Lappeenranta University of Technology

Faculty of Technology

Energy Technology department

BH10A0201 Energiatekniikan kandidaatintyö ja seminaari

Mittausdatan havainnollistaminen MATLAB animaationa

Visualization of measurement data using MATLAB animation

Examiner/Supervisor: Arto Ylönen

Lappeenranta: 10.12.2014

Nicolas Juha Taba

TIIVISTELMÄ

Nicolas Juha Taba

Mittausdatan havainnollistaminen MATLAB animaationa

Teknillinen tiedekunta

Energiatekniikan koulutusohjelma

Kandidaatintyö 2014

30 sivua, 14 kuvaa ja 3 liitettä

Hakusanat: visualisointi, mittausdata, MATLAB, animaatio

Mittausdatan visualisointi on tärkeä osa-alue teknisen alan tutkimuksessa sekä tutkimustulosten esittelyssä. Tarvitsemme sopivia visualisointimenetelmiä tieteellisten mittaustulosten käsittelyssä ja havainnollistamisessa.

Tässä työssä esitetään tieteellisen visualisoinnin menetelmiä, jotka ovat perusta abstraktista visualisointiprosessista käytännön teknisiin sovellutuksiin, tilanteen mukaan.

Visualisointityökalun luomiseen käytetään MATLAB-ohjelmistoa. Työkalu on suunniteltu mahdollisimman monikäyttöiseksi erilaisten mittausdatan visualisoinnin tarpeisiin. Työkalu tarjoaa käyttäjälle mahdollisuuksia sekä staattisen että dynaamisen datan visualisointiin.

ABSTRACT

Nicolas Juha Taba

Visualization of measurement data using MATLAB animation

Faculty of Technology

Degree Programme in Energy Technology

Bachelor's thesis 2014

30 pages, 14 figures and 3 appendices

Key words: visualization, measurement data, MATLAB, animation

The visualization of measurement data is important in the fields of engineering for research analysis and presentation purposes. A suitable visualization method for scientific visualization is needed when handling measurement data.

Visualization methods and techniques will be presented throughout this work. They are the bases of scientific visualization from the abstract visualization process to the applied techniques suited for each situation.

This work also proposes a visualization tool using the MATLAB® software. The tool was designed as general as possible to encompass the most needs in terms of measurement data visualization. It offers possibilities for both static and dynamic visualization of the data.

TABLE OF CONTENT

Table of content

SUBSCRIPTS AND ABREVIATIONS

Subscripts

min=minimum

max=maximum

Abreviations

2D = two dimensional

3D= three dimensional

1 INTRODUCTION

Visualization is a process by which a person creates a perceptual model in order to identify patterns in a system to study. It most often yields an image that helps its user understand, analyze and communicate or share abstract data in the most efficient and clear way. In order to do so, the visualization process must satisfy three main criteria: expressiveness, effectiveness and appropriateness. Expressiveness refers to the requirement of showing the information contained in the data of interest as it is. Effectiveness is defined as the extent to which the visualization allows us to appropriately analyze and use the visual representation of the data. Appropriateness is related to the abstract concept of relevance of the visualization with respect to the task that is to be achieved. (Calin Enachescu & Osei Adjei, 2006)

The two main axes that the visualization process that must be studied in order to fulfill the three defined principles are the nature of the data and the purpose of the visualization.

Data can be modeled following a framework considering the location, time and theme of the data. Location defines the spatial location with regards to a reference and associates the data values with respect to each other. Time will describe the characteristics of the data values in relation to a time dimension. The theme of the data describes what is being measured (data values, objects and their relationship) as each of its elements represent an individual variable of a multivariate dataset. We can consider data in the broadest manner as independent or dependent variables. Independent variables define an n-dimensional set. In this set, variables dependent of another variable *a* define the *a*-variate dataset. If at least one variable is dependent of a time dimension, the data is time-oriented.

The purpose of visualization can be explorative analysis, confirmative analysis or the presentation of analysis results. Confirmative analysis is a type of oriented search that bases itself on a pre-existing hypothesis in the visualization process. The visualization

process then is built and optimized to describe said hypothesis that can then brought to the presentation step, once ascertained, in order to communicate the data analysis results. Explorative analysis involves indirect search with no prior hypothesis. It relies heavily on filtering and accentuation. Filtering is defined as the omission of less relevant data and accentuation consists in highlighting important information in the dataset. These two processes cannot be used as easily in explorative analysis as in confirmative analysis because the relationship of variables in the studied dataset is usually not known. This can lead to misinterpretation of the visualization. Filtering and accentuation help enhance expressiveness and effectiveness in the visualization. (M. J. Patizzo, R. F. Erbacher & L. B. Feldman, 2002). It is also important to take into account the finite resolution of the display when dealing with visualization. This finite display limits the injective nature of data representation as two points of a dataset may have the same location and value on a display. (Calin Enachescu & Osei Adjei, 2006)

Keeping these points in mind, this work will address the process of visualization in a conceptual way as well as the different techniques of visualization. It is impossible to define every visualization techniques as being part of a category as some categories would inevitably overlap and not be able to encompass all other elements within those categories. It is however more accurate to explain the advantages and disadvantages of each of the specific attributes that are part of the produced image in visualization. We will then present the tool built using the different methods, its advantages and shortcomings.

2 VISUALIZATION PROCESS

The visualization process consists in transforming raw data of any form into image data that accurately represent the relationship between subsets contained in the dataset that is studied. In other words, data must be mapped to visual attributes called visual variables. The visualization process pipeline chosen to study here is the extended variant from Dos Santos and Brodlie, 2004 originally from Haber and McNabb, 1990.

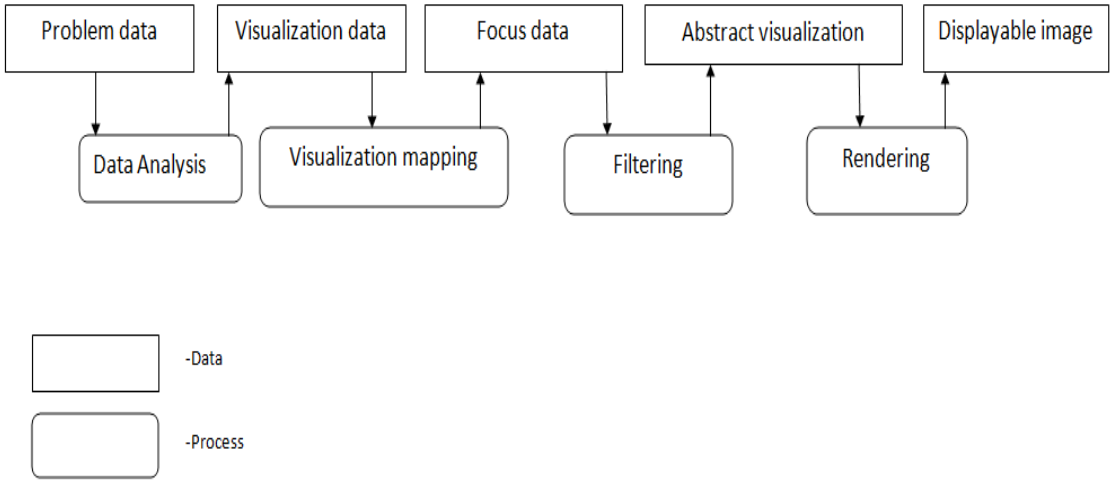


Figure 2.1: Dos Santos and Brodlie model for scientific visualization

This process takes into account higher dimensional data visualization as it divides the filtering operation into two distinct operations. Data analysis allows for computations such as interpolation or pattern recognition. It allows the filtering step to deal with less data more efficiently. This step mostly affects appropriateness in multidimensional data visualization. Filtering in higher dimensional data is very important as the visualization can rapidly become complex and overloaded with information which in turn would decrease expressiveness and effectiveness. Mapping consists in transforming elementary information into visual variables allowing to pass from data to geometric

data. Once the geometric data is stored, it can then be rendered into image data. The user can interact with this process at each step in various ways and however the user deems suitable. (Selan dos Santos & Ken Brodlie,2004)

The scope of this work is focused on measurement data visualization and therefore most of this paper will deal with time oriented data. In scientific measurements and engineering, measurements are done by observing how the changes of a given variable affect a studied system. These measurements and variables are inherently monitored in time.

