PONTIFICIA
UNIVERSIDAD
CATÓLICA DE
VALPARAÍSO

pucv.cl

Tool Tracking Dataset

Multi-modal / multi-class Time-Series Classification and Segmentation Dataset

Escuela de Ingeniería Informática

Valparaíso, 2025

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

Agenda

- Materials
- Motivation
- System Design
- Hand-held tools
- Time-Series Dataset
- Summary

Materials

Data

- GitHub: https://github.com/crispchris/tool-tracking
- Data https://christofferloeffler.com/tooltracking/tool-tracking-data.zip

Papers

- Löffler et al.: *"Automated Quality Assurance for Hand-Held Tools via Embedded Classification and AutoML"*. In: ECML 2020. pdf demo presentation
- Redzepagic et al.: *"A Sense of Quality for Augmented Reality Assisted Process Guidance"*. In: ISMAR 2020. paper
- Mishra et al.: *"Recipes for Post-training Quantization of Deep Neural Networks"*. In: EMC^2 2020. paper
- Löffler et al.: *"Don't Get Me Wrong: How to Apply Deep Visual Interpretations to Time Series"*. Arxiv preprint, 2023. preprint

Project websites

- https://www.iis.fraunhofer.de/en/ff/lv/dataanalytics/anwproj/werkzeug.html
- https://www.iis.fraunhofer.de/en/ff/lv/dataanalytics/anwproj/ki4tools.html

Images: © Fraunhofer IIS

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

Motivation

Quality Assurance in Assembly

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

Tesla again shuts down Model 3 production line
Model 3 production halted

Automation knows limits

Human labor remains important

- flexible, literally handy with many kinds of tools

Meanwhile at BMW (Regensburg, Germany)

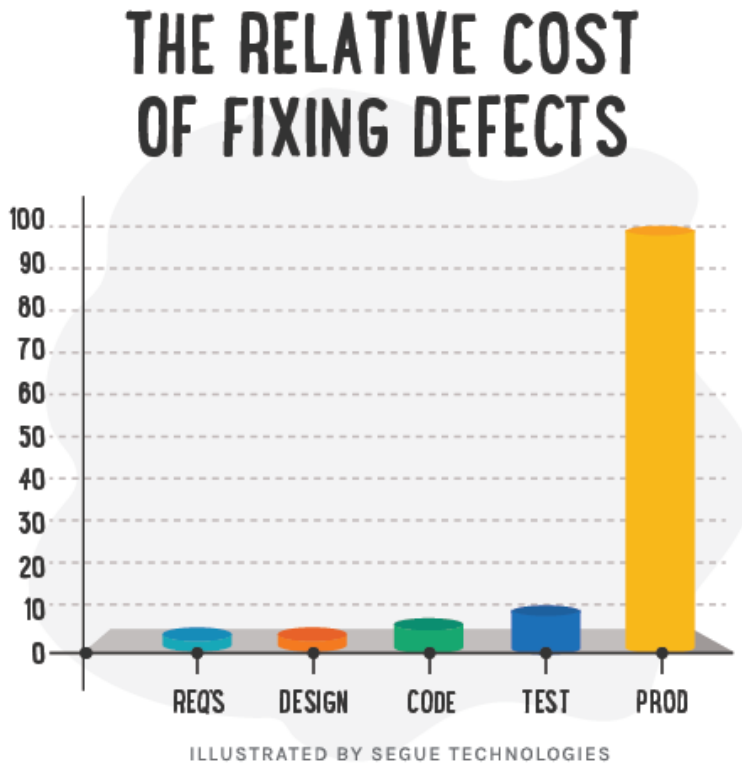
However, human labor is not automated

- gaps in quality assurance (QA)
  - long work shifts and distractions can lead to mistakes

Sometimes human workers make mistakes

It is worth it to detect mistakes early

- Cost of fixing mistakes relative to when they are discovered.



Problem analysis and how to assure quality

Problem Analysis

- Workers in assembly perform a manual and repetitive task
- Mistakes through lacking concentration or changing assembly steps (build-to-order, customized cars, ..)
- Work is harsh on the worker's body

Quality Assurance systems

- as post-production step for randomly sampled cars
- Expensive, heavy tools
- Optical camera systems are limited by design

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

System Design

Requirements and Solutions

Requirements for next generation QA systems

Tools themselves must detect and monitor the work process

- Make tools intelligent and light weight
- Automate Quality Assurance

System's challenges

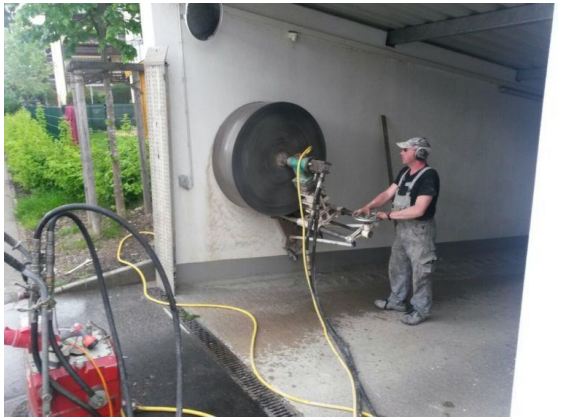
Challenges

1. Must work with many different hand-held tools to be financially viable as a product

System's challenges

Challenges

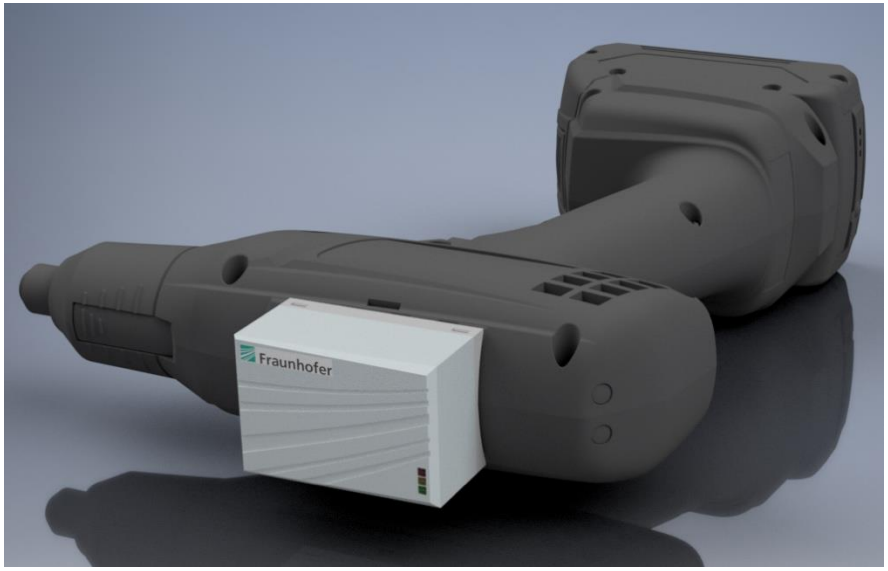
2. Must be easily applied to a wide variety of tasks, quickly and cheaply



