Technology by

Software for Printed Sensor Matrices

SensorMatrixLAB

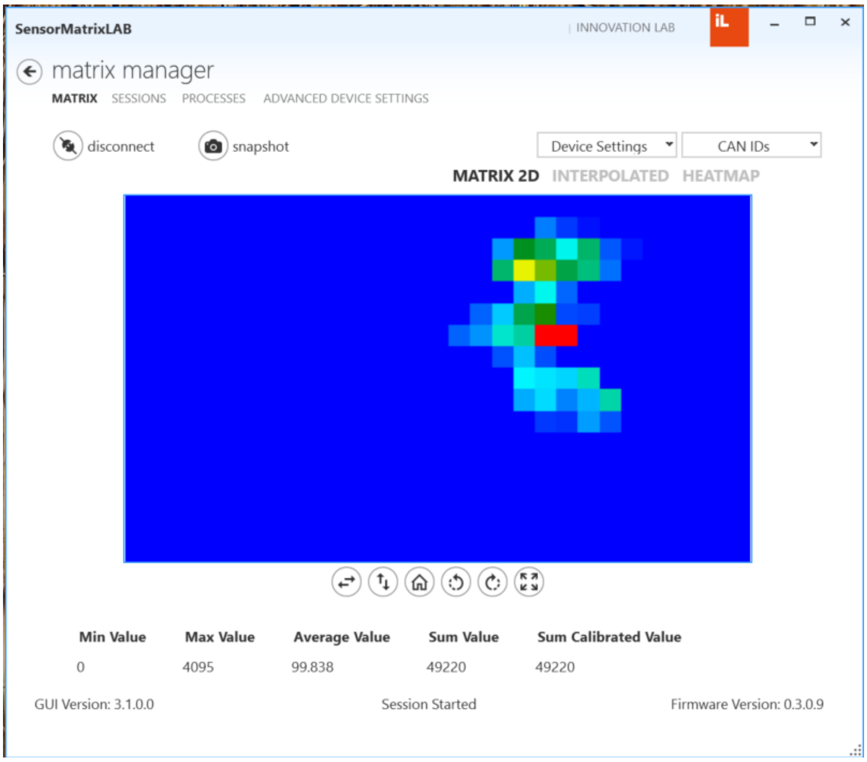
User manual

Table of Contents

General information

The “SensorMatrixLAB” software is a Microsoft Windows application developed to visualize and process the data from InnovationLab’s Pressure Sensor Matrixes. The software supports following main features:

- Advanced data visualization,
- High speed data recording and replaying,
- API sensor data access,
- Communication Interfaces: USB Serial, Wi-Fi, Ethernet,
- Support of customized printed sensor matrixes,
- Advanced hardware configuration,
- Configuration of readout electronics, including standalone operation via CAN,
- Full proprietary protocol support.



Main view of SensorMatrixLAB software

Contact information

The correspondence related to the current manual as well as any questions related to the software should be addressed to:

InnovationLab GmbH
Speyerer Straße 4
69115 Heidelberg, Germany

T +49 6221 - 54 19 100
F +49 6221 - 54 19 110
E info@innovationlab.de

Hardware requirements

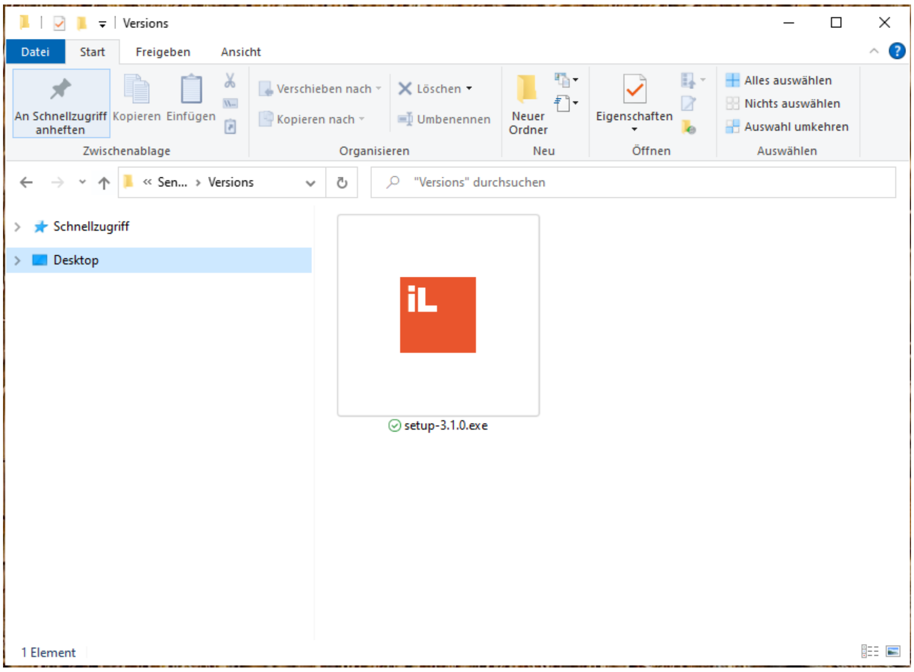
- InnovationLab's hardware (electronics) with a latest firmware version
- InnovationLab's printed sensor matrix
- Personal computer (Windows-based) with x86/x64 processor
- USB port, USB-to-serial driver installed
- Internet connection
- Optional: Connection to local network via Ethernet (for ETH connectivity)

Software requirements

- Microsoft Windows 10 or higher
- Microsoft .NET framework 4.7.2
- Web browser (Mozilla Firefox, Google Chrome, or Microsoft Edge) of latest version
- Microsoft Excel (2016 or higher)

Installing the Software

To install a software on your PC running Windows 10 or higher you should run the received executable installer file:

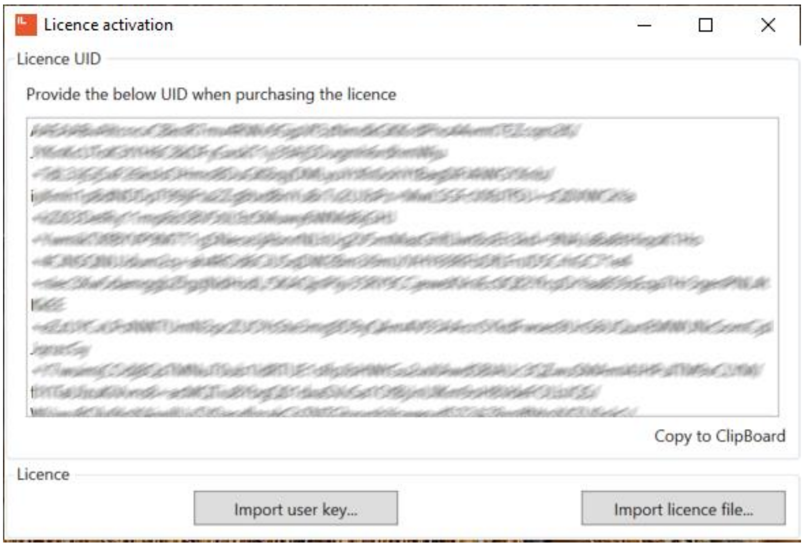


*Locating the Installer file *.exe in Microsoft Windows*

Note that SensorMatrixLAB does not require a local user to have administrator privileges on the machine to install or use the software. However, administrator privileges might be needed to use auto-discovery function.

Activating the Software

After installation the software should be activated. To do so, run the software using a shortcut. You will be prompted to activate.