A crucial part of the visualization process is the choice of the correct mapping technique for the data. This is done by choosing the technique most suited for the viewer to either see a feature of the data that is designed to be shown or find a feature within the data. We must also take into account aesthetics in this part, as it is often linked to the expressiveness of the visualization.

2.1 Visualization techniques

The first challenge of the visualization process with measurement data is to choose the most appropriate way to model time. Although time is intuitively continuous, data is stored in a discrete and ordered causal manner. The events that take place between the recordings of data need to be interpolated by different mathematical methods and scientific theories. These interpolations are approximations and we must keep this into account when presenting data on a continuous scale. A discrete scale has the advantage of allowing to accurately represent the way data is stored, however depending on the unit, it is sometimes impossible to establish the order of events. If the time unit used in the data representation is greater than the occurrence of two events, these two events will appear to be simultaneous (Figure 2.2).

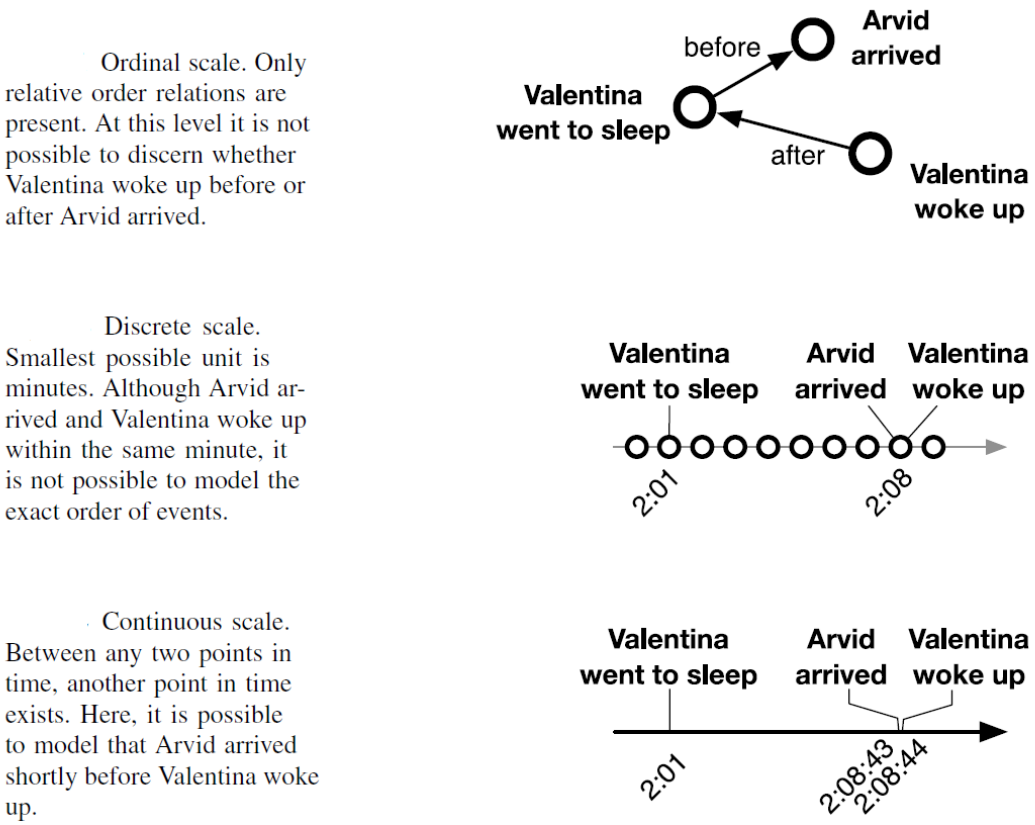


Figure 2.2: Modeling time using different scales (Wolfgang Aigner et al., 2011, pp 48)

The choice of time representation is also linked to the nature of the data as either linear with no relevant relation to a cycle or linked to a recurring time period. Time is also often considered as absolute and thus is often represented in a linear fashion rather than by changing it into a color or another visual object. Colors can be used to represent time when measuring the relationship of two or more variables with respect to the time the data was taken. In this case different transparencies and shades can be used. Certain color pallets are used instinctively to represent time as they fall both in the domain of aesthetics and comprehension without aid from additional data cues. As such as the data measurements get "older" we will use more transparent colors. This is limited to the extent at which the person viewing the data is able to distinguish different colors although the main goal of visualization is allowing to easily observe differences rather than absolute values. Visualization is also limited in complexity. Inputting too much

data into the produced image leads to a visualization that is difficult to analyze as the different points become undistinguishable from one another. Thus complexity leads to a loss of values of all the main characteristics of effectiveness, expressiveness and appropriateness (Graham Wills, 2012, pp 63-69).

Dynamic representation of time allows linking the characteristics of different sets within the data. It is most often used when linking location and data value relationship in time. The successive frames generated are in fact considered as the visualization in time as they are the way time is encoded into the process of visualization. Each frame or image represents the data at a specific time instant. Depending on the level of precision of time steps that are to be presented, this method is sometimes limited by the approximation using interpolation of intermediate data values between time instants. This method is also limited by the perception that is to be conveyed to the viewer of the impression of dynamism in the representation. The dynamic representation of a small set of frames is better represented as a slide show (usually 2 to 4 frames per second). On the other hand, when measuring data during a large amount of time, the dynamic process is best represented as an animation (15 to 25 frames per second). In the case of dynamic representation of the data, it is often difficult to visualize multiple measurement variables at the same time as different measurement variables may use different scales. This causes the user to being unable to follow the changes in variables when visual cues and information cannot all be processed in a satisfactory way. In static representation of time on the other hand, time change needs to be represented on screen and as represents a non-negligible amount of space on the image. This method however has the advantage of fully focusing the user on the relationship of the data with time and data values with each other rather than on the general dynamic change of the system. The visualization of large datasets is however difficult in this case as the representation can be overcrowded with visual cues. It is difficult to evaluate the values and relationship between data subsets due to a high number of visual cues on screen (Figure 2.3). However, this represents the first step in data analysis and trends or extremums can be further analyzed from this parallel graph. (Aigner et al., 2011, pp 76-83)