Hardware description

Add Machine Learning to any hand-held tool

- Multi-modal sensors
- Analyze data on device
- Wireless data transfer to QA



Hardware description

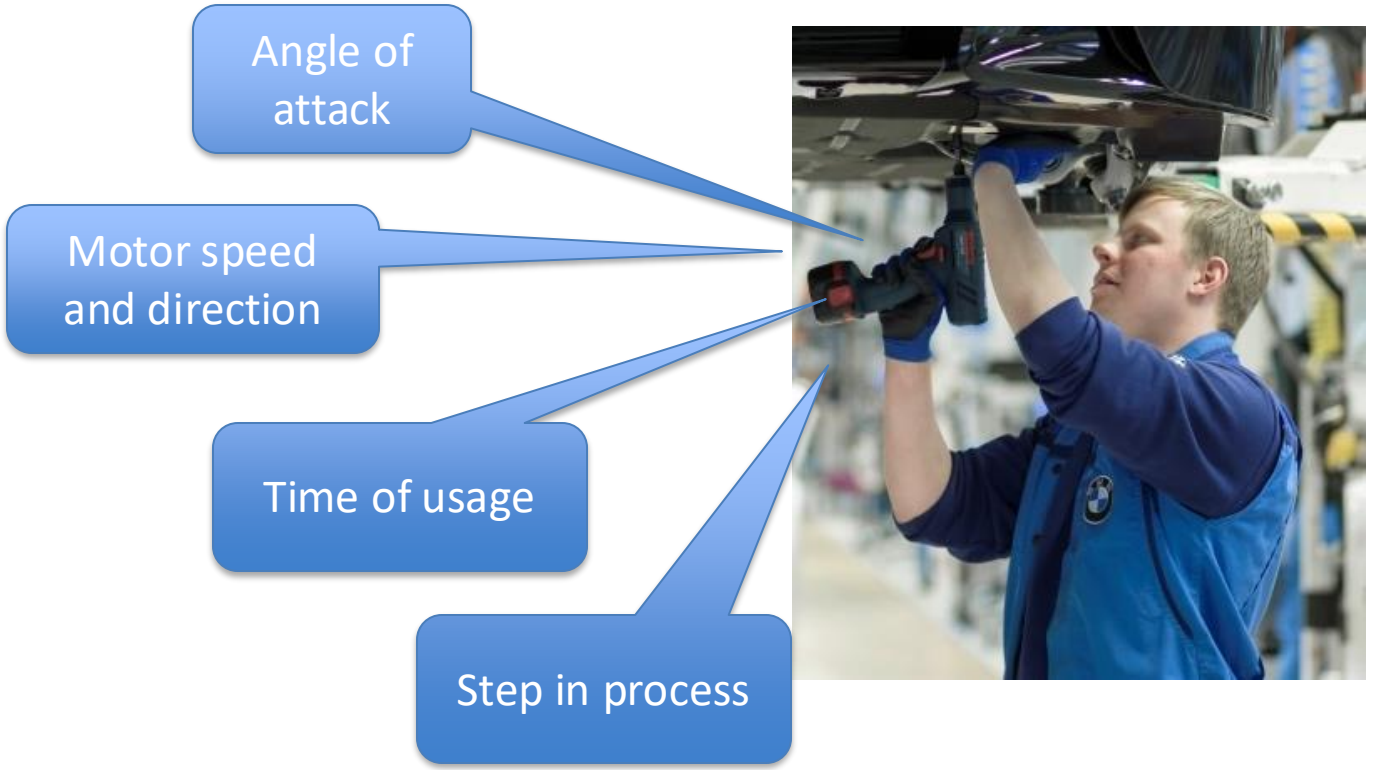
After-market add-on box

- 3D-printed socket, slide on hardware to fix to device

What do we need to measure?

Depends on how a hand-held tool is used!

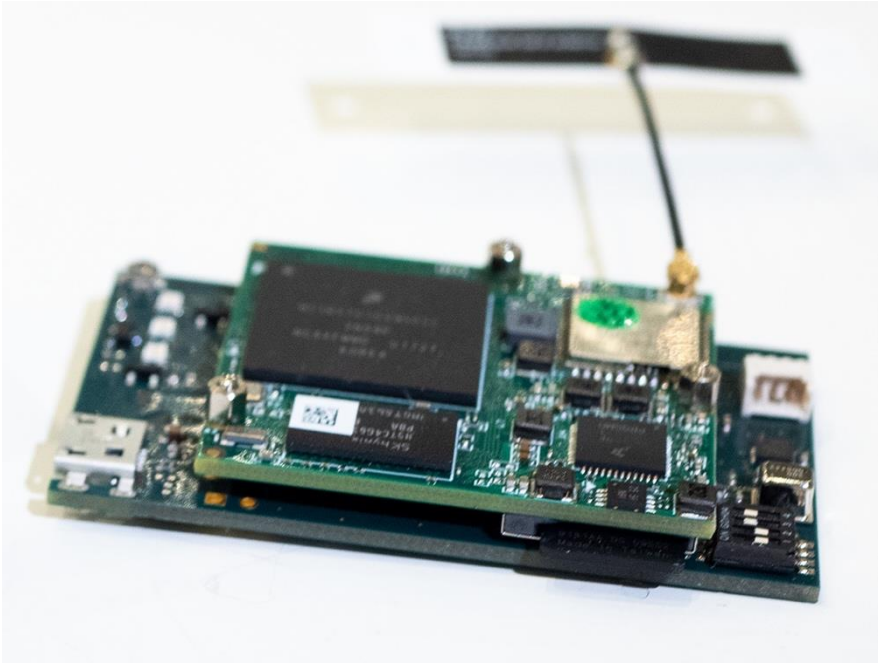
Example: Electric Screwdriver



A look inside

Sensor specifications

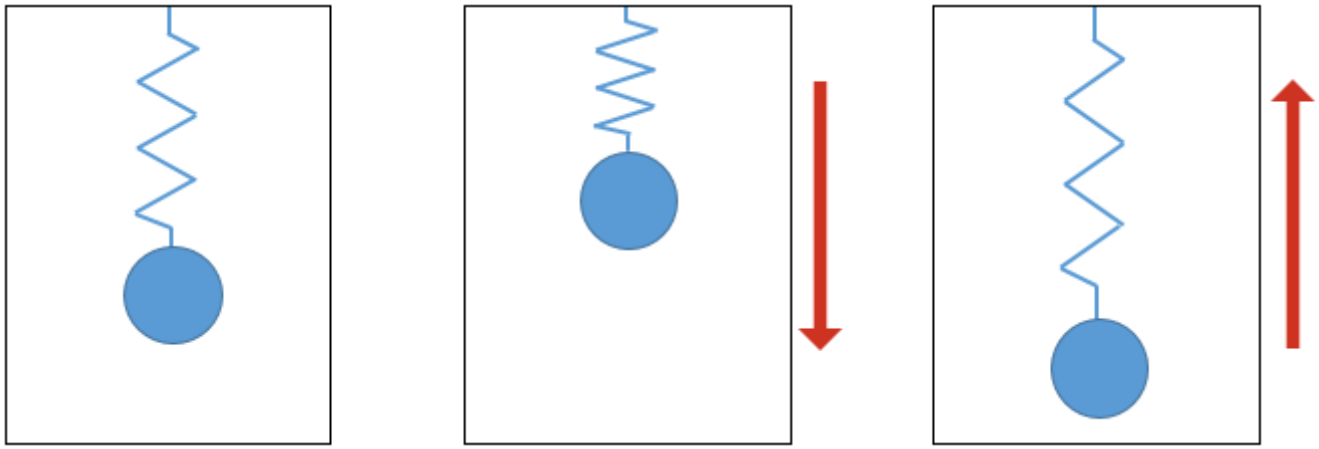
- 9-axis Inertial Measurement Unit (IMU)
- Microphone



Inertial Measurement Unit

Accelerometer

- 3-axis at 102.292 Hz
- Measures acceleration $\frac{m}{s^2}$
- Acceleration towards earth is "gravity" $9.81\frac{m}{s^2}$



Motion down compresses spring

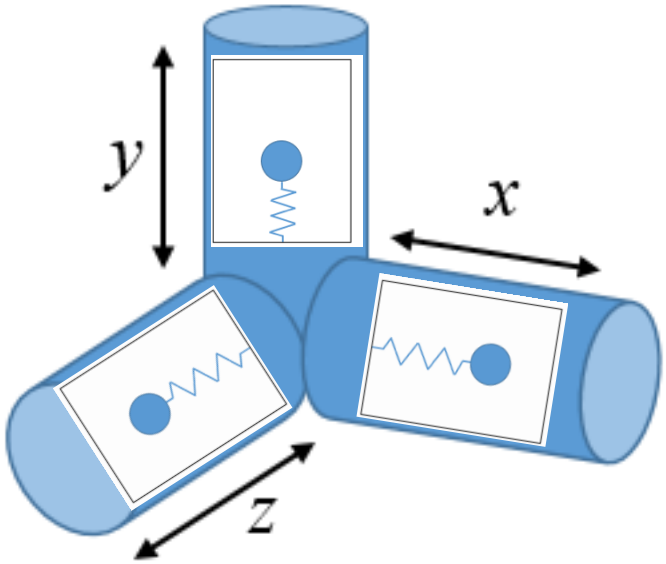
Motion up decompresses spring

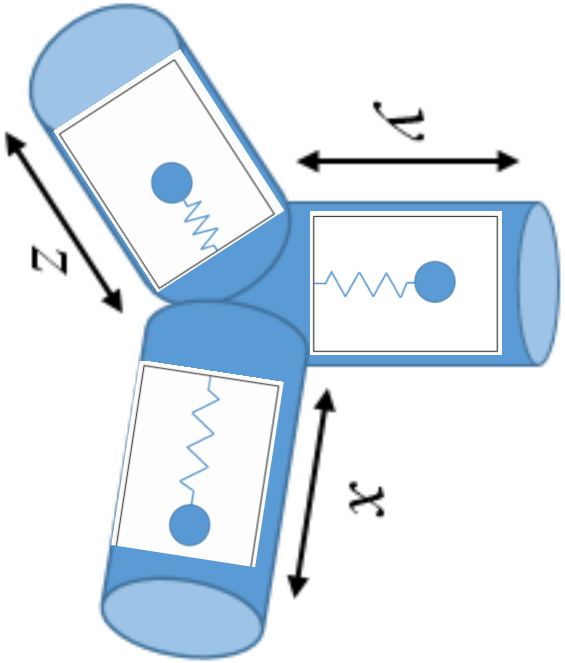
Source: https://bdavison.napier.ac.uk/iot/Notes/components/accelerometer/

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

Accelerometer

- 3-axis: Why?