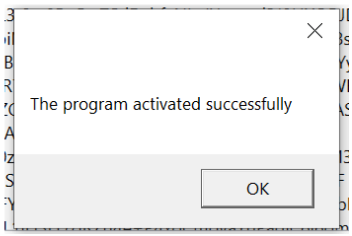


Activation Window

Copy your *License UID* into the clipboard and request a license via InnovationLab's web site. For the activation you will get two files:

- User key
- License file

To activate, **import the user key** in activator, then **import the license file.** If all operations were done correctly, you will see the confirmation:



Congratulations 👏 the license activation is done!

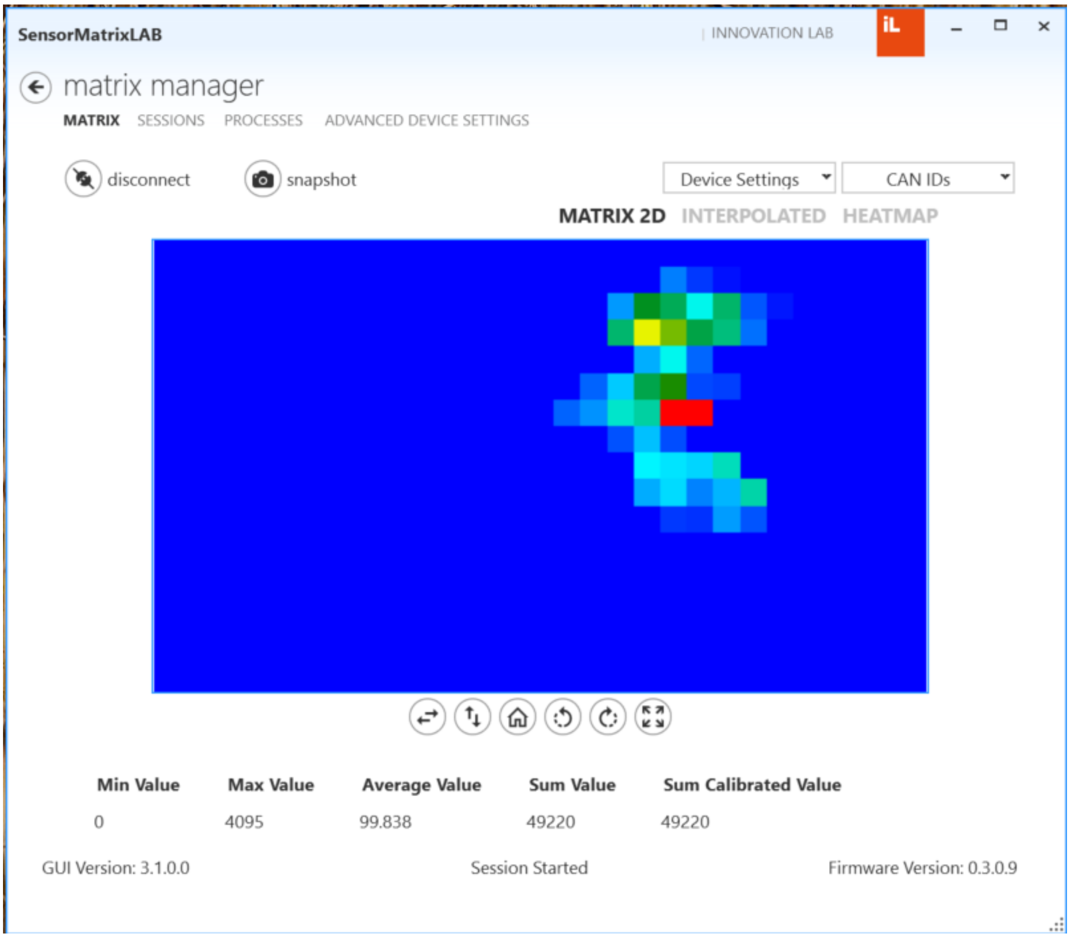
Updating the Software

In order to update SensorMatrixLAB to a newer version follow these steps:

1. Uninstall current version from your PC
2. Download a latest version from InnovationLab's web site
3. Install new version and perform activation same as for first installation.

The user interface of the SensorMatrixLAB software

Figure below shows the main window of the "SensorMatrixLAB" software with its main elements:



SensorMatrixLAB software - session started

Main User Interface Elements

Settings - global settings of an application:

- Menu "Connection"
- Menu "Frame Processing"
- Menu "View"

- Menu “Storage”
- Menu “API Access”
- Menu “Appearance”

Device Settings

Session settings of the hardware

CAN Settings

Settings for Controller Area Network (CAN bus)

Tabs

- Matrix: represents a color-coded 2-dimentional view of sensor data
- Sessions: contains previously captured sessions
- Processes: information about background threads related to Software
- Advanced Device Settings: Time-domain filtering, Wi-Fi, In-board frame processing

Content View

Element is used for visual demonstration of the live sensor data or recorded sessions

App and Firmware version

Version of the application and firmware

Connect / disconnect

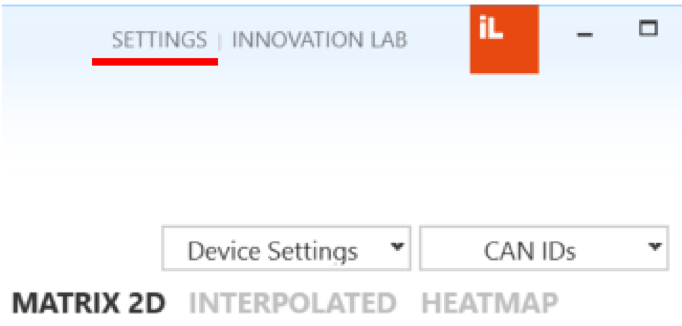
Button for starting and stopping a session

Frame statistics

Current sensor frame information

Settings

Global settings concerning an application and connection to hardware can be changed under ***SETTINGS***, top right corner of main window.



SensorMatrixLAB software - Settings

General application settings are saved in local user folder. Categories of settings are listed in sub-sections below.

Menu "Connection"

The *CONNECTION* section allows to select between connection interfaces:

- USB connection
- TCP/IP

Not all connection methods are supported by each hardware. Refer to hardware manual for more information.

The **USB connection** is used as a virtual serial port. To use it, a proper driver installation is required on your Windows machine. Contact your system administrator to install proper drivers.

In case of USB connection, the proper port should be selected. Refer to *Windows Device Manager*, section *Ports* for available devices.

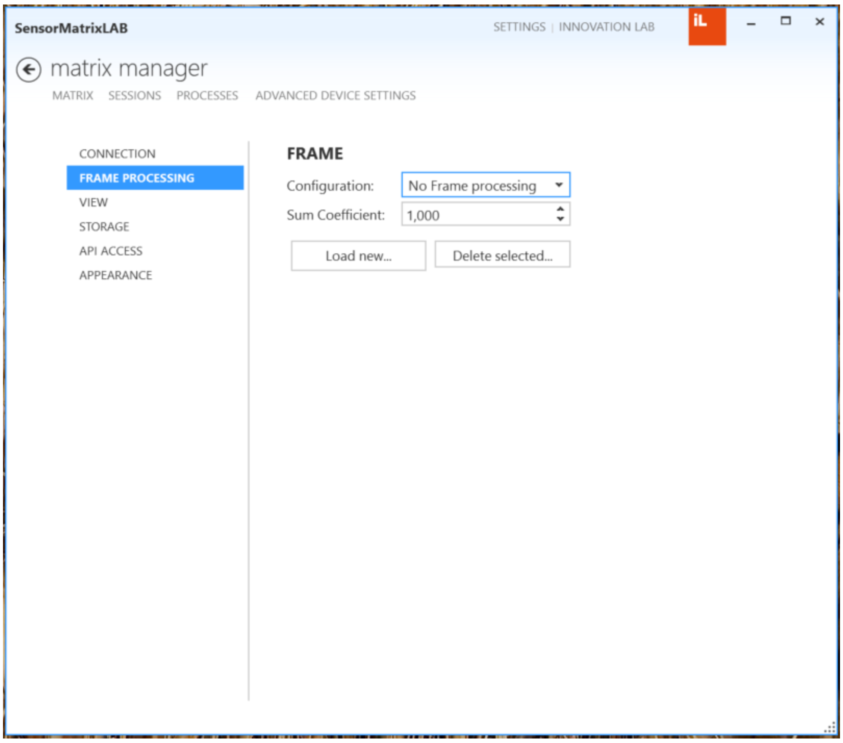
The USB baud rate should correspond to used hardware. Refer to hardware manual for this value. For most InnovationLab's boards `baudrate = 1000000`.

