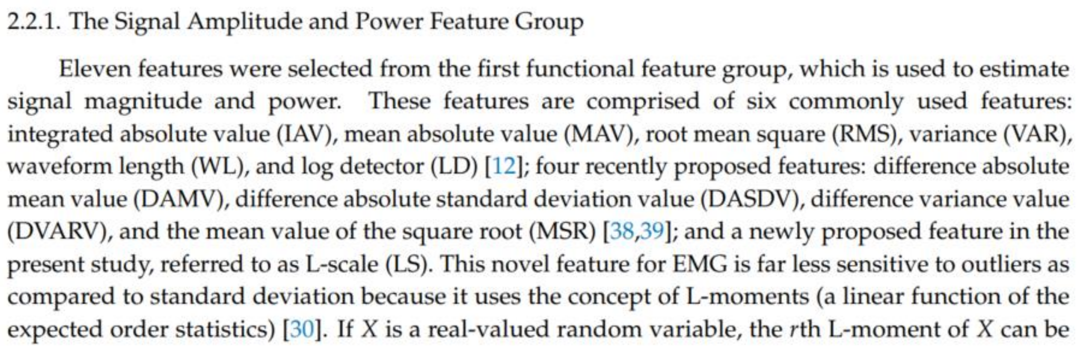
2.2. Feature Extraction

Twenty-six feature extraction methods in time domain (24) and frequency domain (2) were selected from four functional EMG feature groups: the signal amplitude and power feature group, the nonlinear complexity and frequency information feature group, the time-series modeling feature group, and the unique feature group, to cover all types of meaningful information for EMG signal classification [32]. All feature extraction methods were performed using a window size of 250 ms with an increment of 125 ms (50% overlap), which has been shown to be suitable for use in real-time on an embedded system [37].

Angkoon Phinyomark, Rami N. Khushaba and Erik Scheme,
Feature Extraction and Selection for Myoelectric Control Based on Wearable EMG Sensors,
MDPI Sensors 2018, 18, 1615

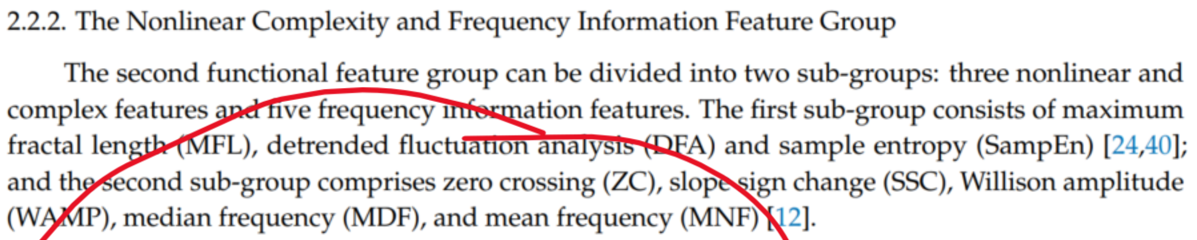
Features

- a more complex one, this time starting from the raw EMG signal:



2.2.1. The Signal Amplitude and Power Feature Group

Eleven features were selected from the first functional feature group, which is used to estimate signal magnitude and power. These features are comprised of six commonly used features: integrated absolute value (IAV), mean absolute value (MAV), root mean square (RMS), variance (VAR), waveform length (WL), and log detector (LD) [12]; four recently proposed features: difference absolute mean value (DAMV), difference absolute standard deviation value (DASDV), difference variance value (DVARV), and the mean value of the square root (MSR) [38,39]; and a newly proposed feature in the present study, referred to as L-scale (LS). This novel feature for EMG is far less sensitive to outliers as compared to standard deviation because it uses the concept of L-moments (a linear function of the expected order statistics) [30]. If X is a real-valued random variable, the rth L-moment of X can be



2.2.2. The Nonlinear Complexity and Frequency Information Feature Group

The second functional feature group can be divided into two sub-groups: three nonlinear and complex features and five frequency information features. The first sub-group consists of maximum fractal length (MFL), detrended fluctuation analysis (DFA) and sample entropy (SampEn) [24,40]; and the second sub-group comprises zero crossing (ZC), slope sign change (SSC), Willison amplitude (WAMP), median frequency (MDF), and mean frequency (MNF) [12].

Angkoon Phinyomark, Rami N. Khushaba and Erik Scheme,
Feature Extraction and Selection for Myoelectric Control Based on Wearable EMG Sensors,
MDPI Sensors 2018, 18, 1615

Features

- a more complex one, this time starting from the raw EMG signal: Zero crossing

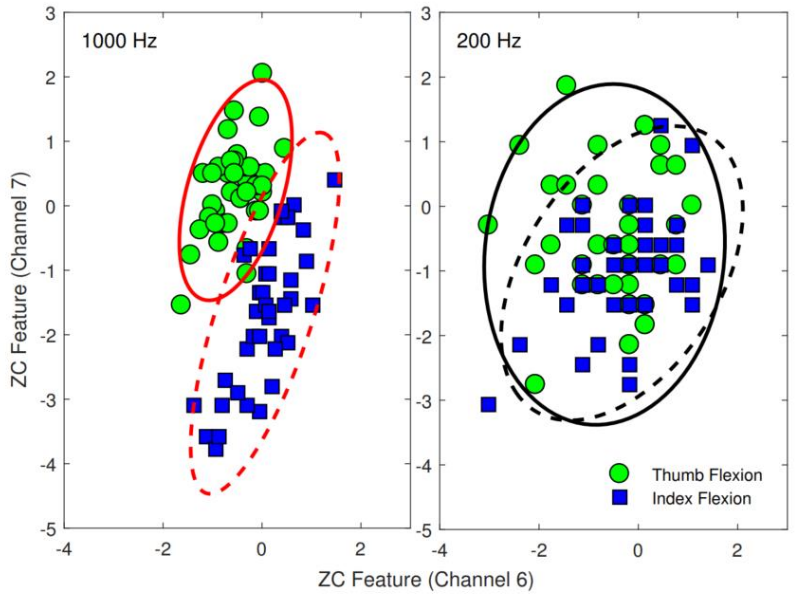


Figure 2. Differences in EMG patterns between using: (**left**) a 1000 Hz sampling rate; and (**right**) a 200 Hz sampling rate. ZC features are extracted from two different EMG channels (6 and 7) during thumb flexion (green circle markers and solid lines) and index flexion (blue square markers and dashed lines). Samples are from Subject 1 of Database 3.

Zero Crossing (ZC): The number of times the EMG signal crosses the zero line, indicating frequency content.

Angkoon Phinyomark, Rami N. Khushaba and Erik Scheme,
Feature Extraction and Selection for Myoelectric Control Based on Wearable EMG Sensors,
MDPI Sensors 2018, 18, 1615

Features - The slide illustrates the classification performance of various feature sets extracted from raw EMG signals

- a more complex one, this time starting from the raw EMG signal:

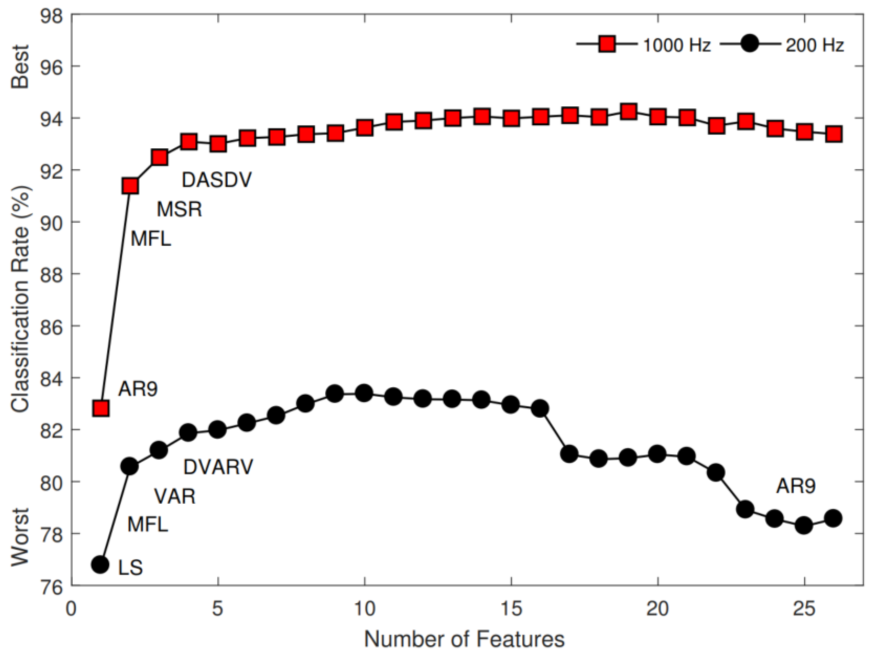


Figure 3. The classification performance of the multi-feature sets sequentially selected by the SFS method using the two different sampling rates (1000 Hz vs. 200 Hz) for Dataset 3.

After some point, adding more features can lead to drop in the performance
Window Length: This is the duration of the signal segment used for feature extraction.
For instance, a 100 ms window means that features are extracted from 100 milliseconds of the signal.
Divide the whole signal to small windows to extract features

Angkoon Phinyomark, Rami N. Khushaba and Erik Scheme,
:ction for Myoelectric Control Based on Wearable EMG Sensors,
MDPI Sensors 2018, 18, 1615

Features

- more examples... (is this of *any* use?)

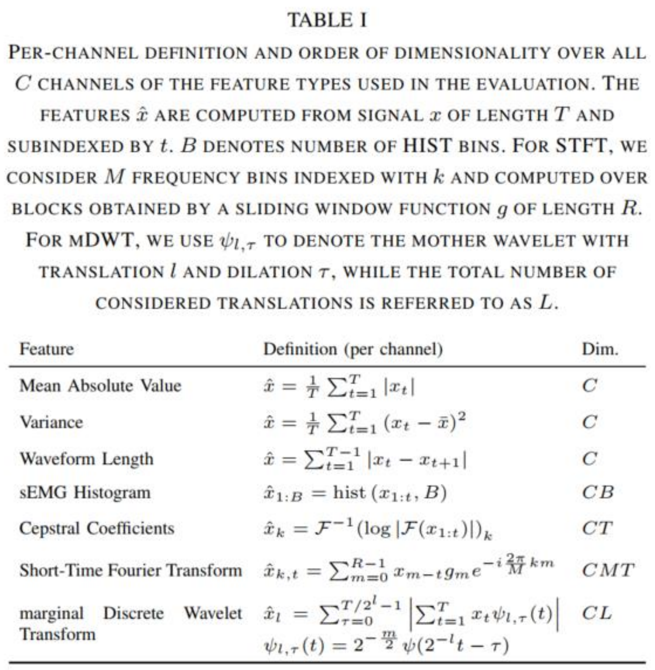
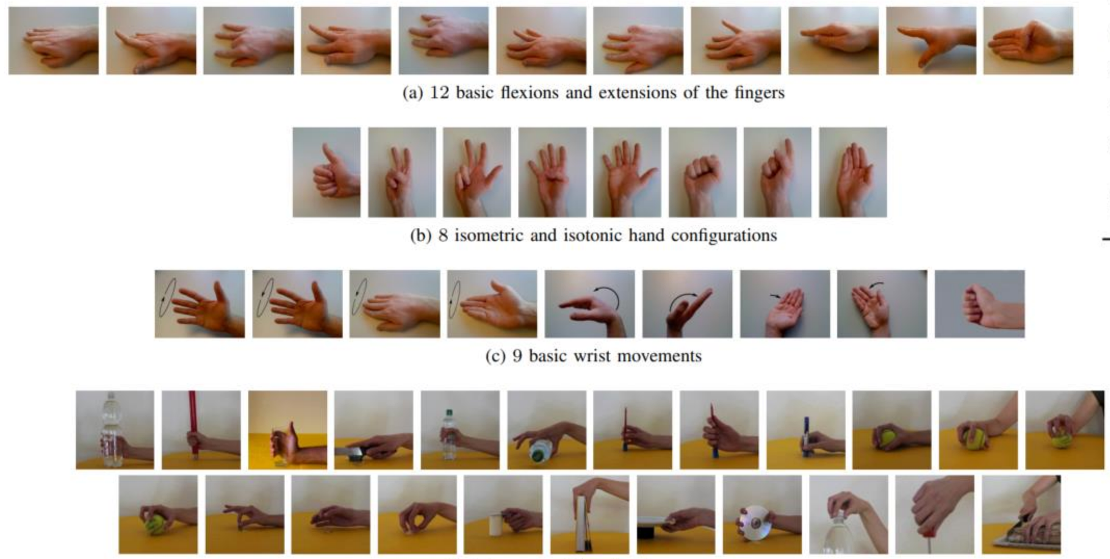


TABLE I

PER-CHANNEL DEFINITION AND ORDER OF DIMENSIONALITY OVER ALL C CHANNELS OF THE FEATURE TYPES USED IN THE EVALUATION. THE FEATURES $\hat{x}$ ARE COMPUTED FROM SIGNAL x OF LENGTH T AND SUBINDEXED BY t. B DENOTES NUMBER OF HIST BINS. FOR STFT, WE CONSIDER M FREQUENCY BINS INDEXED WITH k AND COMPUTED OVER BLOCKS OBTAINED BY A SLIDING WINDOW FUNCTION g OF LENGTH R. FOR mDWT, WE USE $\psi_{l,\tau}$ TO DENOTE THE MOTHER WAVELET WITH TRANSLATION l AND DILATION τ, WHILE THE TOTAL NUMBER OF CONSIDERED TRANSLATIONS IS REFERRED TO AS L.

Feature	Definition (per channel)	Dim.
Mean Absolute Value	$\hat{x} = \frac{1}{T}\sum_{t=1}^{T}\|x_t\|$	C
Variance	$\hat{x} = \frac{1}{T}\sum_{t=1}^{T}(x_t - \bar{x})^2$	C
Waveform Length	$\hat{x} = \sum_{t=1}^{T-1}\|x_t - x_{t+1}\|$	C
sEMG Histogram	$\hat{x}_{1:B} = \mathrm{hist}\,(x_{1:t}, B)$	CB
Cepstral Coefficients	$\hat{x}_k = \mathcal{F}^{-1}(\log\|\mathcal{F}(x_{1:t})\|)_k$	CT
Short-Time Fourier Transform	$\hat{x}_{k,t} = \sum_{m=0}^{R-1} x_{m-t}\, g_m\, e^{-i\frac{2\pi}{M}km}$	CMT
marginal Discrete Wavelet Transform	$\hat{x}_l = \sum_{\tau=0}^{T/2^l-1}\left\|\sum_{t=1}^{T} x_t \psi_{l,\tau}(t)\right\|$ $\psi_{l,\tau}(t) = 2^{-\frac{m}{2}}\psi(2^{-l}t - \tau)$	CL



(a) 12 basic flexions and extensions of the fingers

(b) 8 isometric and isotonic hand configurations

(c) 9 basic wrist movements

(d) 23 grasp and functional movements

Fig. 1. The 52 movements considered in the NinaPro dataset.

Ilja Kuzborskij, Arjan Gijsberts, and Barbara Caputo, On the Challenge of Classifying 52 Hand Movements from Surface Electromyography, 2012

Features - The slide provides a detailed comparison of classification accuracies for different feature extraction methods and classifiers when applied to EMG signals

- more examples... (is this of *any* use?)