Source: https://bdavison.napier.ac.uk/iot/Notes/components/accelerometer/

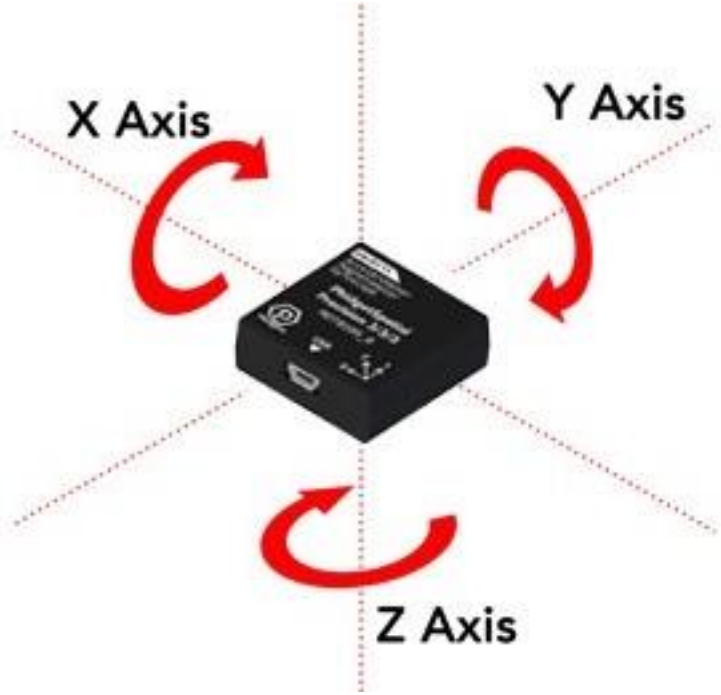
Inertial Measurement Unit

Gyroscope

- 3 axis at 102.292 Hz
- Measures rotation degrees $\frac{°}{s}$

Example:

- no motion, values 0.0
- Rotation around axis



Source: https://www.phidgets.com/docs/Gyroscope_Guide

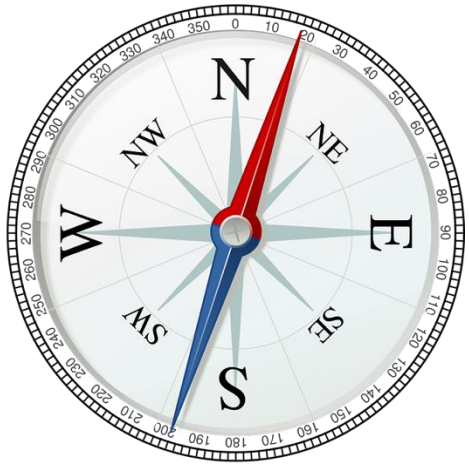
Inertial Measurement Unit

Magnetic Field sensor

- 3-axis compass at 154.646 Hz
- Measures strength of magnetic field in Gauss
- Axis that points north should have higher reading

Example:

- Earth magnetic field: 0.25 to 0.65 G
- Toaster: 2-10 mG
- Power Transmission Line: 10-200 mG
- Engine: 4000-6000 G

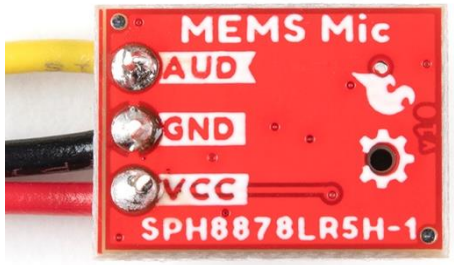


Source: https://www.arpansa.gov.au/understanding-radiation/radiation-sources/more-radiation-sources/measuring-magnetic-fields

Microphone

Microphone
- Records mono sounds at 44 kHz

Examples:
- Engine sounds
- Hammering
- Anything in the background



Proposed system

9-axis Inertial Measurement Unit (IMU)

- Movements and accelerations
  - Vibrations, motion, hits, ..
- Rotations
  - Interpret motion better
- Magnetic or electro-magnetic fields
  - Absolute orientation, electric motors, ..

Source: https://www.youtube.com/watch?v=i5Z8JOdyCpk

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

Hand-held Tools

The tools and their characteristics

Hand-held tools

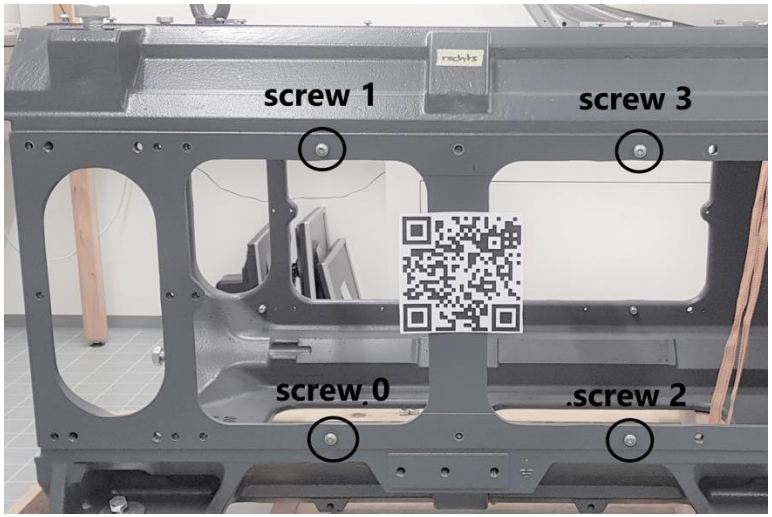
Tools recorded in the dataset

- Electric Screwdriver
- Pneumatic Rivet Gun
- Pneumatic Screwdriver

Hand-held tools

Electric Screwdriver

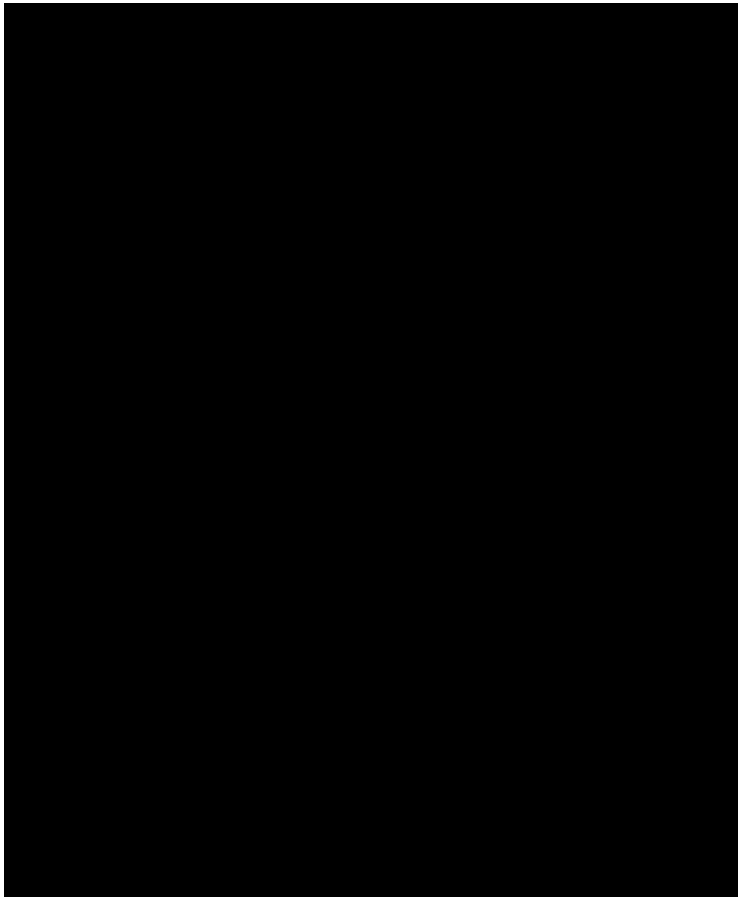
- Has an electric engine
  - Induces strong magnetic field → mag
  - Smooth operation, low vibration
- Tighten screws until tight enough, stops automatically
  - Strong twisting rotation measurable → gyr



PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

Demonstration

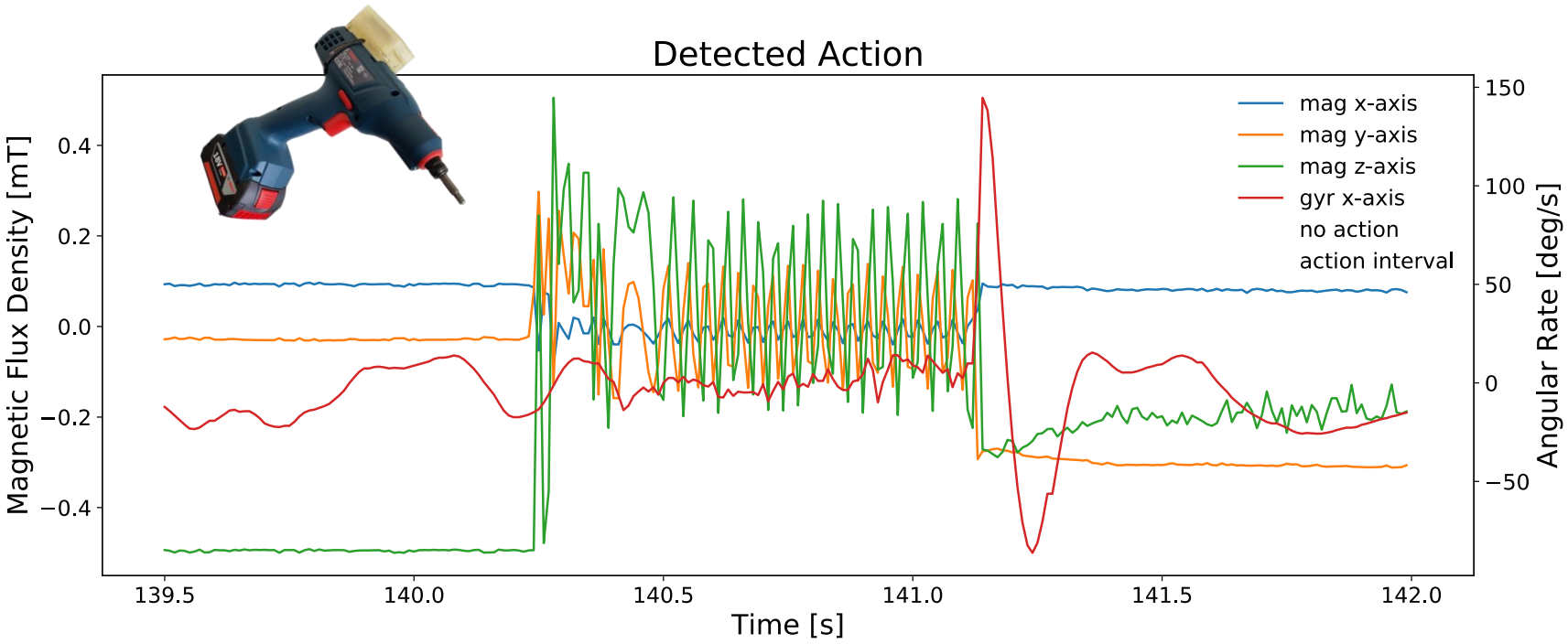
- Notice the motion
- End of screwing: harsh stop



Hand-held tools

Typical sensor reading

- Magnetic field very strong
- Gyroscope shows twisting rotation at the end



Hand-held tools

Pneumatic Screwdriver

- Similar to electric variant
- Made for heavy duty work

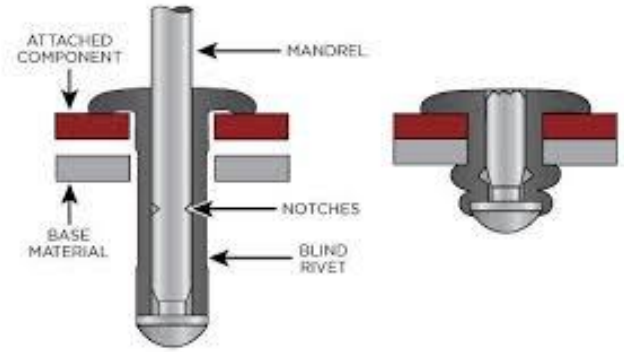
Works with pressurized air

- No induced electromagnetism → mag is useless
- Longer, very loud action with specific acceleration profile
  - Mic, acc, gyr
- No hard deceleration when done
  - Loud, continuous clacking noise instead

Hand-held tools

Pneumatic Rivet Gun

- Works with pressurized air
  - No induced electromagnetism → mag is useless
- Short, loud action with specific acceleration profile
  - Mic, acc, gyr



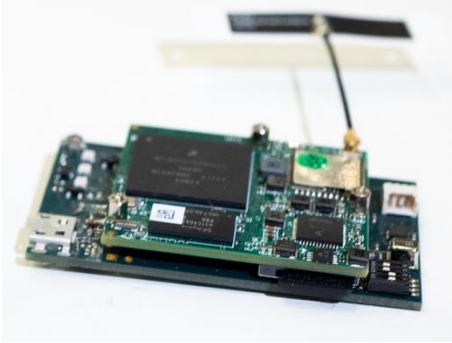
Sources: youtube and https://www.huyett.com/blog/what-is-a-blind-rivet

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

Time-Series Dataset

multimodality, multiclass,
anomalies & out-of-distribution,
classification or segmentation, ...

Terminology



Multimodality:
- 4 types of sensors
- **Orthogonal information**

Multiclass:
- More than 2 classes per tool
- Multilabel: **hierarchical labels** can be constructed from data

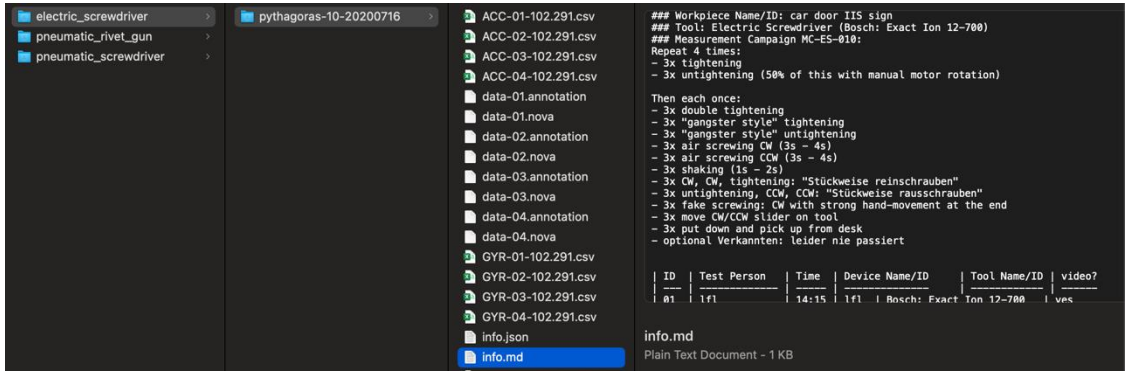
Difficulty and Realism
- **Anomalies**: recordings of errors on purpose
- **Out-of-distribution**: test your models against specific hold-out classes unseen during training, e.g., erroneous anomalies

Tasks
- **Classification** of windows or **segmentation** of data stream

Hand-held tools: Actions

Electric Screwdriver's actions
- Tighten, Untighten

Special actions (Optional anomalies)
- *Screwing (Clockwise, Counter Clockwise)*
- *Double tightening (already tightened)*
- *Held sideways ("gangster style")*
- *Screwing but in Air*
- *Shaking the tool*
- *Picking it up, setting it down*
- *Faking the screwing by rotation device head by hand*
- *Moving the settings-slider on the tool*



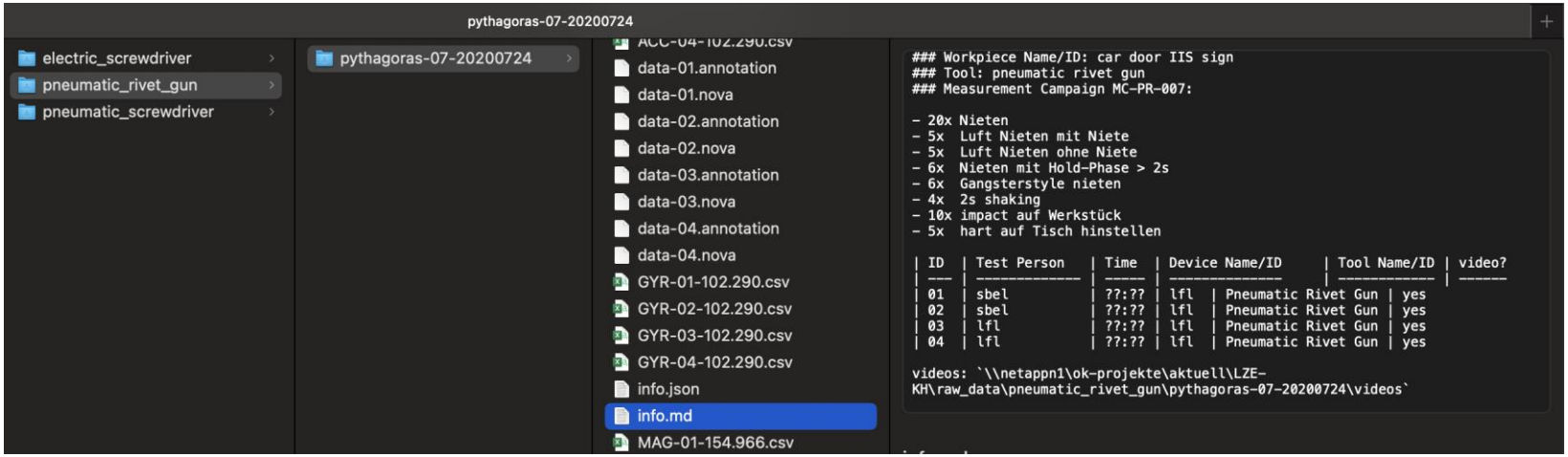
Hand-held tools: Actions

Pneumatic Rivet Gun's actions

- Rivetting

Special actions (Optional anomalies)