The **TCP/IP connection** enables data transfer from the board with Ethernet or Wi-Fi. For correct operation the hardware should be in the same network, as the Windows machine. At the same time, correct hardware *IP address* and *port* should be specified. Use **Scan** button for automatically acquiring this information. *IP address* and *Port* of the *Hardware* can be also found in your router in connected devices list, and then input manually. If you need an information regarding hardware *IP address* and *Port*, contact your system administrator.

Note that for using the Wi-Fi connection, the prior hardware configuration is necessary. Refer to section **Advanced Device Settings** for more details.

Menu "Frame Processing"

The *Frame Processing* section allows configuring the frame representation from sensor data, coming from the board.



Settings – menu "Frame Processing"

Configuration - corresponds to the type of Printed Sensor Matrix attached to the hardware. The selection of this fields will impact how the data stream coming from your Hardware will be converted into a 2D visual representation in the Software. The

PC rearranges the byte order for proper visualization. This option is implemented to support different types of sensor matrixes on the same hardware. The selected "Configuration" should match with the sensor type used.

The list of available Configurations is determined by the content of `frame.config` file, located in:

`C:\Users\{user}\AppData\Roaming\INNOVATION LAB\SensorMatrixLAB\.`

If option "**No Frame Processing**" is selected in Frame/Configuration, the data stream will not be processed on the PC side and the frame will be visualized as received from the board. The matrix size for visualization will be taken as Width and Height as specified in Device Configuration.

Use "**No Frame Processing**" option in case that raw sensor data without rearrangement is needed or in-board rearrangement is used.

If the proper Configuration for Printed Sensor Matrix cannot be found, contact InnovationLab GmbH. Alternatively, build your own `frame.config` file and add it as described below. Refer to **Appendix B. JSON config file structure** for more information.

Note that **Hardware-side rearrangement** and **PC-side rearrangement** can interfere each other. Use only one type of rearrangement at the same time.

Sum Coefficient - coefficient used for calibrating sensors. The *Sum Calibrated Value* is taken as:

*Sum Calibrated Value = Sum Coefficient * Sum Value.*

Frame statistics including *Calibrated Sum* is shown in main window when session is running:

Min Value	Max Value	Average Value	Sum Value	Sum Calibrated Value
0	315	0.52	676	676

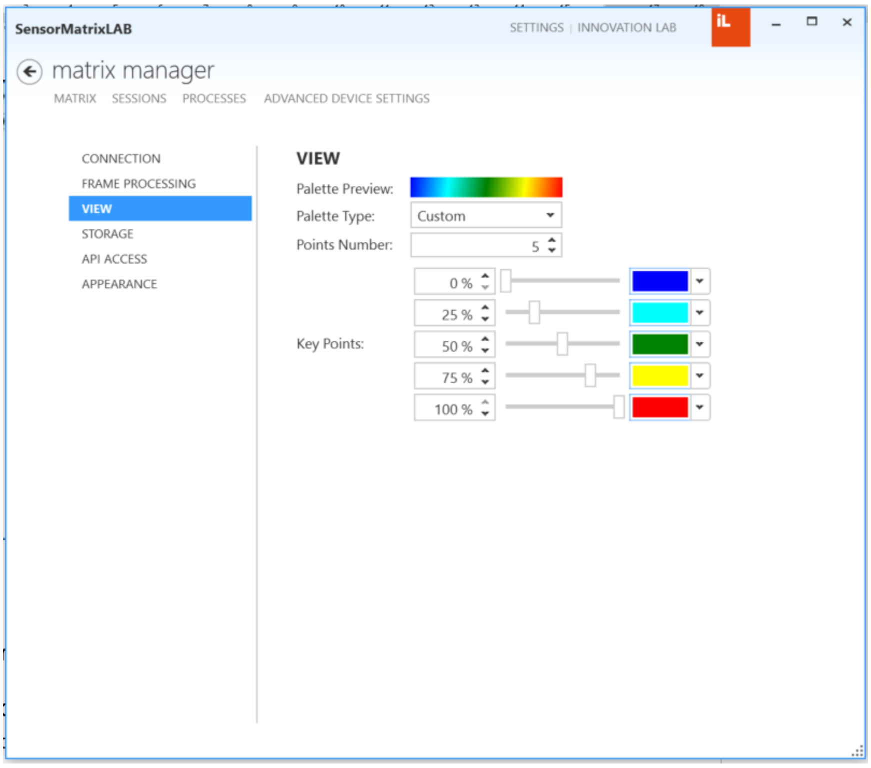
SensorMatrixLAB – frame statistics

User can add new configurations into the SensorMatrixLAB software to support new matrix types. To do so, use the **Load new** button. Using load file dialog, navigate to your *.json file, containing Frame configuration received from InnovationLab. Alternatively it is possible to prepare the *.json Frame configuration according to **Appendix B. JSON config file structure**. If the loaded file is correct, the form will ask which of the found configurations should be added. If configuration with the same name already exist, a warning will be shown. After configuration is loaded, select it in the list to apply.

When needed, the configuration can be deleted with **Delete selected** button.

Menu "View"

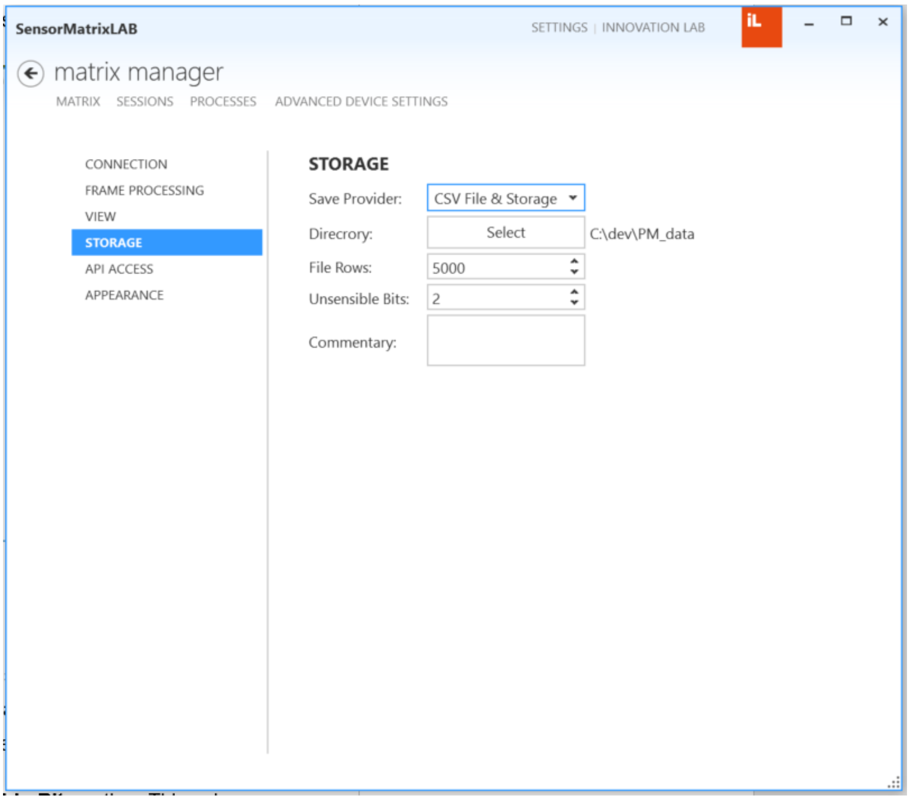
The *VIEW* section allows to select and configure a color map for data presentation in the main window.