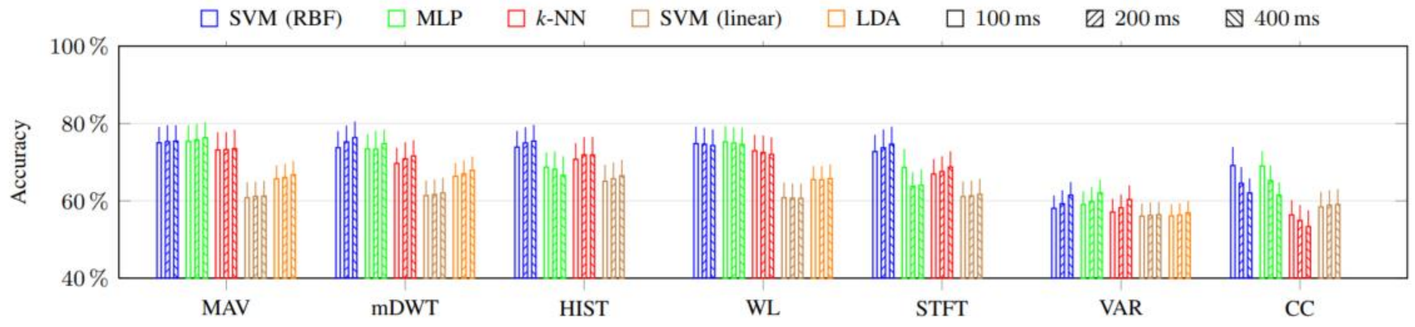


Fig. 2. Classification accuracies. Each bar represents method classification accuracy with respect to feature representation and window length, while line atop the bar is one standard deviation of accuracy. Classifiers are grouped by feature representations and labeled by different colors. Window lengths are represented in increasing order, namely 100ms, 200ms and 400ms and are tagged with different textures. Note, that LDA results are missing in case of STFT, CC and HIST due to reasons described in Section IV.

Ilja Kuzborskij, Arjan Gijsberts, and Barbara Caputo, On the Challenge of Classifying 52 Hand Movements from Surface Electromyography, 2012

Features

- what if there are *too many* „sensors“?
- this is usually the case with *images*, if we consider each pixel a sensor
- in general, what if d is *really* large?

- in this case, you need to exploit the structure of the input space.
  - high-density sEMG readings are images
  - TMG yields images („heatmaps“)
  - EIT yields images (reconstructed inner structure)
  - ultrasound B-mode yields images (as well, reconstructed)

- and images have spatial structure:
  - extract corners, edges, salient points
  - identify RoIs (possibly overlapping with one another)
    and extract fetautes from each one
  - use DL approaches (e.g., CNN)

Features

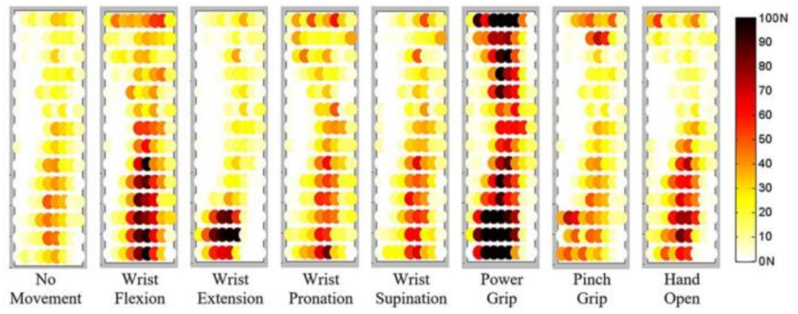
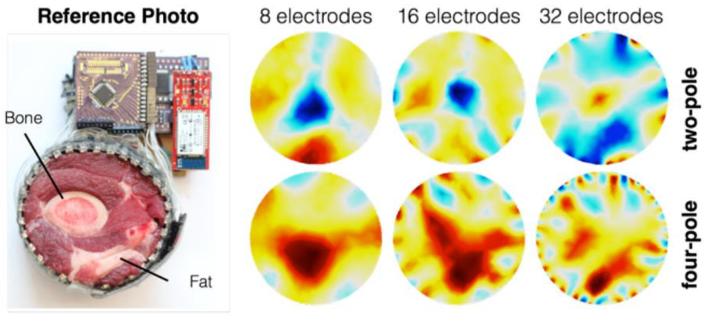


Figure 6.
Examples of acquired pressure map images for the motion classes performed in a fixed static position (darker areas correspond to higher pressure).



-
- we consider each pixel a sensor
- in general, what if d is *really* larg

- in this case, you need to exploit pace.
    - high-density sEMG readings are imag
    - TMG yields images („heatmaps“)
    - EIT yields images (reconstructed inne
    - ll, re

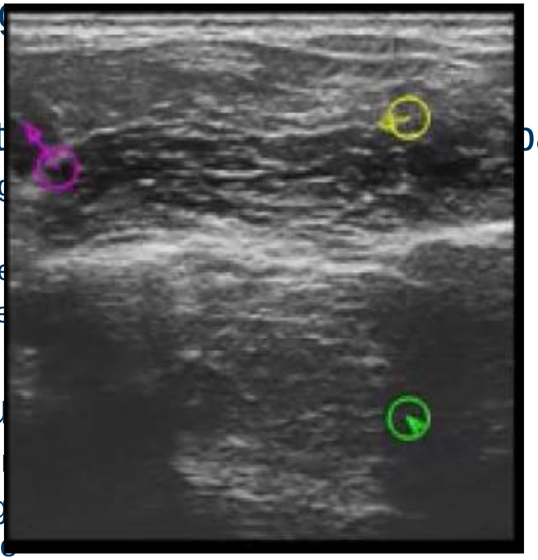
ctu

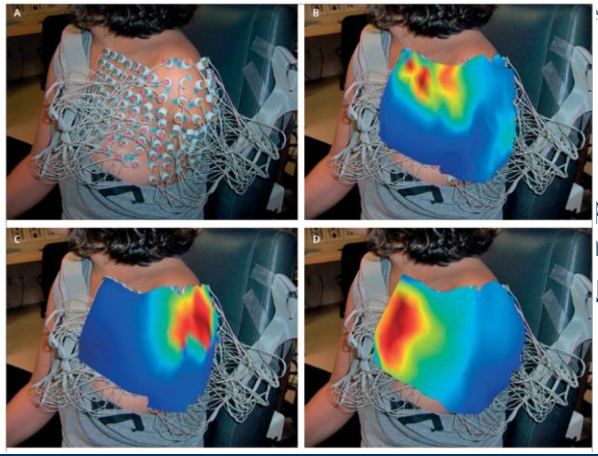
poi

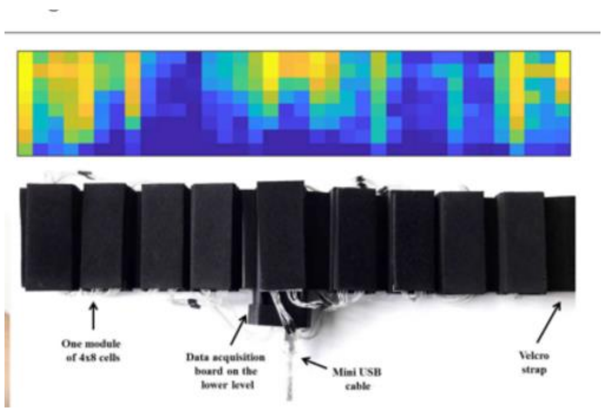
ping

one

l)







Features - This slide discusses a feature extraction method from image data, specifically the linearization of grey values in regions of interest (ROIs) for intent detection and somatosensory feedback

- example: linearisation of grey values in a RoI.
  - build the plane approximating the grey-value distribution in each of the N RoIs
  - each plane is represented by a vector $(\alpha_i, \beta_i, \gamma_i)$
  - end up with $3N$ features
  - started out with $500 * 400$ pixels!