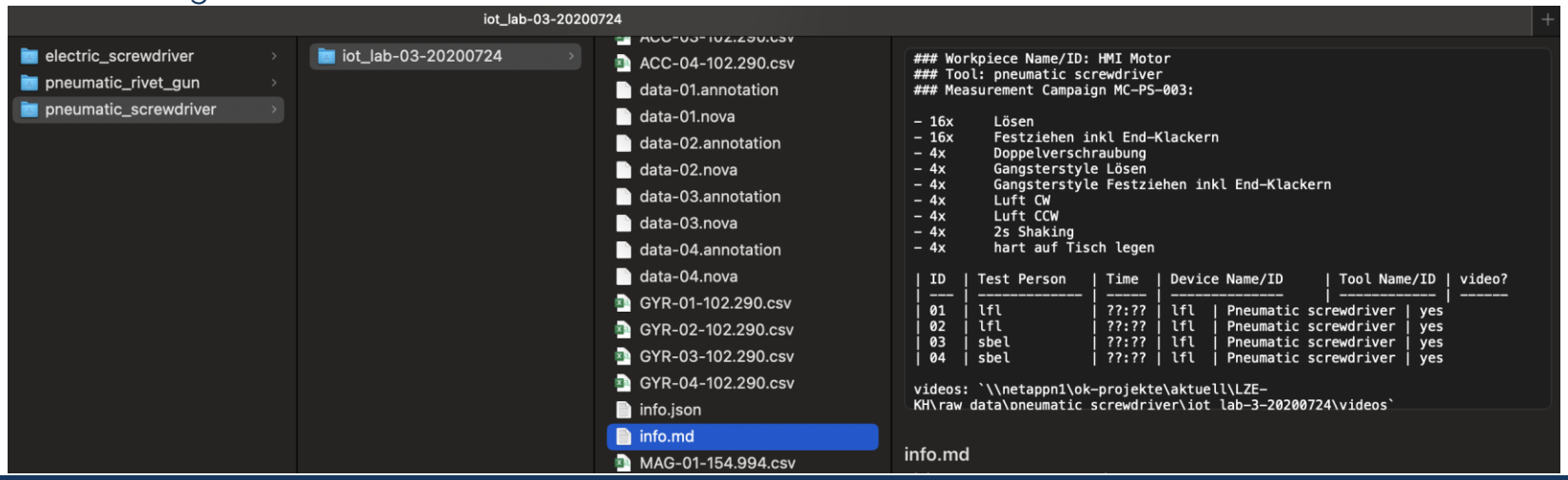
- Rivetting in the air: with/without rivet inserted before
- Rivetting done with waiting before moving tool again
- Shaking the riveter
- Held sideways
- Hitting the work piece
- Putting down hard on table



Hand-held tools: Actions

Pneumatic Screwdriver's actions
- Tightening
- Untightening

Special actions (Optional anomalies)
- Double tightening
- Tool-use in air without screws (CW/CCW)
- Shaking
- Held sideways
- Putting down on table



Dataset

Recording of **large dataset**

- 3 tools:
  - Electric Screwdriver
  - Pneumatic Rivet Gun
  - Pneumatic Screwdriver

Repeated **recordings**

- 2 different workers (lfl, sbel)
- 2 recording runs per person

Structured recordings

- exact list what to do with each tool, simplifies annotation
- Repetitions of actions within each run

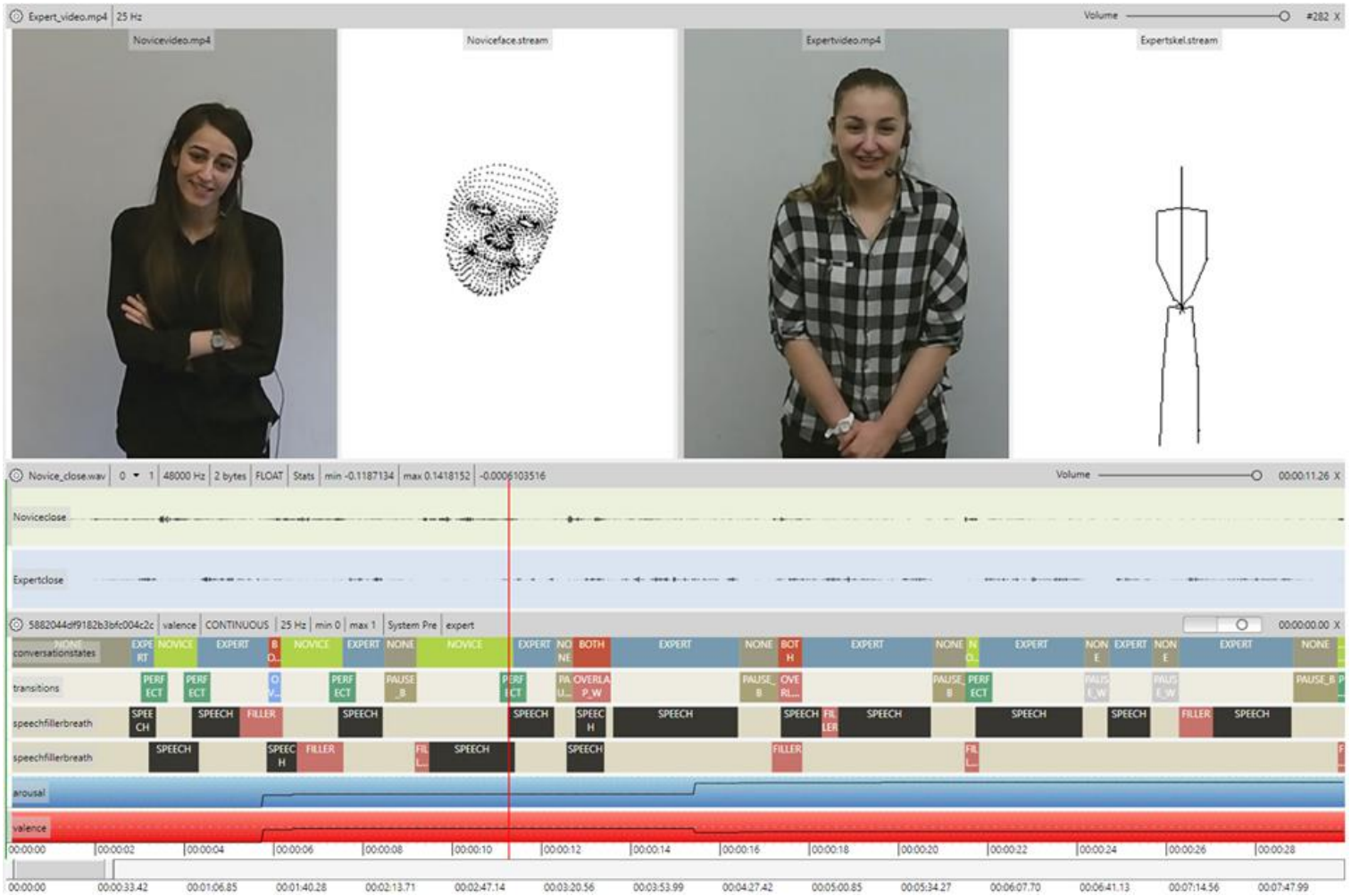
```
### Workpiece Name/ID: car door IIS sign
### Tool: Electric Screwdriver (Bosch: Exact Ion 12-700)
### Measurement Campaign MC-ES-010:
Repeat 4 times:
- 3x tightening
- 3x untightening (50% of this with manual motor rotation)

Then each once:
- 3x double tightening
- 3x "gangster style" tightening
- 3x "gangster style" untightening
- 3x air screwing CW (3s - 4s)
- 3x air screwing CCW (3s - 4s)
- 3x shaking (1s - 2s)
- 3x CW, CW, tightening: "Stückweise reinschrauben"
- 3x untightening, CCW, CCW: "Stückweise rausschrauben"
- 3x fake screwing: CW with strong hand-movement at the end
- 3x move CW/CW slider on tool
- 3x put down and pick up from desk
- optional Verkannten: leider nie passiert


| ID  | Test Person  | Time  | Device Name/ID      | Tool Name
| --- | ------------ | ----- | ------------------- | ----------
| 01  | lfl          | 14:15 | lfl  | Bosch: Exact Ion 12-700
| 02  | lfl          | 14:45 | lfl  | Bosch: Exact Ion 12-700
| 03  | sbel         | 15:?? | lfl  | Bosch: Exact Ion 12-700
| 04  | sbel         | 15:?? | lfl  | Bosch: Exact Ion 12-700
```

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

Annotations tool: Nova



Source: https://github.com/hcmlab/nova

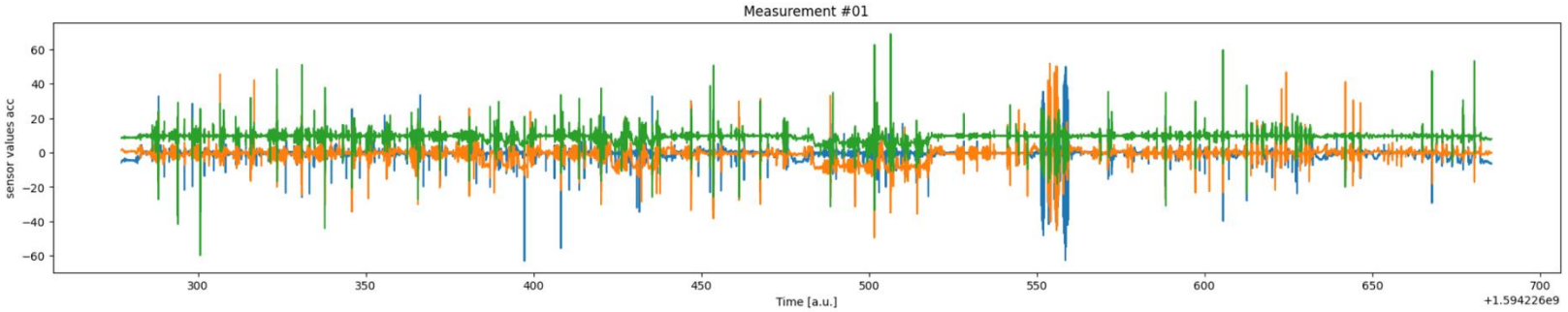
PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