Settings – menu “View”

Menu “Storage”

The *STORAGE* section allows saving of a session in two ways: *CSV* (csv file) or *Database* (storage file). Selection of these modes can be made under **Save Provider.** Make sure that a **Directory** is set. Both data types will be saved in this path.



Settings – menu "Storage"

To prevent unnecessary large **.csv* files, it is possible to split a session into several files. **File Rows** defines the maximum number of rows in a single **.csv* file. One row is presenting a single frame. If this amount is reached, a new **.csv* file will be created.

It is also possible to reduce data size using the **Unsensible Bits** option. This value represents a range where the sensor values can vary without being saved. This can be used for long time measurements with not much action and will prevent that insignificant noise will be saved.

The data in **.csv* files is arranged in following format:

Column	1	2	3	4 and following
Description	Time Stamp (in ms)	Sample Number	CAN Timestamp (if configured)	Data of sensor pixel

A text field **Commentary** can be used to add a text string to the captured data.

Additionally to data itself, a *session.txt* or *session.db* file will be generated, containing important information about session. An example file can be seen below (Listing 1).

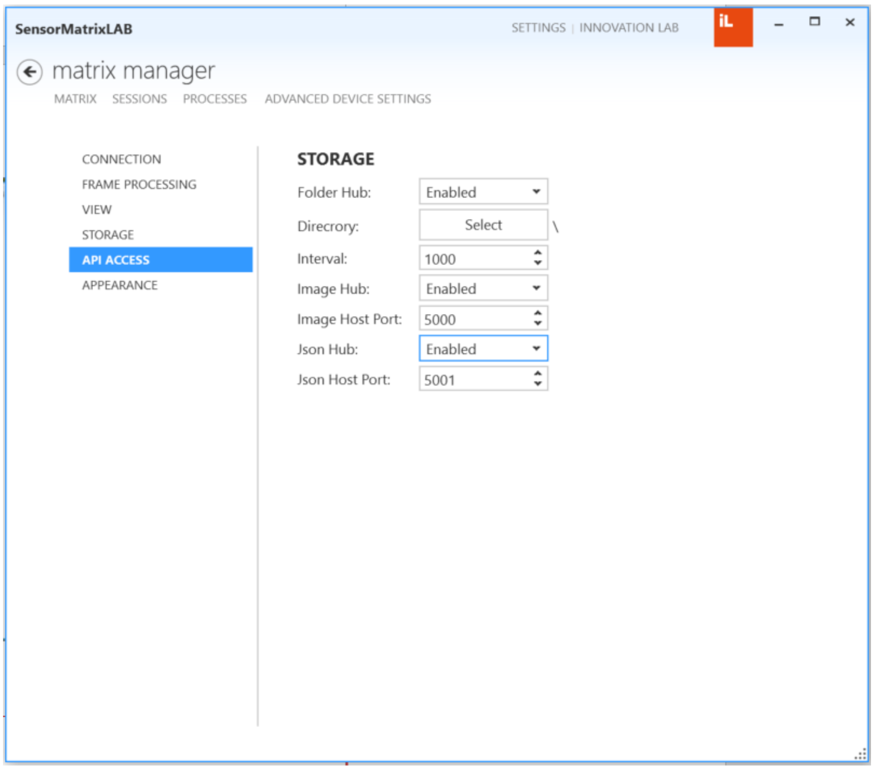
Listing 1 – Session description file session.txt

```
SESSION
Start Time:              2021-06-07 13:12:28
Stop Time:               2021-06-07 13:12:33
Session Time:            5,18 seconds
Received Samples:        97
Saved Samples:           97
Frequency:               18,72 Hz


SETTINGS
Frame Configuration:     65x20pix_530x105mm
Frequency:               unlimited
Switch-read Delay:       0 us
Number of Samples:       unlimited
CAN Timestamp:           1970-01-01 00:00:00
Firmware version:        0.2.6.9
GUI version:             3.0.1.0
Unsensible Bits:         2
Commentary:
```

Menu “API Access”

The **API Access** option enables live data integration from InnovationLab’s software into other programs on the same PC. Integration options available are described below.



Settings – menu "API Access"

PNG-Image, interval-based generation

In this mode, grayscale PNG images are generated when session is running. The data is written into a selected **Directory** with a selected **Interval** (in ms). To turn on this mode, use **Folder Hub** setting.

Data generated in this mode can be viewed with any Windows based graphics viewer/editor, e.g. MS Paint. Each pixel of the created graphics is corresponding to the pixel of sensor matrix, and the luminosity contains information about sensor value.

The following configuration is used for PNG encoding (**Listing 2**):

Listing 2 – configuration of PNG encoding:

```
ColorType = Grayscale;
BitDepth = PngBitDepth.Bit16;
CompressionLevel = PngCompressionLevel.NoCompression;
FilterMethod = PngFilterMethod.None;
InterlaceMethod = PngInterlaceMode.None;
```

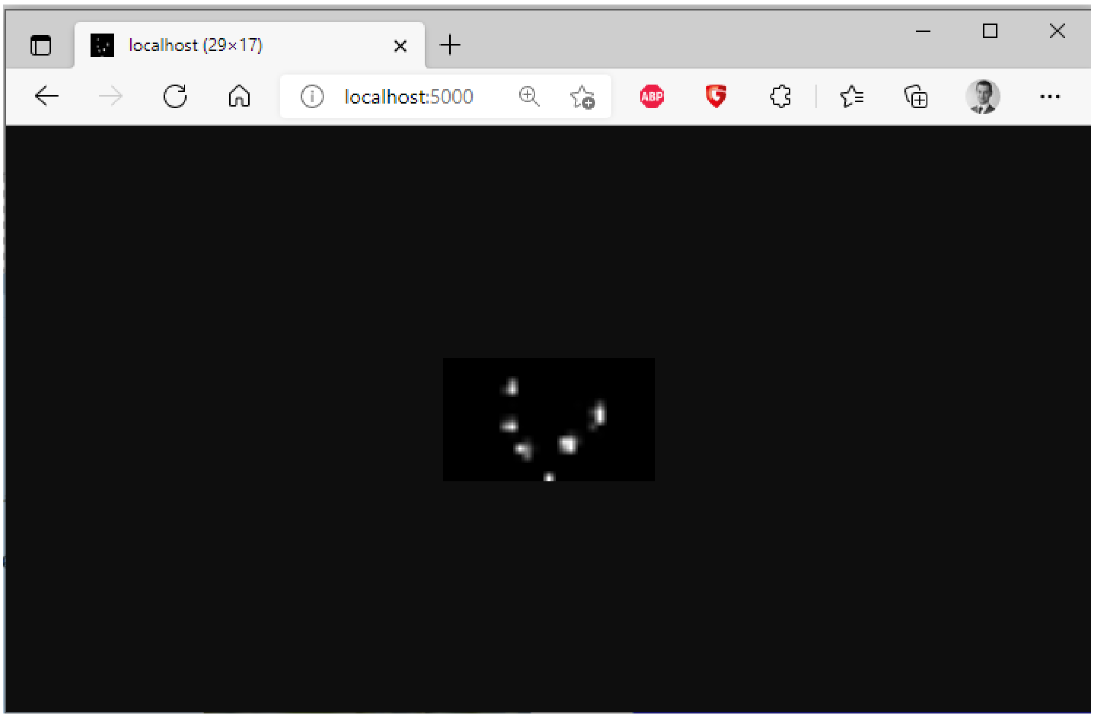
To ensure the high contrast in graphics viewer program, the 12-bit sensor data in 16-bit PNG is aligned left. This means, that 12 most significant bits of 16-bit luminosity values are used and remaining 4 bits can be ignored. To get sensor raw values from saved PNGs, luminosity values be shifted 4 bits right (**Listing 3**):