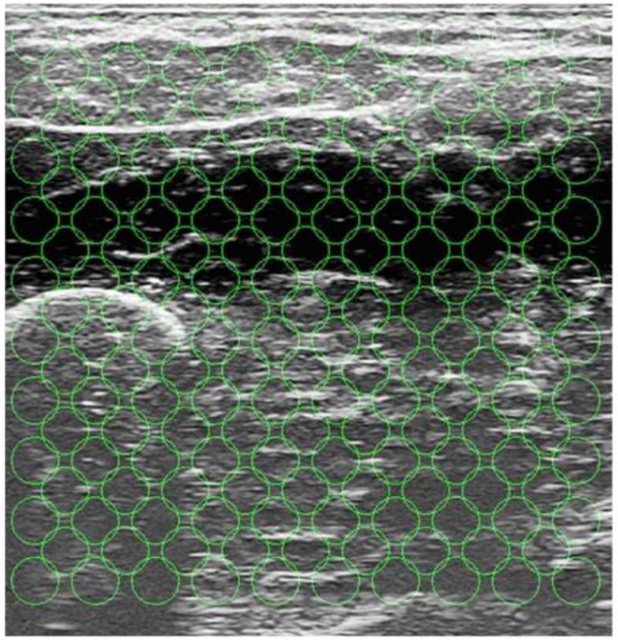
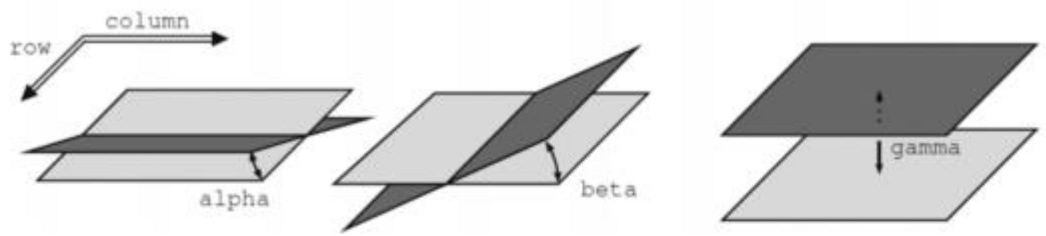


FIGURE 5 | The grid of ROIs, superimposed to the typical shot see Figure 1. Each ROI has a radius of 20 pixels, and each ROI center is 5 pixels apart from each other.



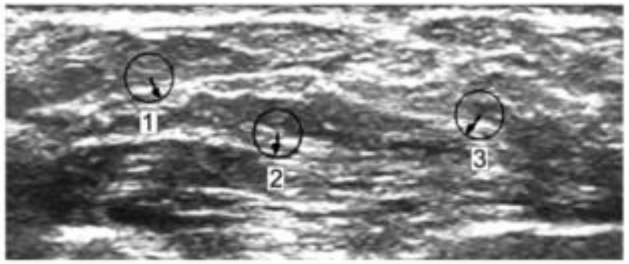


Fig. 4. (Left) A graphical representation of the three features α, β, γ extracted from each interest point. (right) Three ROIs located near the zones of the image where most changes are seen for pinkie flexion and thumb adduction (1,3) and where almost no movement is observed (2). The features are represented as normalized vectors where α and β are the vector components. Consider also the movie "features.avi," included in the supplemental material.

Intent detection and somatosensory feedback

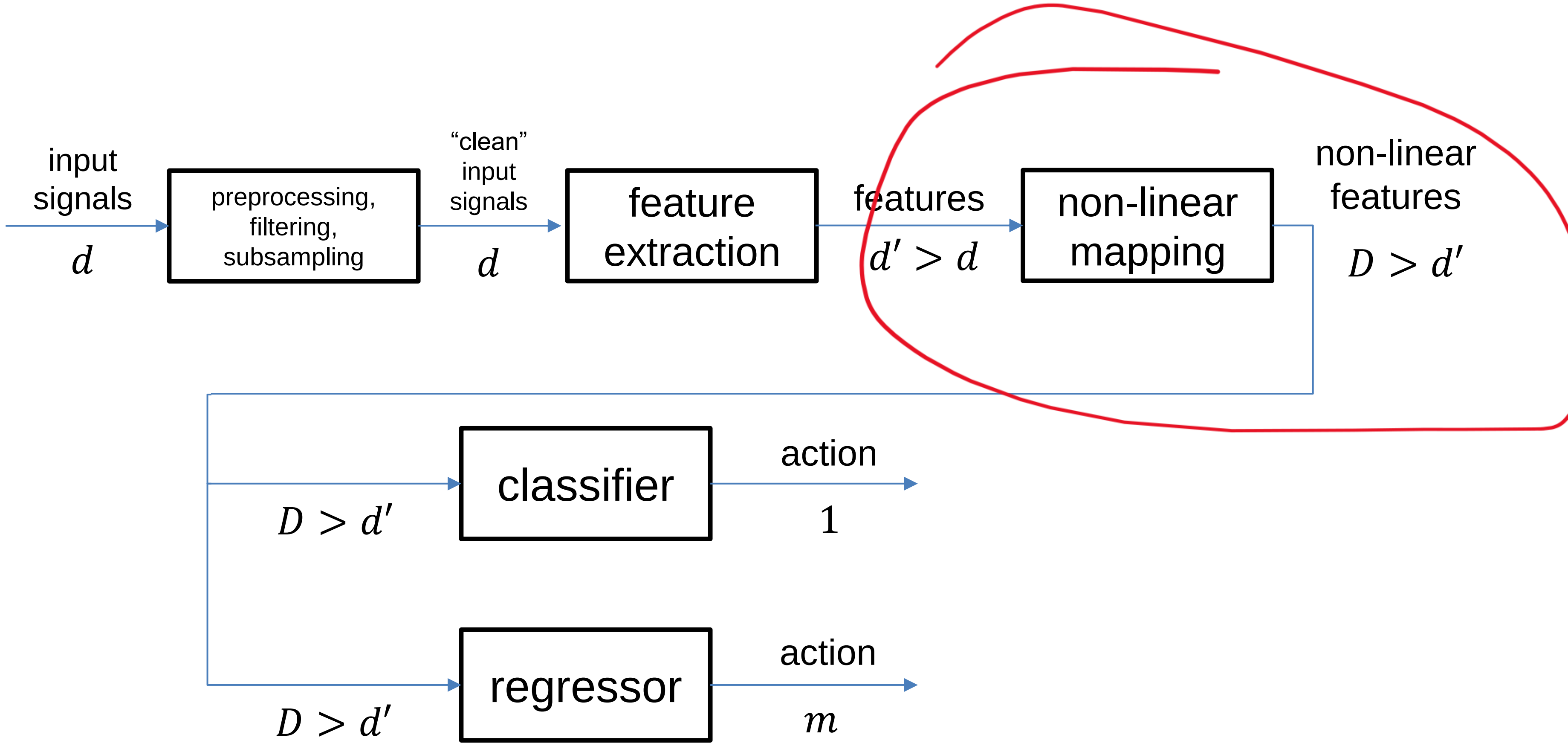
- **Linearization of Grey Values in ROIs**:
- **Objective:** To approximate the grey-value distribution within each ROI using a plane.
- **Method:** Fit a plane to the grey-value data within each ROI.
- This plane can be mathematically represented, and its parameters serve as features.
- **Representation of Each Plane**:
- Each plane is represented by a vector (αi,βi,γi)where:
    - αi: The coefficient representing the gradient along the x-axis (column direction).
    - βi: The coefficient representing the gradient along the y-axis (row direction).
    - γi: The intercept, representing the base grey value when both x and y are zero.
- These coefficients describe the grey-value distribution within the ROI in a compact form.
- **Feature Extraction**:
- The image is divided into N ROIs.
- For each ROI, three features are extracted (the coefficients αi,βi,γi).
- This results in a total of 3N features for the entire image.
- **Initial Image Size**:
- The original image size is 500×400 pixels.
- This indicates a significant reduction in data dimensionality when extracting
- features compared to using raw pixel values.