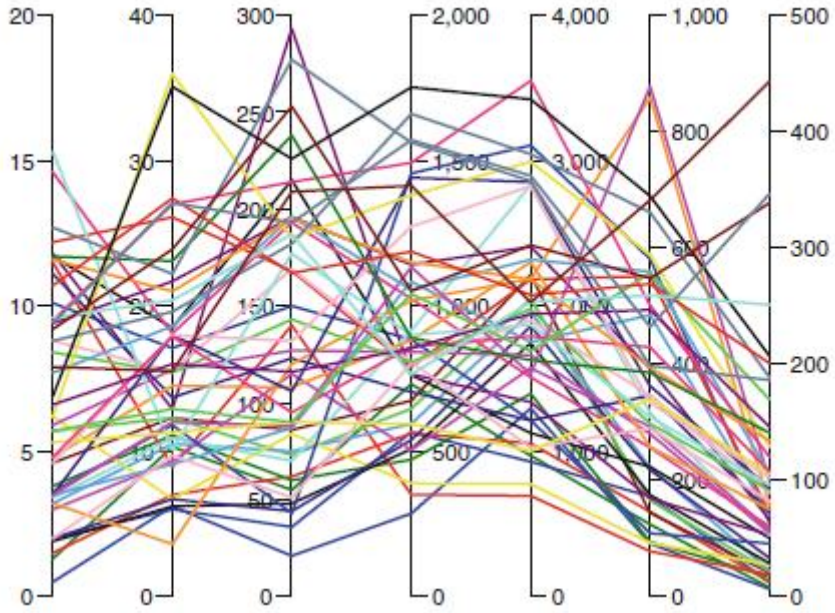


Figure 2.3: Rate of occurrence of crimes in the US using parallel axis visualization (Graham Wills, 2012, pp51)

Measurement data is always quantitative and spatially defined. This allows comparison between events, location in time and space. The difficulty most often lies in representing multivariate independent datasets where multiple data values are associated with one time event or instant. In this particular case, additional dimensions have to be added to the generated image in order to account for the additional information from the data. Measurement data is also inherently discrete and the choice is left to represent the data as discrete, interpolate data values between points using the method with most significance or assign the data value to the interval; thus creating an interval based representation of the data. The choice of method depends of the data analysis that one can make of the data before mapping it to visual variables. This encompasses the choice of using points, lines or areas when representing the data.

Two-dimensional and three-dimensional (2D and 3D) representations of data are as well two ways of visualizing data. 2D visualization uses only a plane in order to represent the data. Time is then usually represented on one of the axis (most often the horizontal

axis). The geometry of the axis may vary depending on the data that has to be visualized. As such affine transformations or the use of spiral or circular coordinates in the case of a cycle or simply can be applied to the axis. 2D visualization of data can also be enhanced using multiple ways of representing data at the same time in order to convey more information and make more accurate analysis. Time is represented as both cyclic and linear in order to enhance the comparison between each data sets and their relationship (Figure 2.4).

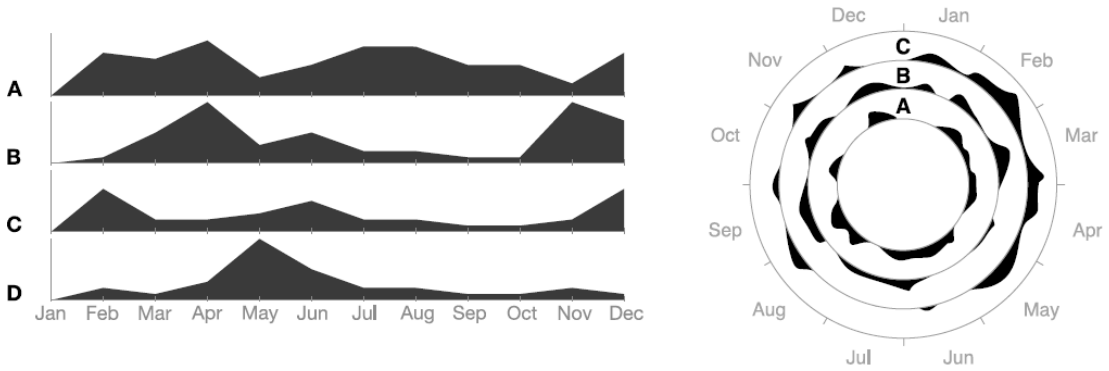


Figure 2.4: Yearly occurrence of a phenomena using linear and cyclical representation of time. (Wolfgang Aigner et al., 2011)

3D visualization enables to give the user the illusion of depth on a plane display. This is achieved by rendering skewed plane images that accurately depict the appearance of perspective. This method is the one commonly used when mapping more than one independent variable as it would create an overlapping of data on a 2D representation. Although a 2D plot may accurately display multivariate problems through parallel coordinates all these dimensions are dependent to another in order to have effectiveness. (Alfred Inselberg, 2002). In the case of independent variables, large data sets would overlap and hinder further analysis of the produced image. A 3D plot is able to represent data in several dimensions: three spatial dimensions and visual cues (shape and color) as in Figure 2.5.