Annotations

Dense Labels

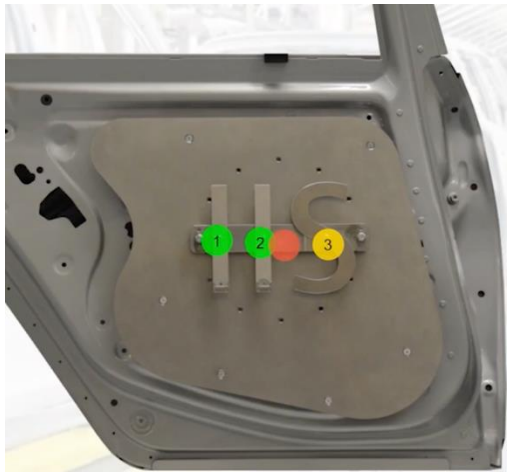
- each sample from each sensor has class annotation

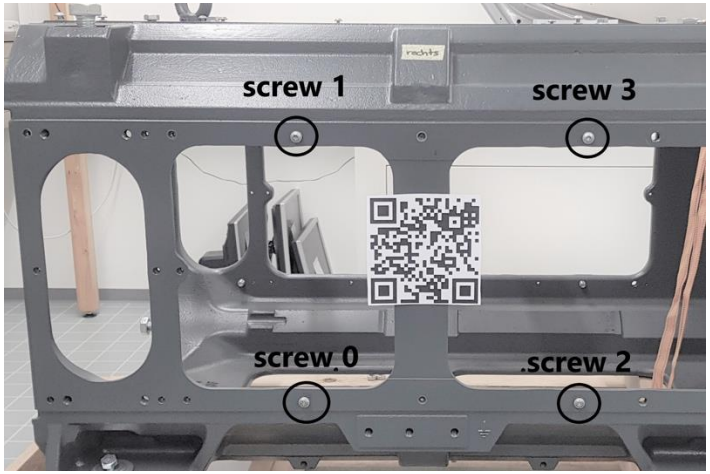
→ You can cut windows from the data (overlapping, tumbling, ..)
→ You can continuously classify the “data stream”
→ Implement anomaly/out-of-distribution detection
→ You can implement time-series segmentation (sample-level prediction)



PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

Different work pieces:
BMW doors or engine block.

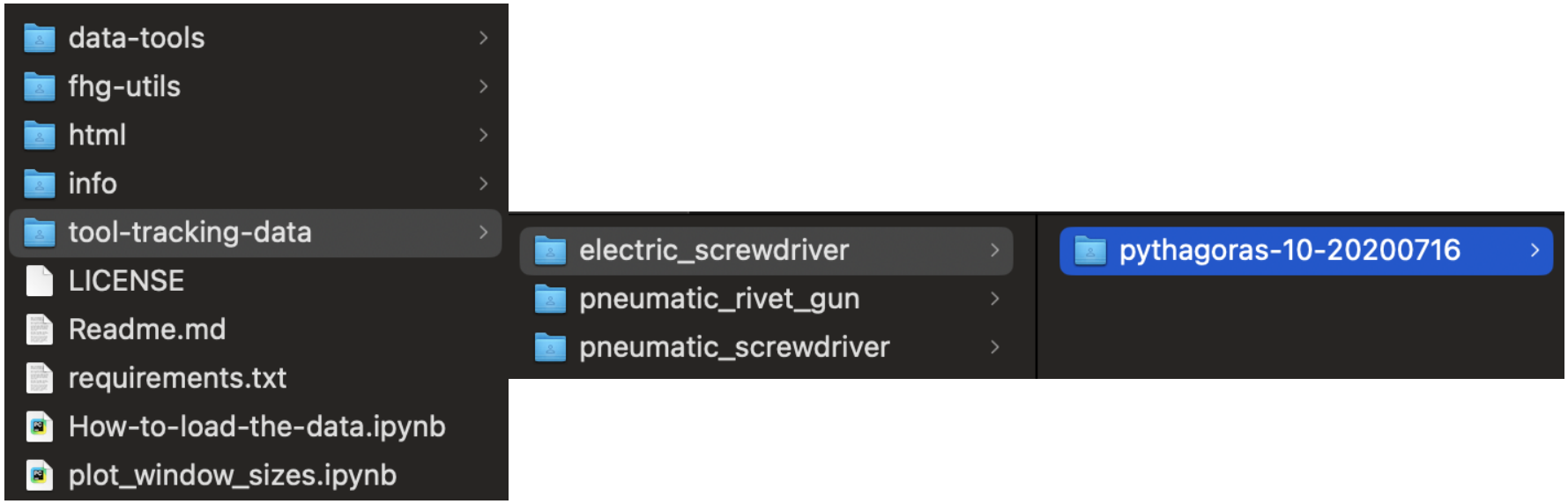




Example data: Electric Screwdriver

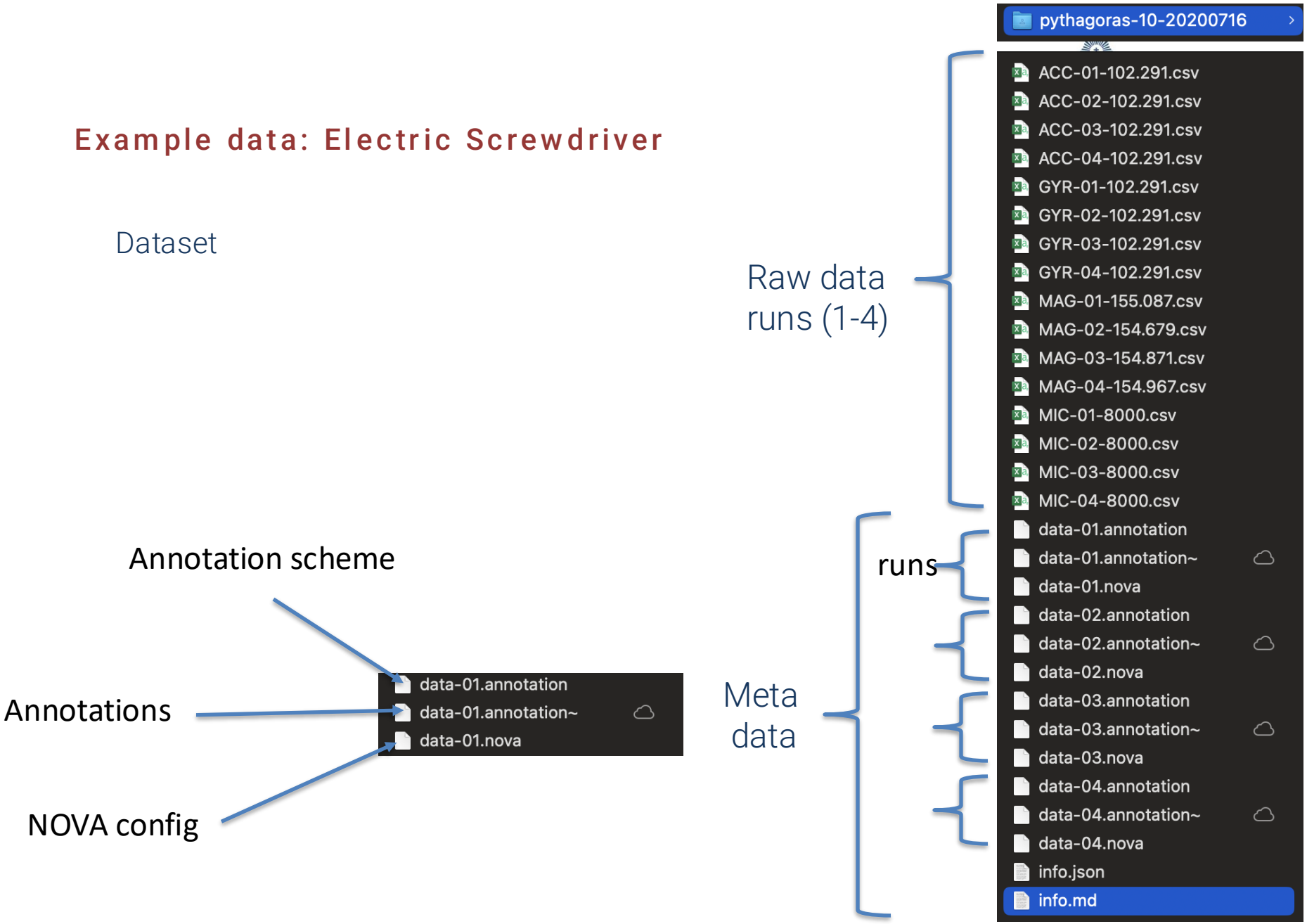
Folder structure

Data:	`tool-tracking-data`
Data Loader:	`data-tools, fhg-utils`
Documentation:	`html`
Demonstrations:	`How-to-load-the-data.ipynb`
	`plot_window_sizes.ipynb`