Castellini, Claudio / Passig, Georg, **Ultrasound image features of the wrist are linearly related to finger positions**, 2011. *Proceedings of IROS - International Conference on Intelligent Robots and Systems*, p. 2108-2114

Features

- example: linearisation of grey values in a RoI.
    - build the plane approximating the grey-value distribution in each of the N RoIs
    - each plane is represented by a vector $(\alpha_i, \beta_i, \gamma_i)$
    - end up with $3N$ features
    - started out with $500 * 400$ pixels!
- feed the features to *linear regression,* it works *perfectly*.
- why does it work? because the *displacement* in the images is proportional to the muscle effort.
- how did we figure that out? *by looking at the live images.*
- take-home lesson: use your sensitivity and feeling.

The Intent Detection pipeline

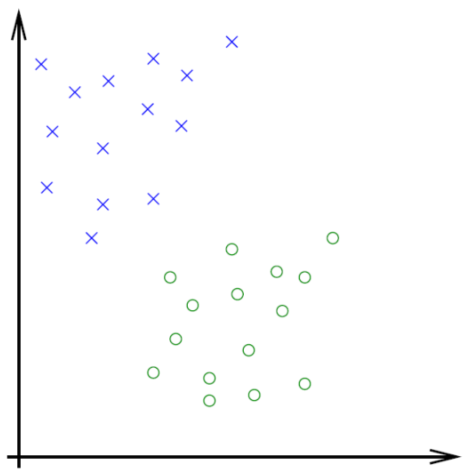


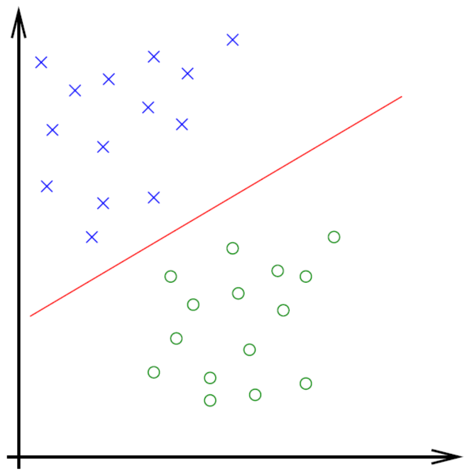
Non-linear mappings

- feature extraction: mapping your (preprocessed) signals onto… something smart
- can we somehow mechanise the procedure?
  - make your algorithm smart by construction?
- remember the lesson we obtain from DL:
  - use a large amount of data to automatically find out salient features!
- what about something more, ahem, mathematical?
- what about making the problem simpler / better conditioned by *expanding* the input space?

Non-linear mappings

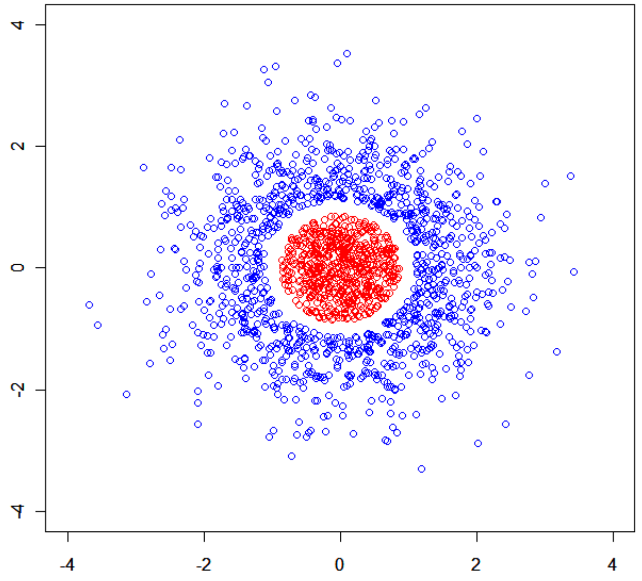
- machine learning prefers to have separable, repeatable patterns (and so do we)
- can samples be better separable if we add some dimensions to their representation?
- here is a simple case: *(linear separability, $d' = 2$)*





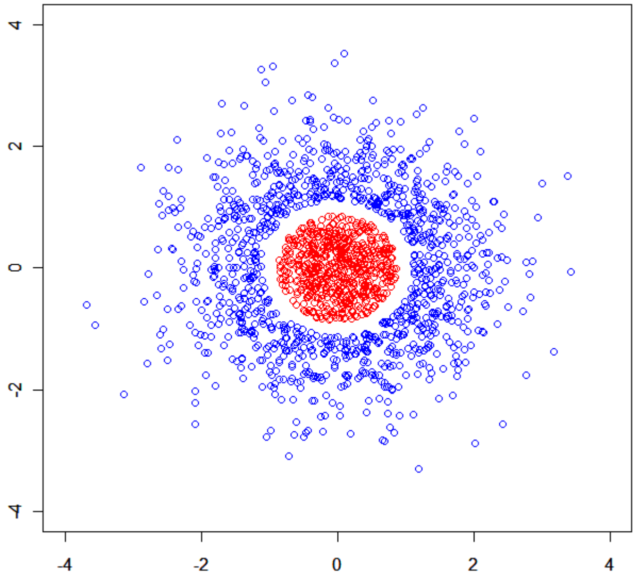
Non-linear mappings

- machine learning prefers to have separable, repeatable patterns (and so do we)
- can samples be better separable if we add some dimensions to their representation?
- here is a more complex one: *(non-linearly separable ,* $d' = 2$*)*



Non-linear mappings

- machine learning prefers to have separable, repeatable patterns (and so do we)
- can samples be better separable if we add some dimensions to their representation?
- here is a more complex one: *(non-linearly separable , $d' = 2, D = 3$)*



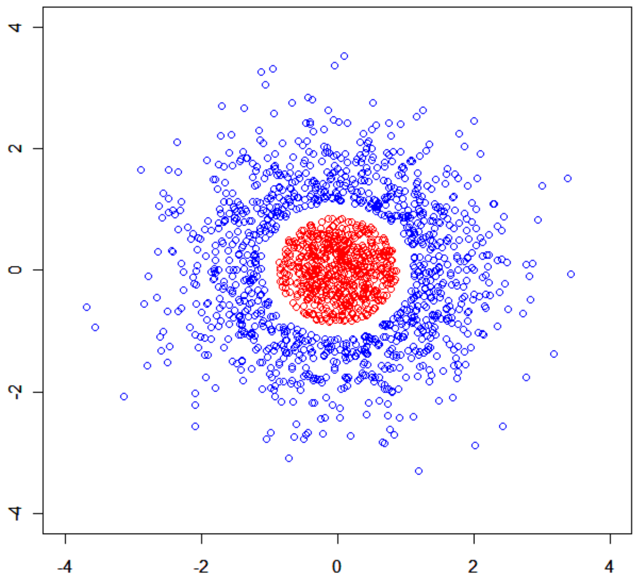
$$\phi(x_1, x_2) = (x_1, x_2, -\|\boldsymbol{x}\|)$$

$\Longrightarrow$

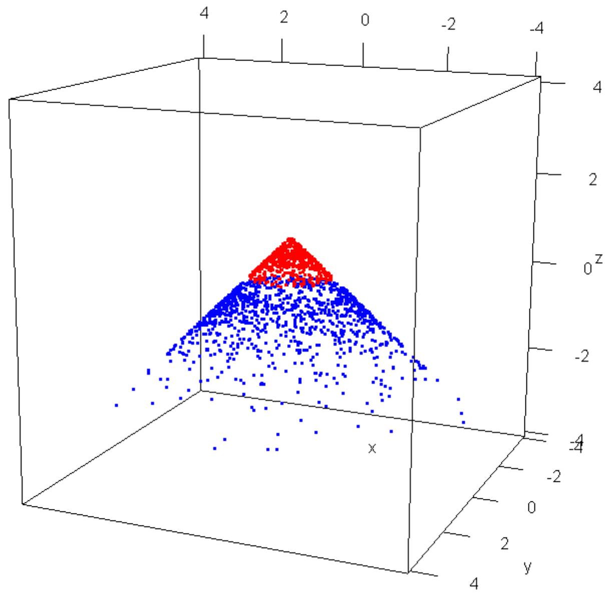
?