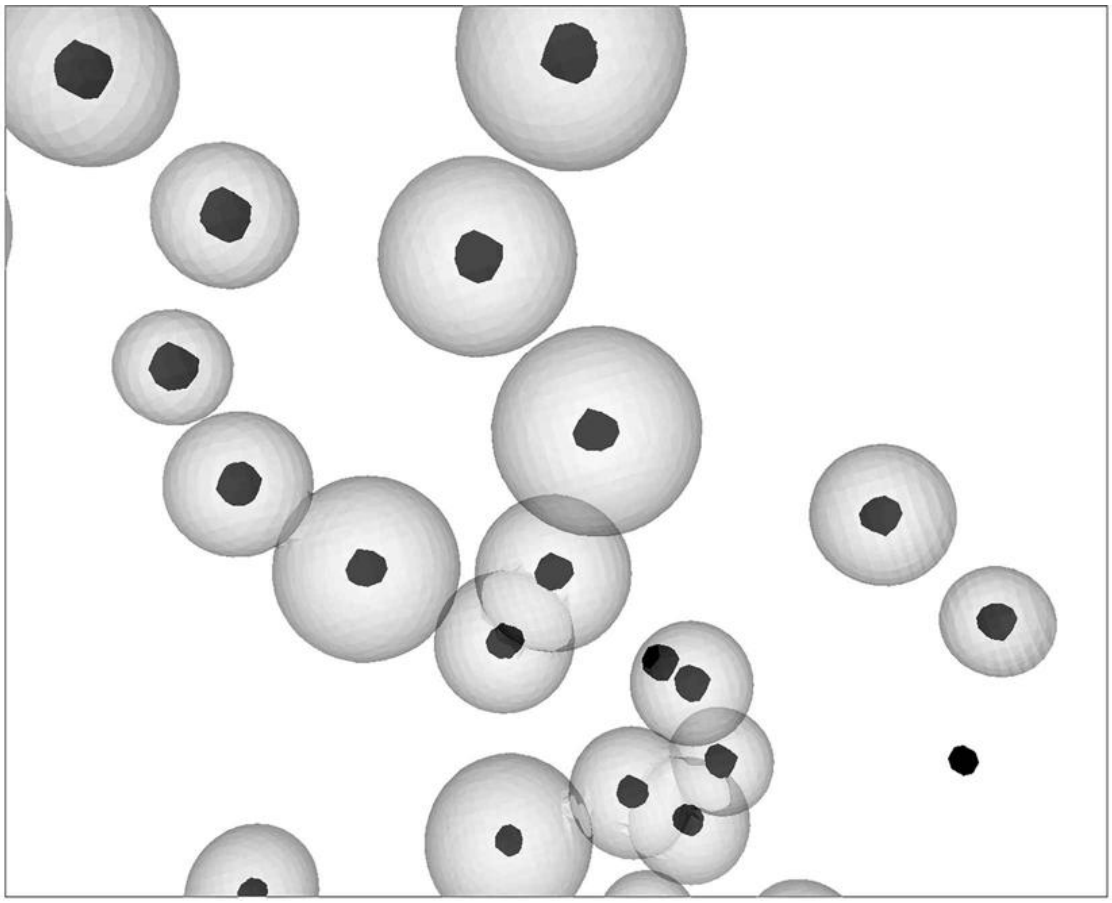


Figure 2.5: Example of 3D visualization. Three variables are mapped on the three primary axes. Two more variables and their relation are mapped by the sphere sizes. (M. J. Patizzo, R. F. Erbacher & L. B. Feldman, 2002)

The need for 3D visualization is however debatable as the information and the possible conclusion done from data analysis can become more difficult due to an added dimension and perspective. However large datasets of independent variables may need a third dimension in order to correctly visualize data without over cumbering the image. A 3D image allows encoding more information. However we need to address the difficulty encountered, as mentioned earlier, when generating such an image. It is important to allow a certain level of interactivity between the user and the image as it allows to remove the aspect of perspective and data that could be omitted. A method called data spinning (Franck M Marchak, 2004) allows the viewer to rotate the produced 3D image through software either passively (animation) or actively (user controlled

rotation). We can also think of the 3D space as a geospatial representation of a data set and use dynamic visualization in order to represent data accurately as the geospatial information is directly encoded in the produced image. The choice of 2D or 3D data is fully dependent on the type of data that is to be presented with no significant advantage that eliminates the other method. (Aigner et al., 2011, pp 96-99).

Color coding is used to map data to color and remapping one information of the data to a different visual cue for the user. Color coding can be understood as the process by which a mapping function transforms a dataset into a color scale (Figure 2.6). This function must be injective, meaning that every associated data value to a color is unique. The goal of visualization being the presentation of data in a simplified manner, similar colors are interpreted as data of similar value and oppositely clearly distinguishable colors as different values without knowledge of the absolute values involved. This mapping of data is used to focus the user on the relative relationship of data values rather than their absolute value.

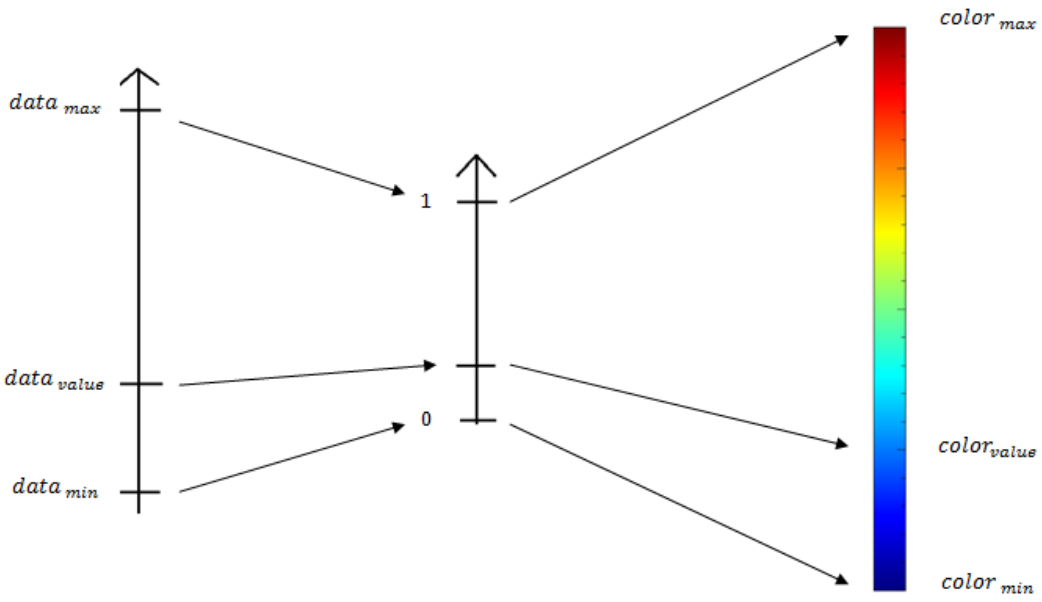


Figure 2.6: Color coding process

In order to transform the data value into a color value, we must first scale the data value to a unit scale and this scaled value can then be associated proportionally to the color value using the following formulas:

$$scaled\ value = sv = \frac{data_{\text{value}} - data_{\text{min}}}{data_{\text{max}} - data_{\text{min}}}$$

$$color\ value = (1 - sv)color_{\text{min}} + sv \cdot color_{\text{max}}$$

When mapping data to colors, it is important to take into account the extreme values as well as the distribution of the data. Rapid variation in datasets or using a non-linear mapping to a linear representation can lead to erroneous analysis of the image. The mapping of the color can also be done in a segmented or continuous way. The segmented mapping of color allows to associate a color to a full range of values in order to further simplify association whereas continuous mapping displays the full range of values of color to data values allowing to analyze important variations. The choice of using a particular type of color scale depends on the aim of the visualization. If only general trend is aimed for rather than accuracy or dynamic visualization rather than that of a static system of the dataset, it is more relevant to choose a segmented color mapping.

If the produced image does not fully allow the user to appreciate the variations of data, the color scale needs to be changed or skewed. The color scale can then be skewed by considering only the data within a certain range of values or by expanding the range of data studied or by using different transformation methods such as logarithmic or exponential scaling when dealing with heavily skewed data sets. Color coding is very

dependent of the user task and as such must be taken into account in the process of visualization when thinking of effectiveness. (Aigner et al., 2011, pp 88-95)

2.2 Data enhancement

Another important aspect in the data visualization process is the way data is handled before rendering it. This specifically is of concern during the data analysis and filtering operations in the visualization process. Data analysis is often associated with data mining. Data mining is defined as the use of search algorithms applied to large data sets in order to gain knowledge on different useful structures in the data. Knowledge of the data set is assumed to be the end goal of data mining. The goals of data analysis and filtering are to reduce the size of the data subsets that are going to be represented.

Several methods are used when carrying out data analysis. Techniques used for time oriented data aim to find trends and unusual values within the data. The first step in finding trends is classification and clustering. They enable us to relate the data into coherent and similar subsets. The difference between classification and clustering is that the classification process assumes that the subsets are known prior to the measurement whereas clustering subsets are not known upfront. These subsets help in the minute analysis of even smaller subsets in order to find patterns in our data. Clustering uses a visualization process of its own. Through visualization of different subsets of data, clusters of these sets can be formed by measuring which ones have the most similarities (least variations between differences of measured data sets). The number of data points in each subset is smaller than the overall dataset and thus allows for an easier analysis of the dataset. Visualization of these clusters allows the user to determine patterns in the data that can then be further explored in the overall visualization process. (Aigner et al., 2011, pp 127-131)

2.3 Most used techniques for visualization

Aigner & al. reviewed 101 existing techniques for visualizing time and time oriented data and observed that certain visualization methods and techniques are used more often than others. Time is most often represented in a linear fashion as the user seem to be more interested in trends evolving linearly rather than cyclically for predictions most often. Time is also mapped as instants in time more often than as intervals as it is easier to record measurements at a given time (Figure 2.7). Intervals are used when clustering data in very large data sets.