Listing 3 – example PNG decoding in C# using *SixLabors.ImageSharp library*

```
for (var i = 0; i < width; i++)
 {
     for (var j = 0; j < height; j++)
     {
         image[i, j] = new L16((ushort)(PNG_value[i, j] >> 4));
     }
 }
```

PNG-Image, API access

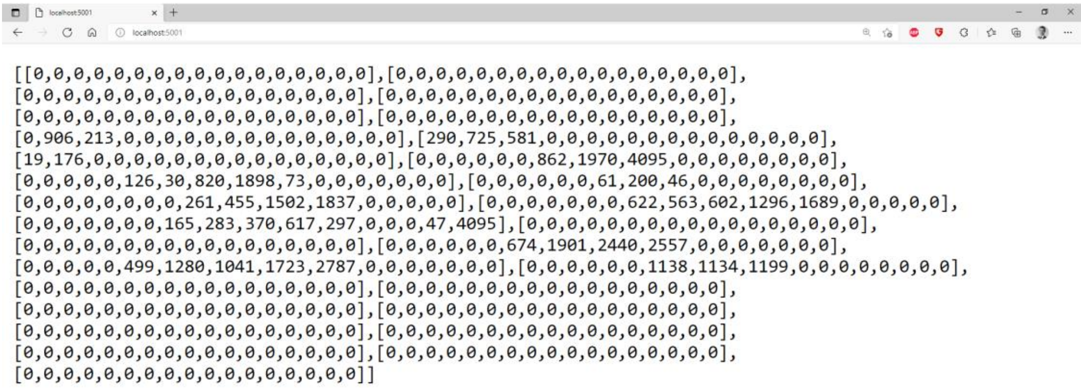
In this mode, the matrix state is generated as a *.png image. This happens on demand upon incoming HTTP requests. Images can be loaded using the HTTP GET request via the selected **Image Host Port**. The data format is the same, as in interval-based generation. To turn on this mode, use **Image Hub** setting.



Example PNG file containing matrix data loaded via Web browser from localhost:5000

JSON file, API access

In this mode, the matrix state is generated as JSON file. This happens on demand upon incoming HTTP requests. The data can be loaded from a third-party software using the HTTP GET request via the selected **JSON Host Port**. To turn on this mode, use **JSON Hub** setting.

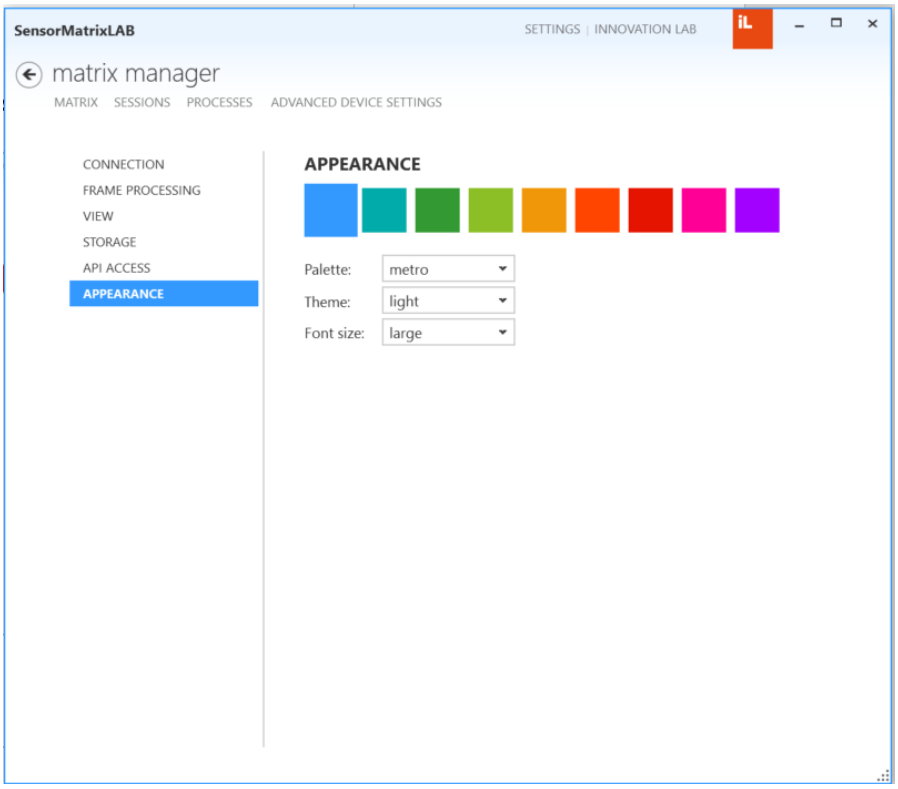


[[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[0,906,213,0,0,0,0,0,0,0,0,0,0,0,0,0,0],[290,725,581,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[19,176,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,862,1970,4095,0,0,0,0,0,0,0,0],
[0,0,0,0,0,126,30,820,1898,73,0,0,0,0,0,0,0],[0,0,0,0,0,0,61,200,46,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,261,455,1502,1837,0,0,0,0,0],[0,0,0,0,0,0,0,622,563,602,1296,1689,0,0,0,0,0],
[0,0,0,0,0,0,0,165,283,370,617,297,0,0,0,47,4095],[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,674,1901,2440,2557,0,0,0,0,0,0,0],
[0,0,0,0,0,499,1280,1041,1723,2787,0,0,0,0,0,0,0],[0,0,0,0,0,0,1138,1134,1199,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]]

Example JSON file containing matrix data loaded via Web browser from localhost:5001

Menu “Appearance”

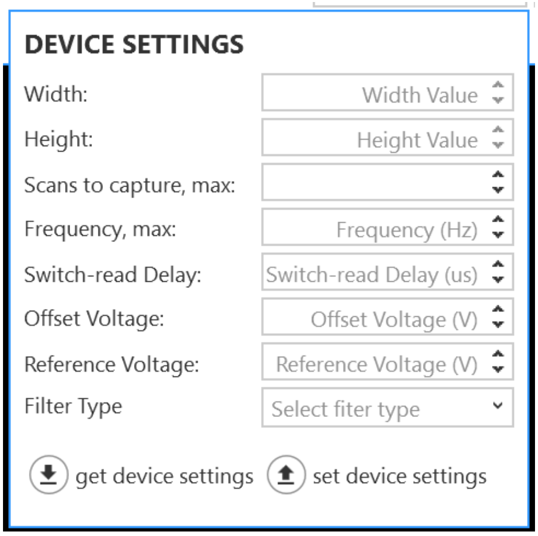
All settings about the app style can be found in the **APPEARANCE** section.