Example data: Electric Screwdriver

Dataset



Example data: Electric Screwdriver

Raw data:
ACC: Accelerometer
01: run ID 01 (here, Ifl recorded this run)
102.291 sampling rate in Hz

ACC-01-102.291.csv

The file size (3.17 MB) exceeds the configured limit (2.56 MB). Code insight features are not available.

```
time [s];acceleration x-axis [m/s^2];acceleration y-axis [m/s^2];acceleration z-axis [m/s^2]
1594226277.18598;-5.877378;1.322051;8.612491
1594226277.1957562;-5.819896530061611;1.2933097562023466;8.59332943464816
1594226277.2055323;-5.75962317633312;1.2423090772136065;8.570147747798787
1594226277.2153087;-5.678890970294857;1.2174626285881522;8.56938
1594226277.2250845;-5.483503755701358;1.2118799999999998;8.418252449596332
1594226277.2348607;-5.425912645010244;1.2384668129938543;8.358493313432836
1594226277.2446368;-5.307209670958254;1.2981775936644415;8.333272023079536
1594226277.2544131;-5.182684541288656;1.2792715650424349;8.30552778197249
1594226277.264189;-5.058761732186216;1.4256730596750091;8.26228634936496
1594226277.2739651;-4.918982586152233;1.430202637026559;8.218632913957515
```

Example data: Electric Screwdriver

Annotation scheme and range-based annotations

"Undefined" (idle) *id* = 8

Invalid (trash) *id* = -1

$t_{start}; t_{stop}; id; 1$

data-01.annotation

```
<?xml version="1.0" ?>
<annotation ssi-v="3">
    <info ftype="ASCII" size="157" />
    <meta annotator="" role=""/>
    <scheme name="electric_screwdriver" type="DISCRETE"  color="#FFFFFFFF">
        <item name="undefined" id="0" color="#FFD3D3D3" />
        <item name="tightening" id="1" color="#FF1F77B4" />
        <item name="untightening" id="2" color="#FFFF7F0E" />
        <item name="motor_activity_cw" id="3" color="#FF2CA02C" />
        <item name="motor_activity_ccw" id="4" color="#FFD62728" />
        <item name="manual_motor_rotation" id="5" color="#FF9467BD" />
        <item name="shaking" id="6" color="#FF8C564B" />
        <item name="tightening_double" id="7" color="#FFE377C2" />
    </scheme>
</annotation>
```

data-01.annotation~

```
0;8.76;0;1
8.76;11.144771;1;1
11.144771;14.56;0;1
14.56;16.88;1;1
16.88;20.920887;0;1
20.92;23.56;1;1
23.56;29.36;0;1
29.36;30.52;2;1
30.52;34.020882;0;1
34.020882;35.08;2;1
35.08;38.48;0;1
38.48;39.6;2;1
39.6;43.72;0;1
43.72;46.44;1;1
46.44;51.36;0;1
51.36;53.88;1;1
53.88;58.08;0;1
58.08;60.72;1;1
60.72;68.64;0;1
68.64;69.76;2;1
```

How to load the data

1. How to load the Data

Software setup:

```
conda create --name tool-tracking_env python=3.12
conda activate tool-tracking_env
pip install -r requirements.txt
jupyter notebook
```

1.2. 1. Load Data with `data-tools`

First we have to connect to our downloaded data using the main class:

[1]:
```
from pathlib import Path
from datatools.fixture import setup_dummy_data
# test_tool
# source = setup_dummy_data(5)

# put data into this source folder
source = "./tool-tracking-data/"
```

[5]:
```
from datatools import MeasurementDataReader, Measurement, MeasurementSeries, Action
mdr = MeasurementDataReader(source=source)
```

How to load the data

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

1.2.1. 1.1 Create a Query to load *Measurements* or *Actions*

Using the `mdr` object we can create a `Query` object to formulate a query to the data source.

[6]:
```
q = mdr.query(query_type=Measurement)
```

Depending on the use case you can query for *Measurements* which represents a temporal contiguous time series with inhomogeneous labels or *Actions* which are parts of *Measurements* but have the same label throughout their time range:

```
from datatools import Action
q = mdr.query(query_type=Action)
```

1.2.2. 1.2 Narrow a Query to load *Measurements* of interest

[7]:
```
from datatools import Tool, Config, MeasurementSeries, Measurement, DataTypes, Action
from datatools import ACC, GYR, MAG, MIC, POS, VEL

mytool = "electric_screwdriver"
#mytool = "pneumatic_screwdriver"
#mytool = "pneumatic_rivet_gun"
#mytool = "test_tool"

data_dict = q.filter_by(Tool == mytool, DataTypes == ACC).get()
```

How to load the data: Load Measurement

1.2.2. 1.2 Narrow a Query to load *Measurements* of interest

[7]:
```
from datatools import Tool, Config, MeasurementSeries, Measurement, DataTypes, Action
from datatools import ACC, GYR, MAG, MIC, POS, VEL

mytool = "electric_screwdriver"
#mytool = "pneumatic_screwdriver"
#mytool = "pneumatic_rivet_gun"
#mytool = "test_tool"

data_dict = q.filter_by(Tool == mytool, DataTypes == ACC).get()
```

```
[INFO] Preparing data from:
  tool-tracking-data/electric_screwdriver/pythagoras-10-20200716
  tool-tracking-data/electric_screwdriver/pythagoras-10-20200716
  tool-tracking-data/electric_screwdriver/pythagoras-10-20200716
  tool-tracking-data/electric_screwdriver/pythagoras-10-20200716
[INFO] Read annotation: 100%|████████████████████████████████████████████████████
████████████████| 16/16 [00:00<00:00, 176.74it/s, file=data-03.annotation]
[WARN] The mean sampling rate from the filename (tool-tracking-data/electric_screwdrive
r/pythagoras-10-20200716/ACC-02-102.291.csv) and from the timestamps differ by 0.001Hz

[WARN] The mean sampling rate from the filename (tool-tracking-data/electric_screwdrive
r/pythagoras-10-20200716/ACC-04-102.291.csv) and from the timestamps differ by 0.001Hz

[WARN] The mean sampling rate from the filename (tool-tracking-data/electric_screwdrive
r/pythagoras-10-20200716/ACC-01-102.291.csv) and from the timestamps differ by 0.001Hz

[WARN] The mean sampling rate from the filename (tool-tracking-data/electric_screwdrive
r/pythagoras-10-20200716/ACC-03-102.291.csv) and from the timestamps differ by 0.001Hz

[INFO] Finished with 4 measurement(s).
```

How to load the data: Load Measurement

1.2.2. 1.2 Narrow a Query to load *Measurements* of interest

[7]:
```
from datatools import Tool, Config, MeasurementSeries, Measurement, DataTypes, Action
from datatools import ACC, GYR, MAG, MIC, POS, VEL

mytool = "electric_screwdriver"
#mytool = "pneumatic_screwdriver"
#mytool = "pneumatic_rivet_gun"
#mytool = "test_tool"

data_dict = q.filter_by(Tool == mytool, DataTypes == ACC).get()
```

...

[8]:
```
data_dict.keys()  # we have loaded five measurements
```

[8]:
```
dict_keys(['02', '04', '01', '03'])
```

[6]:
```
data_bunch = data_dict["01"] # data bunch for measurement with measurement id "01"
type(data_bunch)
```

[6]:
```
datatools.data.DataBunch
```

How to load the data: Load Measurement

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

1.2.3. 1.3 Interaction with the `DataBunch` class

A DataBunch object is a dictionary-like object that exposes its keys as attributes. Access the data with `data_bunch.acc` or `data_bunch['audio']`. You can inspect which data is contained with:

[7]:
```
data_bunch.data_keys()
```

[7]:
```
['acc']
```

[8]:
```
data_bunch.acc.features
```

[8]:
```
['acceleration x-axis [m/s^2]',
 'acceleration y-axis [m/s^2]',
 'acceleration z-axis [m/s^2]']
```

[9]:
```
# matrix with raw sensor data with time column
Xt = data_bunch.acc.ts # numpy array

# matrix with raw sensor data (without time column)
X = data_bunch.acc.X  # numpy array

# target vector
y = data_bunch.acc.y  # numpy array
```

[10]:
```
print("shape of Xt: ", Xt.shape)
print("shape of X: ", X.shape)
print("shape of y: ", y.shape)
```

```
shape of Xt:  (41775, 4)
shape of X:  (41775, 3)
shape of y:  (41775,)
```