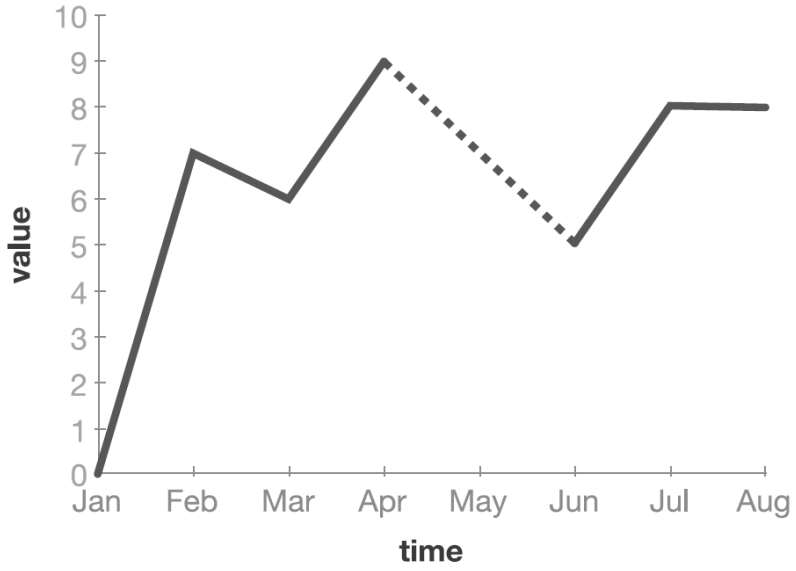


Figure 2.7: Simple 2D line plot with an interpolation of data. (Aigner & al., 2011, pp 153)

Two-dimensional representations are favored over 3D as they can easily be shared without the use of software. However the use of both static 2D and dynamic 3D visualization allows for more possibilities and amount of data that can be handled. This difference in the use of 2D and 3D visualization seem to be diminishing with modern technologies that allow user and visualization designer to operate 3D environments (Figure 2.8). Throughout the review, it is also worth noting that the line plots, point

plots and color coding of data are the most used techniques for the simplest visualization tasks.

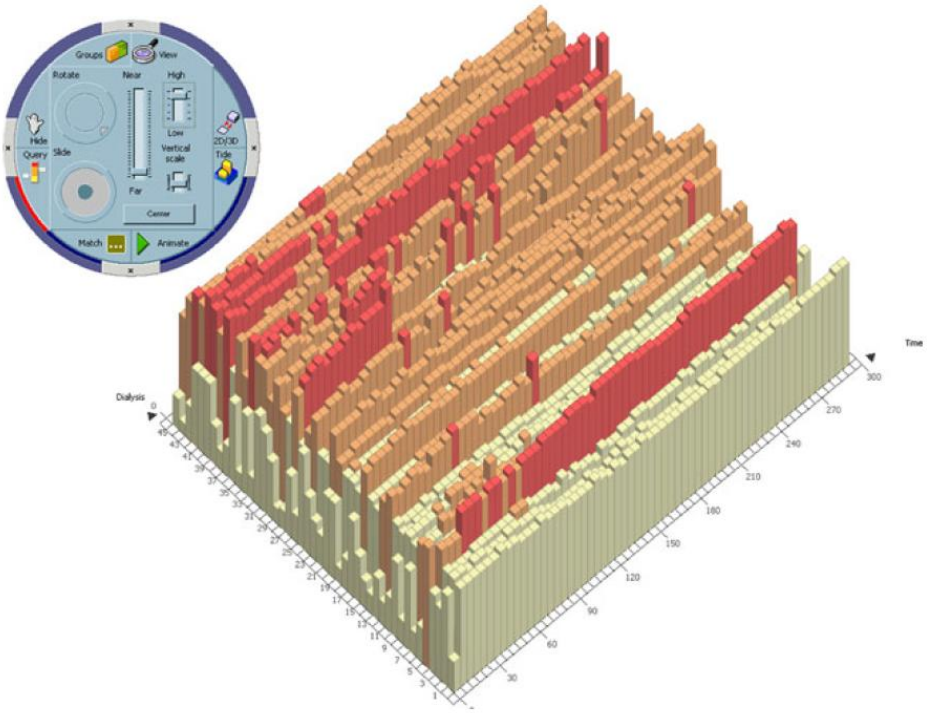


Figure 2.8: Clinical time dependent data visualization in an interactive 3D visualization environment. (Aigner & al., 2011, pp 162)

Other concepts seem to be more balanced in their use. Univariate and multivariate visualization techniques are equal in numbers although it is worth noting that a great number of multivariate visualization techniques are in fact repetition of univariate visualization. (Figure 2.9)

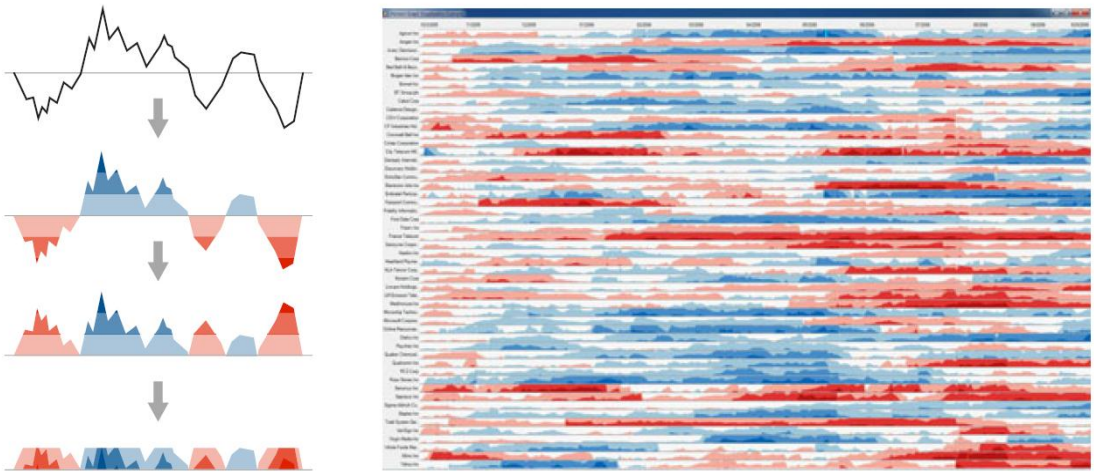


Figure 2.9: Construction of a horizon graph and its use in iterated visualization of stock market data. (Aigner & al., 2011, pp 160)

There are also a balanced number of visualization techniques that use a spatial frame of reference with abstract data as there are techniques that deal with purely abstract data. Techniques that use a spatial frame of reference seem to be mostly used when the location of the measurement is central in the analysis of the visualization. In Figure 2.10, the location of measurement data is important for the implementation of treatment plans and monitoring the patient's health (Aigner et al., 2011, pp 253-254).