Settings – menu "Appearance"

Device Settings

The **Device settings** are saved in the non-volatile memory of the hardware and the same settings will be used for controlling the device via all supported interfaces (e.g. CAN, ETH). Note that to update the device configuration, an existing connection to the hardware is required and session must be stopped.



Device Settings

Providing that the USB connection to the PCB is successfully established, PCB settings can be changed in main window under *Device Settings.* The **get device settings** button loads settings from the board, **set device settings** sends user's input to the board.

- **Width** and **Height** describe the scan area size that will be used during the session. These values are *read-only*. Width and Height of the scan are corresponding to the maximum resolution of the hardware in case in-board frame converter is *off* or to actual selected frame size in case in-board frame converter is *on*. Refer to hardware datasheet for more information.
- **Scans to capture, max** makes it possible to limit the number of captured frames of each session. After this number is reached, session will be stopped. If this value is set to 0, session is not limited and can only be stopped using *disconnect* button.
- **Frequency, max** describes the upper limit for the measurement frequency of a session in Hz (samples per second). If this value is set to 0, measurement will be done with maximum frequency possible for current hardware and sensor matrix. Note that some hardware does not support frequency limitation. Refer to hardware datasheet for more information.
- **Switch-Read Delay** allows to set a delay between multiplexing and reading each matrix pixel. This value might influence the image quality and provides control over timings. The unit of this value is µs.

- **Voltage Offset** allows to set an offset voltage value. The unit of this value is Volts.
- **Voltage Reference** allows to set a reference voltage value on the board. In general, the higher the reference, the higher is the sensitivity of the sensor system to external stimulus. It is recommended to start measurements with low reference voltage and increase it when needed. The unit of this value is Volts.
- **Filter Type** allows applying time-domain filtering on live sensor data. This means that each matrix pixel is filtered as a function of time, independently from other pixels. Note that not all hardware support live filtering. For more information refer to the manual for corresponding hardware.

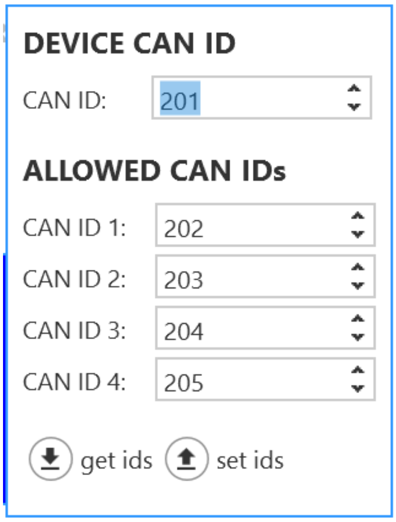
Refer to **Appendix A. Measurement principle** for more information.

- If get/set settings buttons are not functioning, check the connection settings under ***SETTINGS*** - ***CONNECTION***.
- Download and flash the latest Firmware Version into the electronics.

CAN Settings

Section CAN settings are related to communication over Controller Area Network (CAN bus). In order to get or set these settings in hardware, a working USB / TCP connection is required. Following parameters can be set:

- CAN ID – defines the ID of the hardware. Default value is 201.
- CAN ID x - defines the CAN IDs of devices whose commands will be accepted by the hardware.



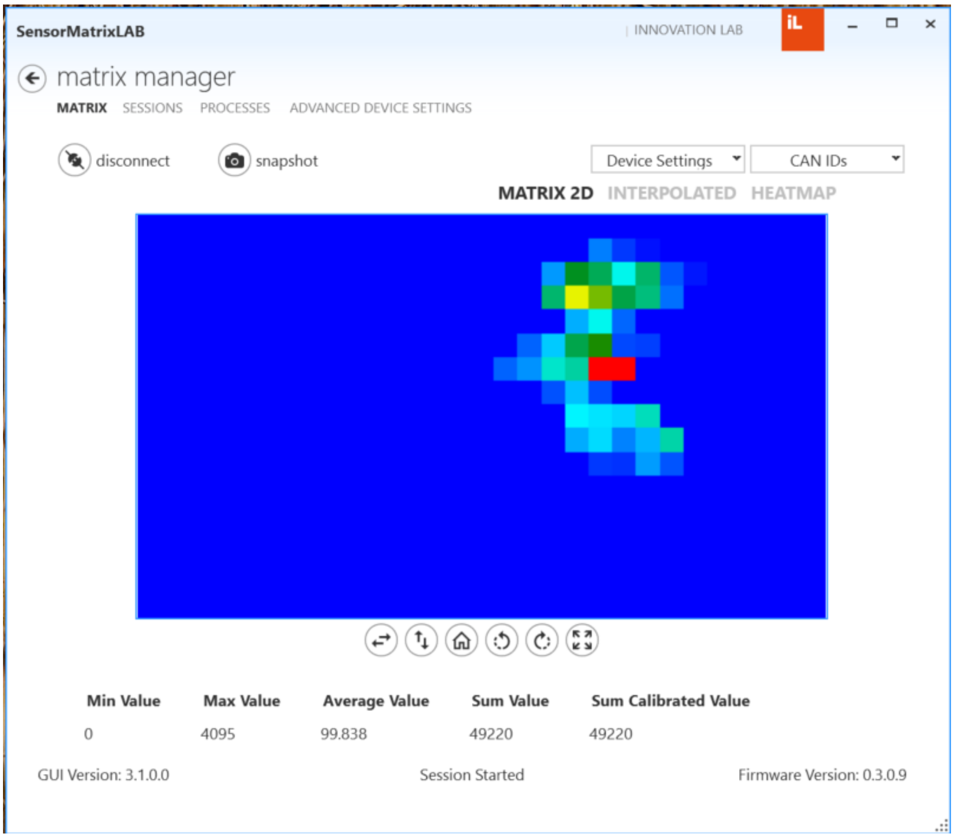
CAN Settings

Tabs

The tabs of main window provide access to the core functionalities of SensorMatrixLAB – visualizing and saving sensor data as well as advanced board configuration. Refer to the sections below for detailed explanations of each tab.

Matrix

This tab contains live sensor data visualized as a heatmap.