Non-linear mappings

- machine learning prefers to have separable, repeatable patterns (and so do we)
- can samples be better separable if we add some dimensions to their representation?
- here is a more complex one: *(non-linearly separable* , $d' = 2, D = 3$*)*



$$\phi(x_1, x_2) = (x_1, x_2, -\|\boldsymbol{x}\|)$$



Non-linear mappings

- machine learning prefers to have separable, repeatable patterns (and so do we)

- can samples be better separable if we add some dimensions to their representation?

- here is a more complex one: *(non-linearly separable , $d' = 2, D = 3$)*

- motto: what you cannot easily separate in a d-dimensional space…
- …could be easily separable in a D-dimensional one.

- is there any way to systematically build a useful ϕ like the one we saw before?

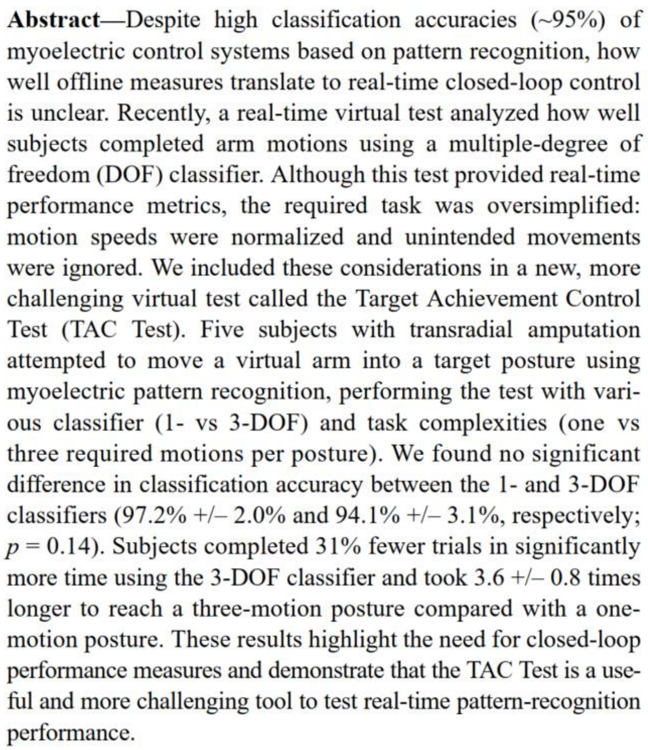
- yes there is, but this is the subject of a further lecture.

Simon AM, Hargrove LJ, Lock BA, Kuiken TA. Target Achievement Control Test: evaluating real-time myoelectric pattern-recognition control of multifunctional upper-limb prostheses. J Rehabil Res Dev. 2011;48(6):619-27.

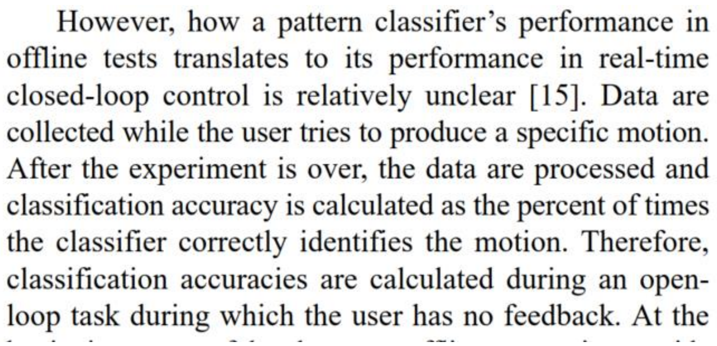
The TAC test

(Target Achievement Control)

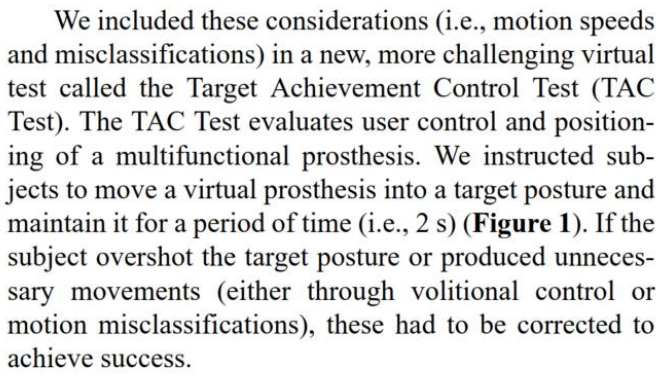
- a simple, easy-to-implement assessment test



Abstract—Despite high classification accuracies (~95%) of myoelectric control systems based on pattern recognition, how well offline measures translate to real-time closed-loop control is unclear. Recently, a real-time virtual test analyzed how well subjects completed arm motions using a multiple-degree of freedom (DOF) classifier. Although this test provided real-time performance metrics, the required task was oversimplified: motion speeds were normalized and unintended movements were ignored. We included these considerations in a new, more challenging virtual test called the Target Achievement Control Test (TAC Test). Five subjects with transradial amputation attempted to move a virtual arm into a target posture using myoelectric pattern recognition, performing the test with various classifier (1- vs 3-DOF) and task complexities (one vs three required motions per posture). We found no significant difference in classification accuracy between the 1- and 3-DOF classifiers (97.2% +/– 2.0% and 94.1% +/– 3.1%, respectively; $p = 0.14$). Subjects completed 31% fewer trials in significantly more time using the 3-DOF classifier and took 3.6 +/– 0.8 times longer to reach a three-motion posture compared with a one-motion posture. These results highlight the need for closed-loop performance measures and demonstrate that the TAC Test is a useful and more challenging tool to test real-time pattern-recognition performance.



However, how a pattern classifier's performance in offline tests translates to its performance in real-time closed-loop control is relatively unclear [15]. Data are collected while the user tries to produce a specific motion. After the experiment is over, the data are processed and classification accuracy is calculated as the percent of times the classifier correctly identifies the motion. Therefore, classification accuracies are calculated during an open-loop task during which the user has no feedback. At the



We included these considerations (i.e., motion speeds and misclassifications) in a new, more challenging virtual test called the Target Achievement Control Test (TAC Test). The TAC Test evaluates user control and positioning of a multifunctional prosthesis. We instructed subjects to move a virtual prosthesis into a target posture and maintain it for a period of time (i.e., 2 s) (**Figure 1**). If the subject overshot the target posture or produced unnecessary movements (either through volitional control or motion misclassifications), these had to be corrected to achieve success.

Simon AM, Hargrove LJ, Lock BA, Kuiken TA. Target Achievement Control Test: evaluating real-time myoelectric pattern-recognition control of multifunctional upper-limb prostheses. J Rehabil Res Dev. 2011;48(6):619-27.