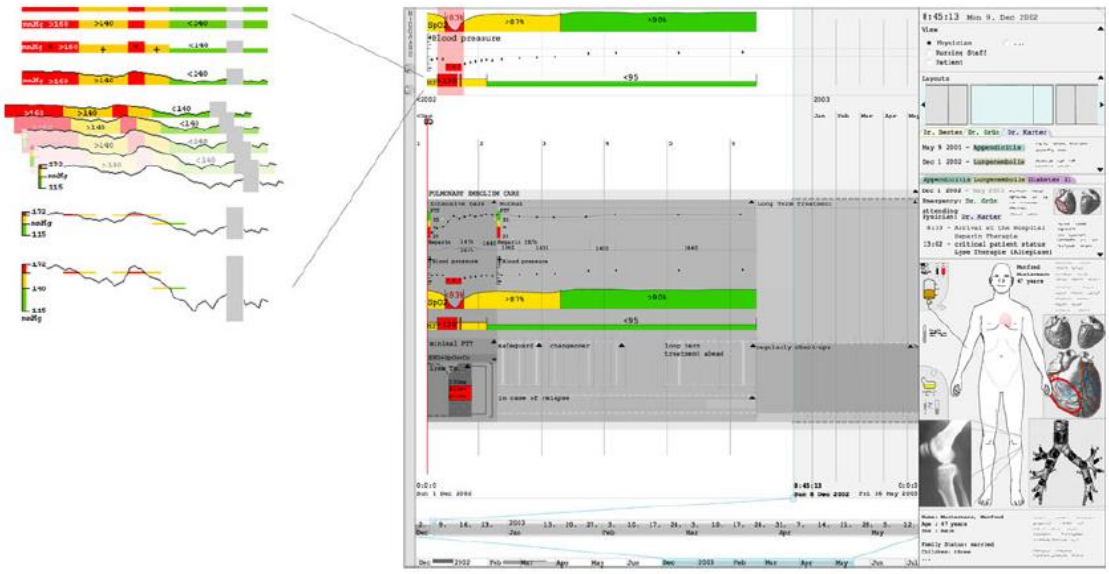


Figure 2.10: Midgaard visualization tool presents time oriented data of patients. On the right : spatial data and treatment data of the patient. On the left: time series related to a specific variable of the patient's data monitored in time. (Aigner & al., 2011, pp 230)

3 MATLAB VISUALIZATION TOOL

Following the previous chapter, a tool was designed using the software MATLAB®. MATLAB® is a high level language and interactive environment for numerical computation, visualization and programming. This tool is designed to be used by a user having access to MATLAB®, a compatible Microsoft Excel software as well as a software enabling to transform multiple pictures from .PNG format into a video.

The tool has two distinct parts: a time line and a test facility visualization. Both of these parts depict the variation of data in different ways and thus have different goals. The user has the choice of using either or both visualization processes with the dataset from a template Microsoft Excel file (Appendix 2). The tool enables the visualization of several measurement points in a facility at the same time. The tool was built in order to accommodate for most of measurement visualization as possible. It should be noted from the start that it was built as a time oriented data visualization tool and that higher dimensionality can be reached for further analysis by reiterating the visualization using a different variable dataset. (Yoh-Han Pao & Zhuo Meng, 1998)

3.1 Line plot

The goal of the timeline line plot is to allow analysis between different measurement points in a facility. As such, it depicts several line plots distinct from each other using different colors and line styles. The measurement values are represented versus time. The number of measurement points when using the timeline is limited to 12 as more lines would over cumber the display and lead to making data analysis more difficult. Each line plot is referenced in a legend to facilitate recognition of data. Line plots are named after the names of variables in the data subsets from the Excel file template (Figure 3.1)

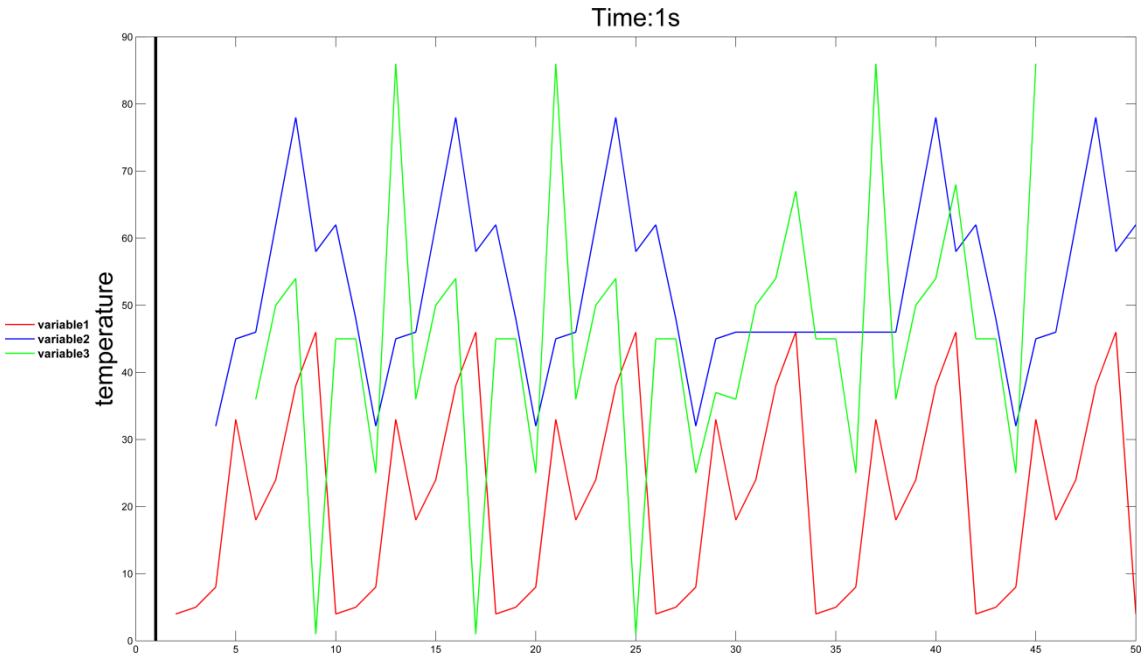


Figure 3.1: Line plot visualization using arbitrary data

Time is treated here on two separate levels. A subset of the vector depicting the total time span over which all measurements were made will govern the time span on which the visualization is wished. The data values for each of the measurement points are then fetched by the closest value to the time of the total time vector. In case of different values for the vectors, the next measurement value is interpolated linearly. This is favorable for big datasets as it is smoothed by lower point wise resolution, but is erroneous on very small datasets as the time unit becomes greater or smaller than the occurrence of two successive measurement.

The timeline line plot also features a vertical line that indicates the current time across the line plots as well as a display of the current time for each frame in the title. The saving of each frame image allows for static frame to frame analysis although the visualization process can also be presented dynamically.

3.2 Test facility visualization

The goal of the test facility visualization is to allow analysis of events in the measurement facility in time. This part of the tool presents the measurement point values using colors referenced in a color bar and the measurement point values in time. Each point is identified by a number that is the rank of the data subset in the excel template.