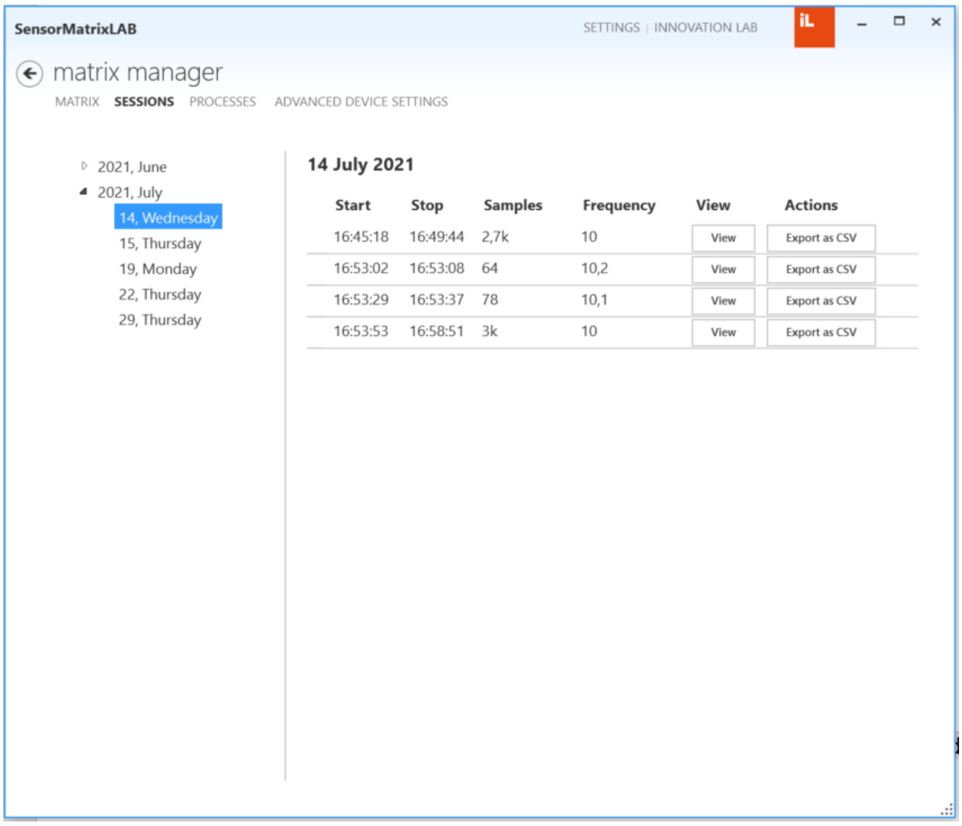


SensorMatrixLAB – tab «Matrix» (2D view)

Additional sensor data representations are available on this tab: 2D raw, interpolated, heatmap.

Sessions

This tab contains information about previously recorded sessions. Note that only sessions saved in database format are displayed. To select a folder for storing sessions, use **Settings - Storage** menu, field **Directory**.

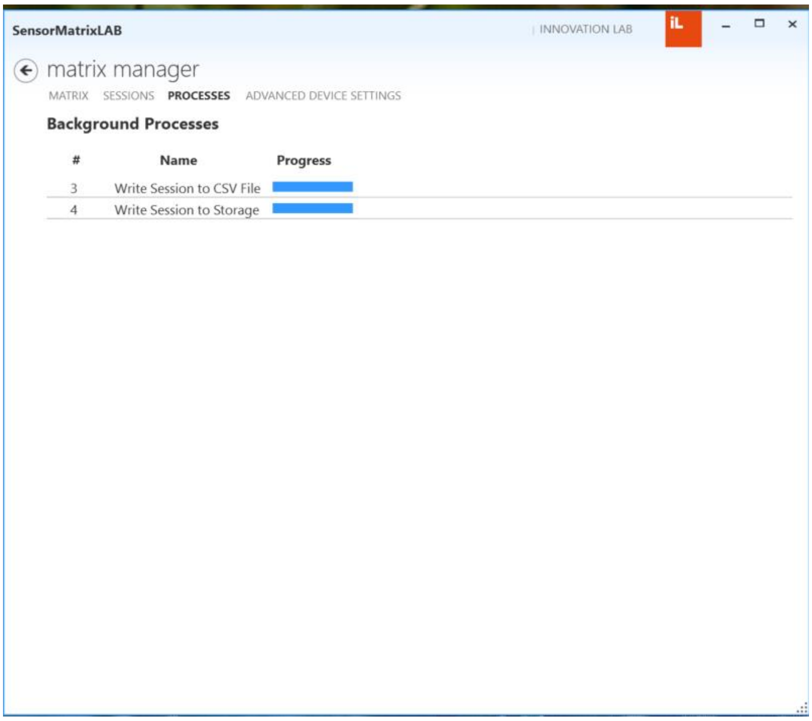


SensorMatrixLAB – tab “Sessions”

From this tab, the user can play previously recorded sessions in an embedded player or export the selected session into CSV format.

Processes

This tab contains a list of subprocesses created by SensorMatrixLAB, such as database writer, CSV writer or other. The **Progress** bar represents the current load of the process. Note that the performance of subprocesses can be affected by the accessibility of saving destination. In this case, please consider using another destination **Directory** in **Settings - Storage** menu.

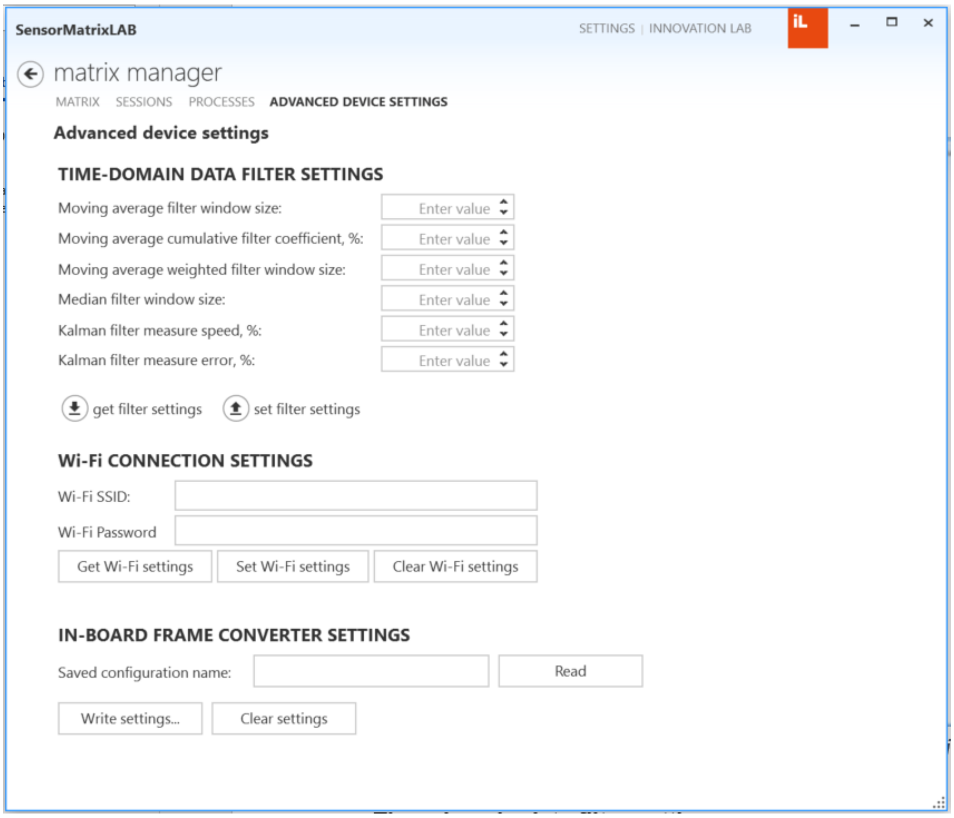


SensorMatrixLAB – tab "Processes"

Advanced Device Settings

The tab **Advanced Device Settings** allows the user to configure advanced in-board functions:

- Hardware-side time-domain filters
- Hardware wi-fi connection
- Hardware-side frame rearrangement



SensorMatrixLAB – tab "Advanced Device Settings"

Time-domain data filter settings

In this section, the user can configure the time-domain filtering of sensor data. This means, each pixel is processed individually without being taking into account the data generated by its neighbors. Following filter algorithms are available:

- moving average
- moving average cumulative
- moving average weighted
- median filter
- Kalmann filter

Selection of a filter type and its configuration type should be based on concrete sensor usage scenario.