Alternative: Load ACTIONS

1.2.2. 1.2 Narrow a Query to load *Measurements* of interest

[7]:
```
from datatools import Tool, Config, MeasurementSeries, Measurement, DataTypes, Action
from datatools import ACC, GYR, MAG, MIC, POS, VEL

mytool = "electric_screwdriver"
#mytool = "pneumatic_screwdriver"
#mytool = "pneumatic_rivet_gun"
#mytool = "test_tool"

data_dict = q.filter_by(Tool == mytool, DataTypes == ACC).get()
```

1.2.4. 1.4 Narrow a Query to load *Actions* of interest

```
data_dict = mdr.query(query_type=Action).filter_by(Action == "tightening",
                                                   Tool == mytool).get()
```

```
[INFO] Read annotation: 100%|████████████████████████████████████████████████████████████| 16/16 [00:01<00:00, 10.90it/s, file=data-03.annotation]
[INFO] Finished with 79 action(s).
```

[12]:
```
data_dict.keys()  # the data is now grouped by actions
```

[12]:
```
dict_keys(['tightening'])
```

[13]:
```
data_bunches = data_dict["tightening"]
print(f"{type(data_bunches)} of {type(data_bunches[0])}")
```

```
<class 'list'> of <class 'datatools.data.DataBunch'>
```

Create windowed Time-Series: seglearn

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

dmbee.github.io/seglearn/

1.3. 2. Create windowed Time-Series

[14]:
```
import numpy as np
from seglearn.base import TS_Data
from seglearn.pipe import Pype
from fhgutils import Segment, contextual_recarray_dtype, filter_ts_data
from datatools import to_ts_data
```

> Note: `to_ts_data` can deal with the output when querying *Measurements* as well as *Actions*

Have a look on the seglearn user guide to understand the data representation with `Xt` and `Xc`.

[15]:
```
window_length=0.4 # unit in s
overlap = 0.5 # unit in percent

# cl
data_
Xt, )

X = 
pipe
    (


])

X_trans, y_trans = pipe.fit_transform(X, y)
```

[34]:
```
print("[INFO] segmented data")
print(len(X_trans))
for i in range(0,4):
    print(X_trans[i].shape, y_trans[i].shape)
```

```
[INFO] segmented data
29060
[41] (41,)
[41] (41,)
[62] (62,)
[3200] (3200,)
```

Filter multimodal data for sensor modalities

Important: discard the TIMESTAMPS ☺

1.3.1. 2.1 Filter windowed Data

[35]:
```
print("[INFO] extract segmented ACC data")
Xt_acc, Xc_acc, y_acc = filter_ts_data(X_trans, y_trans, filt={'desc': ['acc']})

print("[INFO] extract segmented GYR data")
Xt_gyr, Xc_gyr, y_gyr = filter_ts_data(X_trans, y_trans, filt={'desc': ['gyr']})
```

```
[INFO] extract segmented ACC data
[INFO] extract segmented GYR data
```

[36]:
```
print("[INFO] shape of ACC data is", Xt_acc.shape)
print("[INFO] shape of GYR data is", Xt_gyr.shape)
print(f"[INFO] {Xt_acc.shape[0]} windows with {Xt_acc[0].shape[1]} sample each (ti
```

```
[INFO] shape of ACC data is (7265,)
[INFO] shape of GYR data is (7265,)
[INFO] 7265 windows with 4 sample each (time stamps and 3 axis accelerometer)
```

You can discard the time column with e.g.:

```
Xt_acc = Xt_acc[:,:,1:]
```

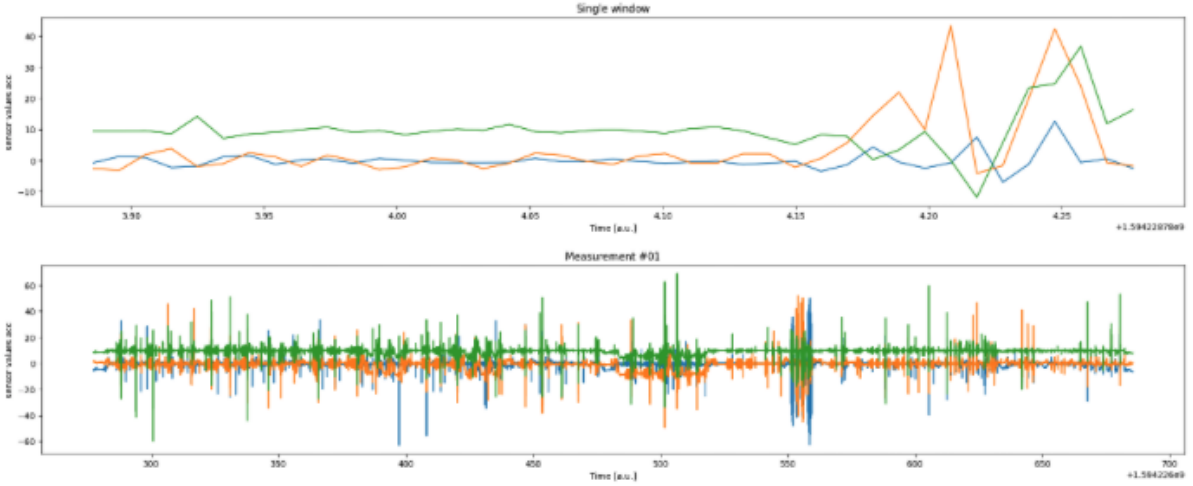
Plot the data

2.2 Plot the Data

[37]:
```
import matplotlib.pyplot as plt

# plot a sample
plt.figure(figsize=(24,4))
plt.title(f"Single window")
t = Xt_acc[70][:, 0]
x_win = Xt_acc[70][:, 1:]
plt.plot(t, x_win)
plt.xlabel('Time [a.u.]')
plt.ylabel('sensor values acc')
plt.show()

# plot a measurement
plotme = data_dict["01"].acc.ts
t = plotme[:, 0]
x_mea = plotme[:, 1:]
plt.figure(figsize=(24,4))
plt.title("Measurement #01")
plt.plot(t, x_mea)
plt.xlabel('Time [a.u.]')
plt.ylabel('sensor values acc')
plt.show()
```



Data Preprocessing: filter out specific points by label

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

1.4. 3. Preprocess the Data

[19]:
```
from fhgutils import filter_labels, one_label_per_window, summarize_labels
```

1.4.1. 3.1 `filter_labels`

- filter out labels [-1]
- also filter out whole windows, in case there is no majority label
- e.g. window length of 5, labels could be [1,1,1,1,0] -> okay. but if [1,1,0,0,2] -> discard window.

[20]:
```
values, counts = np.unique(y_acc, return_counts=True)
for val, count in zip(values, counts):
    print(f"[INFO] label {val} with {count} samples")
```

```
[INFO] label 2.0 with 33998 samples
[INFO] label 3.0 with 14588 samples
[INFO] label 4.0 with 12588 samples
[INFO] label 5.0 with 5614 samples
[INFO] label 6.0 with 3097 samples
[INFO] label 7.0 with 3672 samples
[INFO] label 8.0 with 223675 samples
[INFO] label 14.0 with 633 samples
```

[21]:
```
Xt_acc_f, Xc_acc_f, y_acc_f = filter_labels(labels=[-1], Xt=Xt_acc,
                                            Xc=Xc_acc, y=y_acc)
```

```
[INFO] original Xt: 7265 (41, 4)
[INFO] filtered Xt: 7259 (41, 4)
```

Flatten labels

Uses majority vote to determine window-level labels

- For further label conversion, see OneHotEncoder in scikit-learn

1.4.2. 3.2 `one_label_per_window`

- flatten labels of windows to the majority label
- `[1,1,1,1,2]` -> 1. bad if you've got `[0,0,0,0,0,1,1,1,1,1,1]` -> 1, this creates anomalous samples of class 1

[22]:
```
print("pre", y_acc_f[5])
y_acc_f = one_label_per_window(y=y_acc_f)
print("post", y_acc_f[5])
```

```
pre [2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2.
 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2.]
flattened 8 labels: [ 2  3  4  5  6  7  8 14]
post 2
```

Summarize labels

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

You may combine annotations into a new class to augment the dataset from real data.

1.4.3. 3.3 `summarize_labels`

- some labels are basically very similar, like "pull_trigger, pull_trigger_air" for the Pneumatic Rivet Gun
- same action, but once holding the tool in the air, once using it on a workpiece.

[43]:
```
summarylabels = {5: [0, 1]}
y_acc_f_sum = summarize_labels(y_acc, summarylabels)
```

```
[INFO] Summarized labels from [ 2.  3.  4.  5.  6.  7.  8. 14.] to [ 2.  3.  4.  5.
6.  7.  8. 14.]
```

[44]:
```
X_trans.shape
```

[44]:
```
[29060]
```

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

Summary

Multi-modal
Tool Tracking Dataset

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO

Summary

- Materials
- Motivation
- System description
  - Hardware platform
  - Sensors
- Hand-held tools
  - Tools
  - Important sensors
- Time-Series Dataset
  - Tools
  - Recordings
  - Annotations
  - Folder structure and data
  - Data loader