The figure in which the facility is represented can be tailored by the user. The user can define the unit on both x and y axes and place a background image of the facility in the figure. The measurement points are located according to the data in the excel file as a portion of the total length of the figure representing the facility. It should be noted that the location of the points are mapped according to the absolute value of the length of the background image size in either direction. The coordinates are thus only indicated as a number between 0 and 1. The time and data values are treated in the same manner than in the timeline line plot (Figure 3.2).

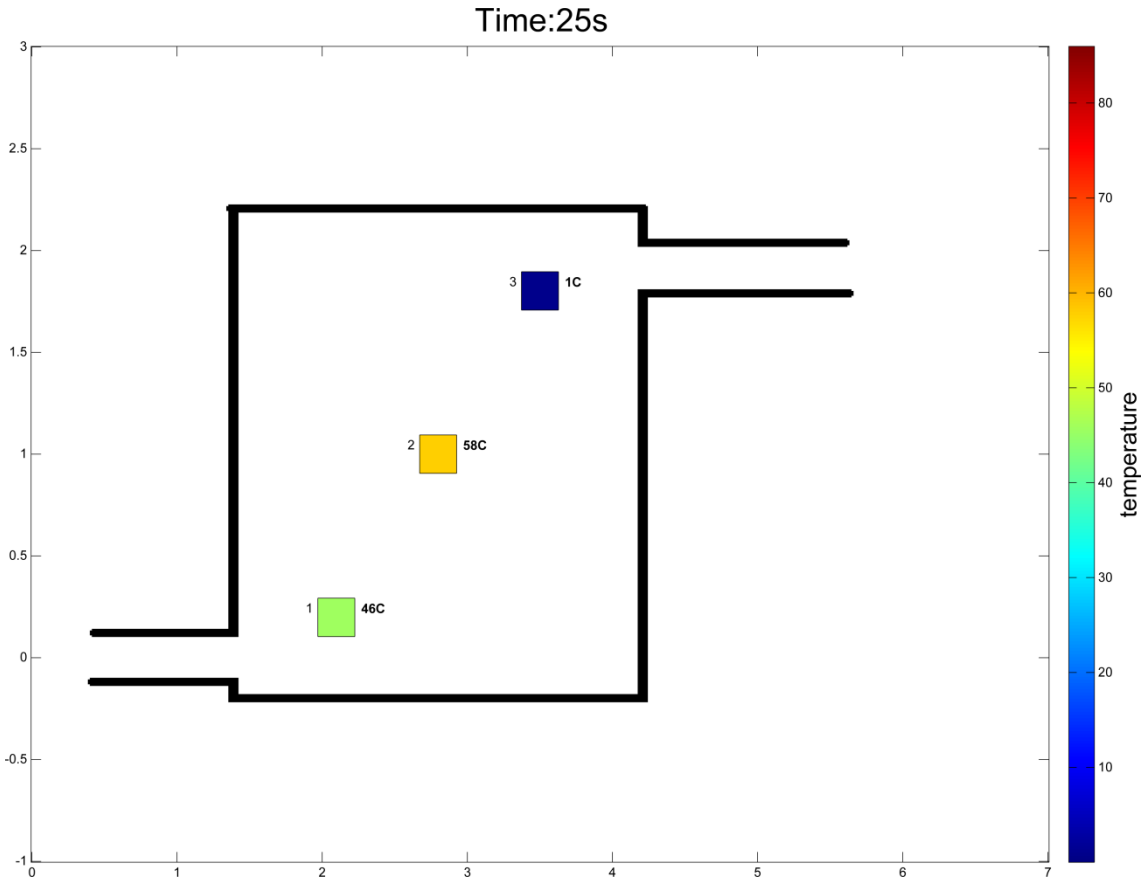


Figure 3.2: Facility visualization using arbitrary data

To relate each value to a color, we used the formulas presented in the second chapter of this work. We used a continuous scale of 255 color values for the color bar and the data is associated to the closest value to this color. In terms of resolution of colors, this process is admittedly inaccurate but the goal of this visualization is to simplify the understanding of the behavior of the studied dataset in time and not to make a quantitative analysis of the data. (Graham Wills, 2012, pp 32). In these terms, the approximation of color is acceptable as subtle changes in color indicate little variation between data values and great variation of color indicate great differences in data values. The color bar can also be scaled to suit the user. The user will be asked if they wish to change the scale of the color bar and may change the extremum values displayed in the visualization. However, the maximum value of the color bar must be

greater than the maximum value of the data subsets and the minimum value must be smaller than the minimum value of the data sets. In order to avoid conflict within the code of the tool, missing values that cannot be interpolated were color coded in white and the value of the data is given as NaN (not a number).

3.3 Advantages and shortcomings of the tool

The timeline plot visualization technique was chosen for its high user recognizable value. The line plot is one of the most used data visualization method and allows for diverse use.

This technique has the advantage of being both efficient for static and dynamic visualization when analyzing and understanding the data. It allows for accurate comparison of data values and prediction. However it is limited by the number of data subsets that can be represented and the fact that the relationship between each measurement point is not always straightforward. Due to the higher resolution of the visualization, it is also to be noted that small data sets will render a result with low effectiveness. This part of the tool is primarily used to compare data values in time and to start the data enhancement process. Iterating the visualization process and changing time intervals allows for suitable filtering for the user.

The facility visualization (spatial visualization) and color mapping of data were chosen for their efficiency when doing dynamic visualization. Their use is limited in static visualization as it is the variation of color in time that allows for an understanding of the data. This is an ideal process for presentation of data. The color mapping was chosen as it enables to convey intensity of data values with respect to a scale. The user here has the opportunity to modify this scale to modify the data analysis outcome. When comparing Figure 3.2 and Figure 3.3, the user can understand the data differently due to the change of color reference scale. The difference between the values at every point seems much smaller in Figure 3.3 than in Figure 3.2 although the data is the same.