The TAC test

- a simple, easy-to-implement assessment test

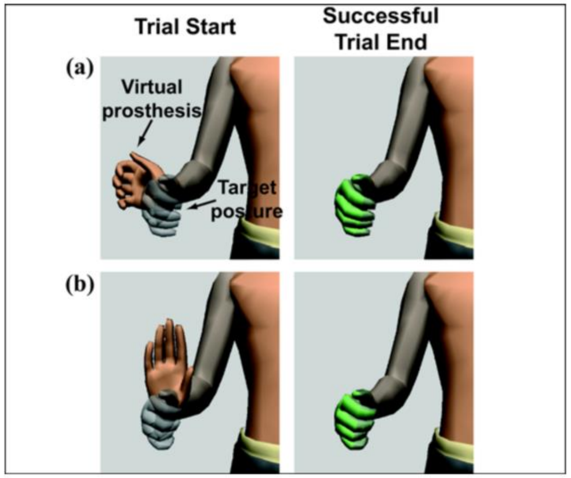


Figure 1.
Target Achievement Control Test (TAC Test). Subjects moved multifunctional virtual prosthesis into target posture. Virtual hand turned green when target was reached within acceptable tolerances (±5° for each degree of freedom). Figure illustrates starting and ending positions for successful trials. **(a)** Example trial from conditions 1 and 2 requiring one motion to reach target posture (e.g., wrist flexion). **(b)** Example trial from condition 3 requiring three motions to reach target posture (e.g., wrist flexion, wrist supination, and hand close).

Subjects trained the system to recognize seven motion classes (wrist flexion, wrist extension, wrist supination, wrist pronation, hand open, hand closed, and no movement). To train the pattern-recognition system, we prompted subjects by demonstrating each movement and asked them to perform the movement at a comfortable and consistent level of effort. In training, subjects held each contraction for 3 s, repeated eight times. We split

For the TAC Test, the subjects moved the virtual prosthesis from a nonneutral position to a neutral posture (target). The neutral position was 0° of wrist flexion or extension and 0° of wrist rotation (see **Figure 1**, successful trial end). To provide visual feedback, the virtual hand turned green when it was within an acceptable tolerance of the target (±5° for each DOF) (**Table 2**). Subjects completed tests more quickly if they only produced the motion(s) necessary to reach the target. If a subject overshot the target posture or produced unnecessary movements, he or she had to correct those motions to achieve success. TAC Test trials ended successfully when subjects were able to keep the virtual prosthesis in the target for 2 s. Target

Intent detection and somatosensory feedback

1. Why was the TAC Test Introduced?

- Limitations of Previous Tests: Prior assessment methods provided high classification accuracies (around 95%) but didn't effectively translate to real-time closed-loop control, making it unclear how these systems performed in practical, real-world scenarios.
- Need for Real-Time Metrics: Previous tests oversimplified tasks by normalizing motion speeds and ignoring unintended movements.
- Comprehensive Evaluation: The TAC Test aims to provide more challenging and realistic conditions to better understand how well the pattern classifiers perform in real-time closed-loop control, which involves subjects receiving no feedback during open-loop tasks.
- Assessment Scope: It evaluates not just classification accuracy but also user control and positioning of the prosthesis. This includes correcting unnecessary movements and maintaining a target posture for a set period.

Simon AM, Hargrove LJ, Lock BA, Kuiken TA. Target Achievement Control Test: evaluating real-time myoelectric pattern-recognition control of multifunctional upper-limb prostheses. J Rehabil Res Dev. 2011;48(6):619-27.

The TAC test

- a simple, easy-to-implement assessment test

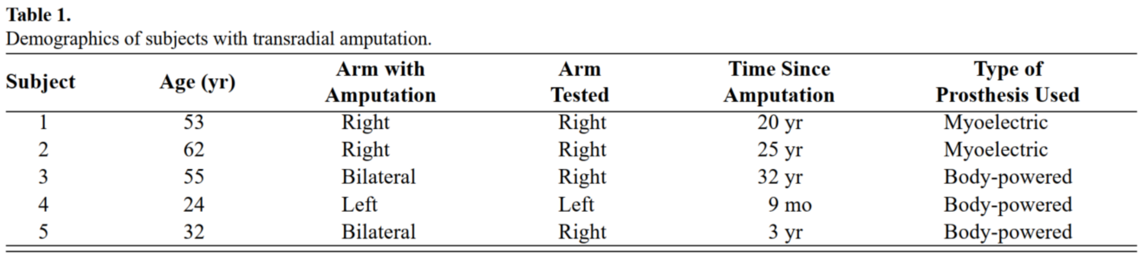


Table 1.
Demographics of subjects with transradial amputation.

Subject	Age (yr)	Arm with Amputation	Arm Tested	Time Since Amputation	Type of Prosthesis Used
1	53	Right	Right	20 yr	Myoelectric
2	62	Right	Right	25 yr	Myoelectric
3	55	Bilateral	Right	32 yr	Body-powered
4	24	Left	Left	9 mo	Body-powered
5	32	Bilateral	Right	3 yr	Body-powered

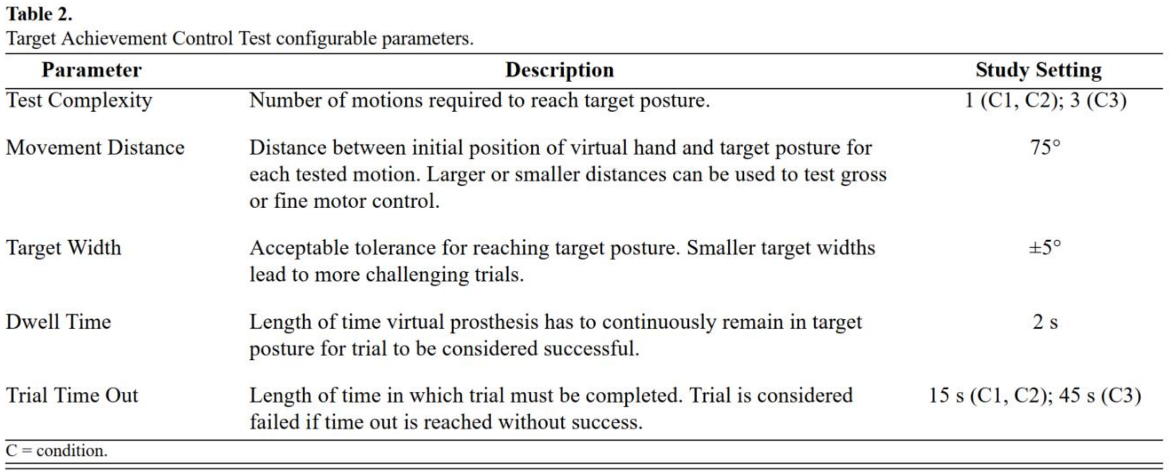


Table 2.
Target Achievement Control Test configurable parameters.

Parameter	Description	Study Setting
Test Complexity	Number of motions required to reach target posture.	1 (C1, C2); 3 (C3)
Movement Distance	Distance between initial position of virtual hand and target posture for each tested motion. Larger or smaller distances can be used to test gross or fine motor control.	75°
Target Width	Acceptable tolerance for reaching target posture. Smaller target widths lead to more challenging trials.	±5°
Dwell Time	Length of time virtual prosthesis has to continuously remain in target posture for trial to be considered successful.	2 s
Trial Time Out	Length of time in which trial must be completed. Trial is considered failed if time out is reached without success.	15 s (C1, C2); 45 s (C3)

C = condition.

Simon AM, Hargrove LJ, Lock BA, Kuiken TA. Target Achievement Control Test: evaluating real-time myoelectric pattern-recognition control of multifunctional upper-limb prostheses. J Rehabil Res Dev. 2011;48(6):619-27.