Wi-Fi connection settings

These settings allow to configure the Wi-Fi Access Point (AP), which will be created by the readout board. The PC with running SensorMatrixLAB should connect to this AP. When Wi-Fi connection is successfully established, the session can be started over TCP/IP, without the need for a USB connection.

Note that for Wi-Fi AP configuration an existing USB/Ethernet connection is mandatory.

To get current Wi-Fi AP connection configuration, use the **Get Wi-Fi settings** button. The data is then loaded from the board. To update values in **Wi-Fi SSID** and **Wi-Fi Password** fields and upload into the hardware, use the **Set Wi-Fi settings** button.

When Wi-Fi configuration is applied, SensorMatrixLAB software can connect to the hardware over Wi-Fi network created by AP by selecting TCP/IP **Connection type** in **Application Settings.** The **IP address** should be set to 192.168.0.1 and **Port** 1000 should be selected.

In-board frame converter settings

The frame rearrangement allows to use the same hardware for reading different printed matrixes with correct data visualization in UI. The in-board frame converter allows frame re-arrangement directly in the Hardware, not on PC side. Thus, the data stream from the hardware can be used afterwards without SensorMatrixLAB, e.g. in embedded applications.

Note that **Hardware-side rearrangement** and **PC-side rearrangement** can interfere with each other. Therefore, please use only one type of rearrangement at the same time.

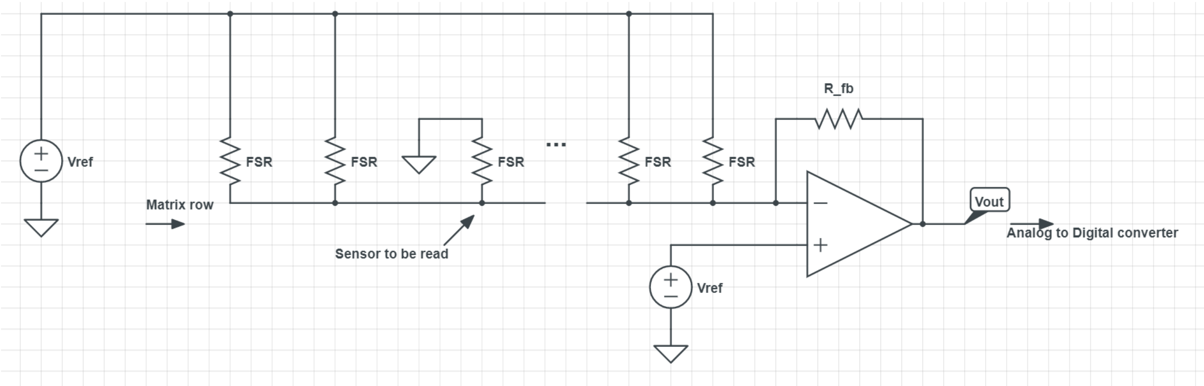
To get the current in-board frame converter settings use **Read** button. The name of current configuration will be shown in the text field. If **No table** response is received, there is no in-board decoding enabled.

Use the **Write settings** button to apply one of existing frame configuration into the board. When configuration is written into the board, the PC-side frame processing is automatically turned off. To add/delete configurations into available configuration list, refer to section **Settings - Menu “Frame”.**

To disable frame processing in the hardware and erase decoding matrix use **Clear Settings** button.

Appendix A. Measurement principle

InnovationLab's crosstalk-less readout electronics use the following schematics for acquiring data from piezo-resistive printed sensor matrices:



Measurement principle of crosstalk-less readout electronics. A single matrix's row is shown.

From this circuit output voltage $\mathrm{V}_{\mathbf{out}}$ can be derived as:

$$\mathrm{V}_{\mathbf{out}} \approx \mathrm{V}_{\mathrm{ref}} + \mathrm{V}_{\mathrm{ref}} \cdot \frac{R_{fb}}{\mathrm{FSR}},$$

where $\mathrm{V}_{\mathrm{out}} \in [\mathrm{V}_{\mathrm{ref}}, \mathrm{V}_{\mathrm{Supply}}]$. Before being forwarded to the input of the Analog to Digital converter, a $\mathrm{V}_{\mathrm{ref}}$ is subtracted from the output $\mathrm{V}_{\mathrm{out}}$, resulting in a following formula for sensor signal:

$$\mathrm{V}_{\mathbf{adc}} \approx \mathrm{V}_{\mathrm{ref}} \cdot \frac{R_{fb}}{\mathrm{FSR}},$$

$\mathrm{V}_{\mathbf{adc}} \in [0, \mathrm{V}_{\mathrm{Supply}} - \mathrm{V}_{\mathrm{ref}}]$, where $\mathrm{V}_{\mathrm{ref}} \in [0, 2.5\mathrm{V}]$ and $\mathrm{V}_{\mathrm{Supply}} \in [0{,}5\mathrm{V}]$.

As the result of using this circuit, following performance parameters are achieved:

1. No crosstalk between different sensors in a matrix,
2. Sensitivity of the matrix is adjustable: the higher $\mathrm{V}_{\mathrm{ref}}$, the higher the sensitivity.

Appendix B. JSON config file structure

The example config file below (**Listing B1**) contains a description for General 12x20 force-sensitive matrix. More `Configurations` can be included into a single JSON file to support more matrixes.

For the selected matrix the layout is done in following way:

- Matrix has total **physical** resolution of 12x20 pix
- First 6 **physical** vertical lines are connected to even **input lines of hardware**, ascending (from 0 to 12, increment 2)
- Last 6 **physical** vertical lines are connected to odd **input lines of hardware**, ascending (from 1 to 11, increment 2)
- First 10 **physical** vertical lines are connected to first 10 **input lines of hardware**, descending (from 9 to 0, increment 1)
- Last 10 **physical** vertical lines are connected to last 10 **input lines of hardware**, ascending (from 10 to 19, increment 1)

To describe the matrix the JSON file contains following sections:

`Name` - name of the configuration, as displayed in UI.

`FrameBits` - ADC resolution per pixel, defined by hardware capabilities.

`PCM` - decoding table, used to matrix transformation from initial 2D-byte image into visualization-ready representation. The size of `PCM` array correspond to target visualization matrix. In the example below the `PCM` has s a size of 12x20. Each pixel has 2 values attributed to it, defining at which coordinates in initial 2D-byte representation the pixel data will originate. For example, in **Listing B1** below for **[0,0]** pixel in visualization-ready representation the coordinates are **[0,9]**. This means, that for value of pixel **[0,0]** in target visualization matrix will be taken value of pixel **[0,9]** of input lines of hardware.

`DisplayRatio` - target matrix Width/Height ratio, used only for visualization purposes.

Listing B1 – Config file for standard General 12x20 Pressure Sensor Matrix

```
{                            //JSON
  "Configurations": [
    {
      "Name": "Matrix:12x20_US_MIX",
      "DriveLines": 12,          //column count
      "ScanLines": 20,           //row count
      "FrameBits": 12,
      "ADCValues": 1,
      "PCM": [
        [                        //first column in visualization-ready representation
          [0, 9],
          [0, 8],
          [0, 7],
          …
          [0, 19]
        ],
        [                        //second column
          [2, 9],
          [2, 8],
          [2, 7],
          …
          [2, 19]
        ],

       …                         //third to eleventh columns

        [                 //last twelfth column in visualization-ready representation
          [11, 9],
          [11, 8],
          [11, 7],
          …
          [11, 19]
        ]
       ],                 //PCM
       "DisplayRatio": 0.6
     }                    //Configuration
   ]
}                         //JSON
```