The TAC test

- a simple, easy-to-implement assessment test
- used completion rate, completion time and path efficiency

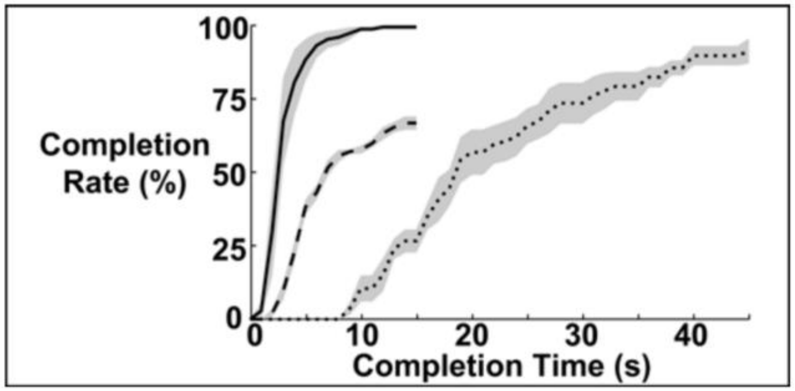
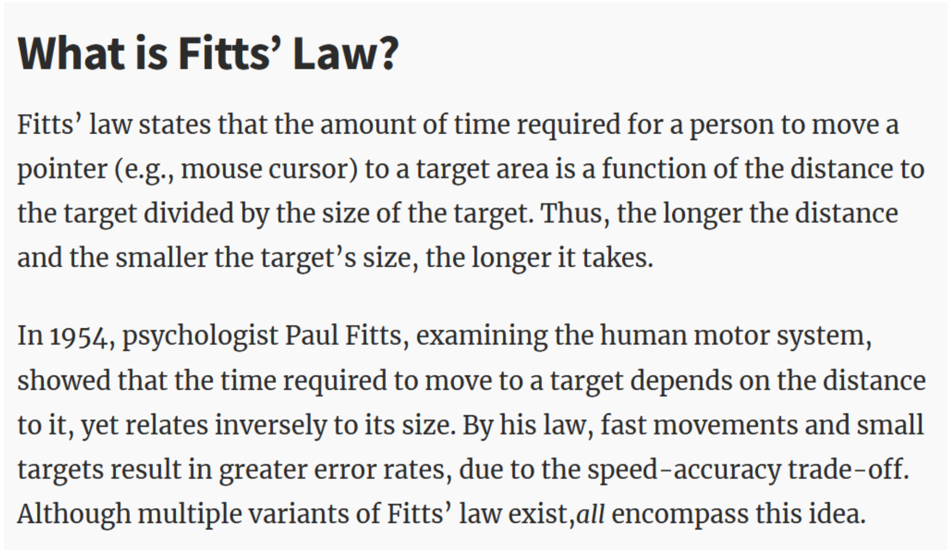


Figure 2.
Average completion rate curves for all three conditions. Solid line indicates performance during trials that required only one motion per posture using 1-degree of freedom (DOF) classifier (condition 1). Dashed line indicates performance during trials that required only one motion per posture using 3-DOF classifier (condition 2). Dotted line indicates performance during trials that required three motions per posture using 3-DOF classifier (condition 3). Shaded regions represent ±1 standard error.

The TAC test

- a simple, easy-to-implement assessment test
- online, highly tunable, easy to tailor on single participants, simple to implement – even on a screen – tunable – all the parameters can be adjusted
- not fully realistic but OK for online testing
- test “zero” in reha-robotics
- actually, an instance of target-reaching tasks, for which Fitts’ Law holds

Fitts' Law

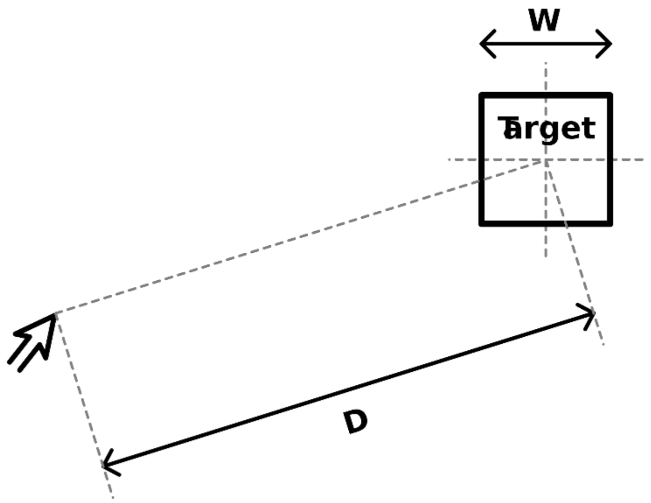


What is Fitts' Law?

Fitts' law states that the amount of time required for a person to move a pointer (e.g., mouse cursor) to a target area is a function of the distance to the target divided by the size of the target. Thus, the longer the distance and the smaller the target's size, the longer it takes.

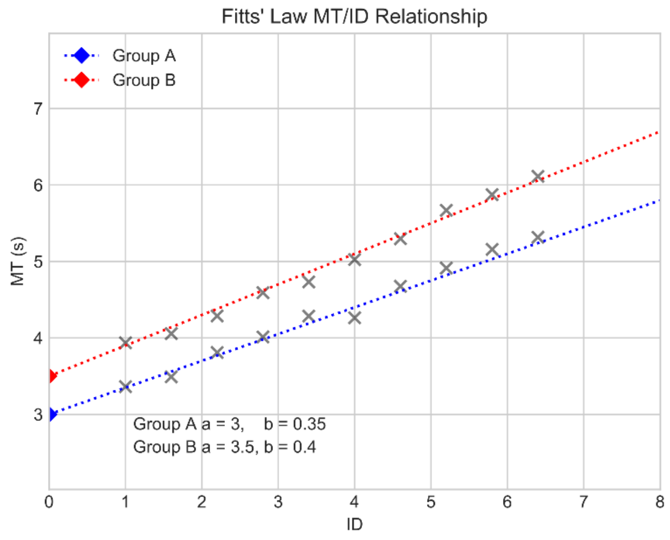
In 1954, psychologist Paul Fitts, examining the human motor system, showed that the time required to move to a target depends on the distance to it, yet relates inversely to its size. By his law, fast movements and small targets result in greater error rates, due to the speed-accuracy trade-off. Although multiple variants of Fitts' law exist,*all* encompass this idea.

- the time required to complete a target-reaching task is



$$ID = \log_2\left(\frac{2D}{W}\right)$$

$$TCT = a + b \cdot ID$$



Summary

- today:
  - ground truth and „defining an action“
- filtering and feature extraction
  - regions of interest and their „gradient“
  - non-linear mappings
- assessing an ID system: the TAC test and Fitts‘ Law

References

- B. Hudgins, P. Parker and R. N. Scott, *The Recognition Of Myoelectric Patterns For Prosthetic Limb Control*, **Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society** Volume 13: 1991, 1991, pp. 2040-2041, doi: 10.1109/IEMBS.1991.684880.
- Castellini, Claudio / Passig, Georg, *Ultrasound image features of the wrist are linearly related to finger positions,* 2011. **Proceedings of IROS - International Conference on Intelligent Robots and Systems,** p. 2108-2114
- Ilja Kuzborskij, Arjan Gijsberts, and Barbara Caputo, *On the Challenge of Classifying 52 Hand Movements from Surface Electromyography*, **Annual International Conference of the IEEE Engineering in Medicine and Biology Society,** pp. 4931-4937, 2012
- John L. Semmlow, Benjamin Griffel, *Biosignal and Medical Image Processing (3rd Edition),* ISBN 9781466567368, CRC Press, 2014.
- Angkoon Phinyomark, Rami N. Khushaba and Erik Scheme, *Feature Extraction and Selection for Myoelectric Control Based on Wearable EMG Sensors,* **MDPI Sensors** 2018, 18, 1615
- Simon AM, Hargrove LJ, Lock BA, Kuiken TA. Target Achievement Control Test: evaluating real-time myoelectric pattern-recognition control of multifunctional upper-limb prostheses. J Rehabil Res Dev. 2011;48(6):619-27.