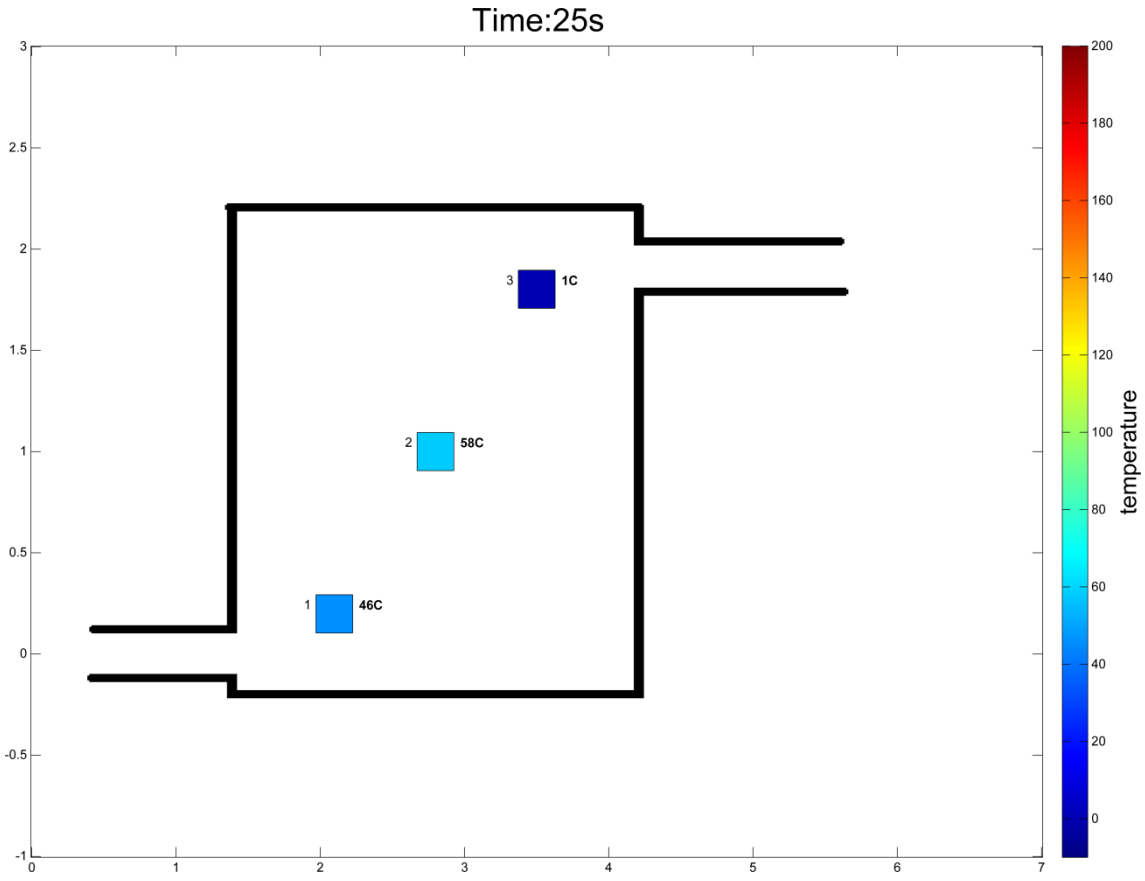


Figure 3.3: Facility visualization using arbitrary data and modified color bar scale.

The shortcomings of this visualization also come from the fact that it shows only a 2D image of the facility thus not accounting for the volume of the measurement facility. However as expressed at the beginning of this chapter, by iterating the process and using an orthogonal view of the facility it is possible to have all 3 dimensions of space and make accurate presentation and analysis of a phenomena in the facility.

When both processes are used at the same time, we lose a lot of resolution for the line plot and a little less for the facility visualization. However in this case, the line plot is a visual aid used by the user as a reference for time with respect to the total time interval as well as a visual cue as to previous or future changes in data. The previously presented shortcomings of the facility visualization are diminished as the approximate variation of data is represented by the line plot (Figure 3.4).

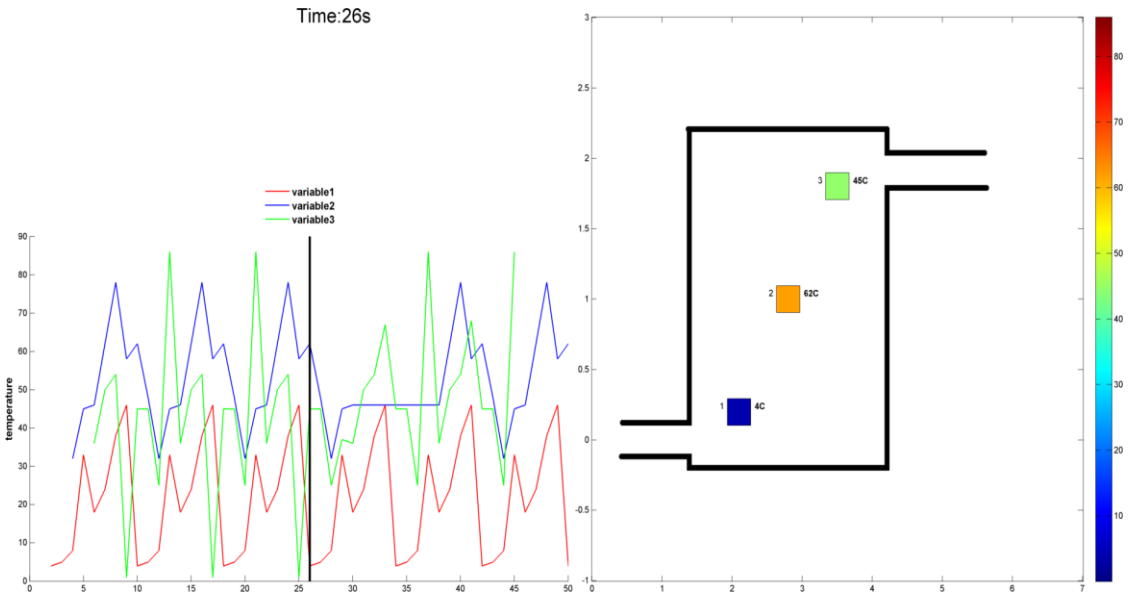


Figure 3.4: Visualization with both line plot and facility using arbitrary data

On the scale of the whole tool, it is undeniable that the use of a particular template for the excel file, the use of a third party function and the fact that this tool works best with large data sets, constitute shortcomings for this tool.

4 SUMMARY

The goal of this work was to present a measurement data visualization tool using MATLAB®. After presenting the scientific process of visualization and the different aspects and methods used in visualization, the tool was designed to offer the most possibilities to the user and uses the most widely spread techniques for visualization. It is however considered as a visualization tool of general purpose and the user will have to keep in mind the task specific aspects and parameters for each visualization. The limitations of the tool are its low efficiency with small datasets and the need for many iteration of the visualization process in order to allow a complete analysis of the data.

The tool could be improved or redesigned for monitoring purposes and allowing an interactive three dimensional model of the facility. The tool code can also be modified following the comments in order to implement suitable interpolation for the user's task.

REFERENCES

Wolfgang Aigner et al., 2011, Visualization of time-oriented Data, Springer-Verlag London Limited, ISBN 978-0-85729-078-6

Alfred Inselberg, 2002, Visualization and data mining of high-dimensional data, Chemometrics and Intelligent Laboratory Systems 60, pp 147– 159

Franck M Marchak, 1994 An overview of scientific visualization techniques applied to experimental psychology, Behavior Research Methods, Instruments & Computers 26 (2), pp 177-180

M. J. Patizzo, R. F. Erbacher & L. B. Feldman, 2002, Multidimensional data visualization, Behavior Research Methods, Instruments, & Computers 34 (2), pp 158-162

Dumitru Radoiu, Calin Enachescu & Osei Adjei, 2006, A systematic approach to scientific visualization, Engineering Computations: International Journal for Computer-Aided Engineering and Software Vol. 23 No. 8, pp. 898-906

Selan dos Santos & Ken Brodlie, 2004, Gaining understanding of multivariate and multidimensional data through visualization, Computers & Graphics 28, pp 311-325

Graham Wills, 2012 Visualizing Time: Designing Graphical Representations for Statistical Data, Springer New York Dordrecht Heidelberg London, ISBN 978-0-387-77906-5

Yoh-Han Pao & Zhuo Meng, 1998, Visualization and the understanding of multidimensional data, Engineering Applications of artificial Intelligence 11, pp 659-667

APPENDIX 1. TOOL MATLAB CODE

```
%This tool uses a third party function. The third party function can
be found on the file exchange of mathworks.com under the name of
export_fig created by Oliver Woodford (2009).
%This tool uses a set excel file template that must be respected for
the code to compute.

%prompt for visualization method.
%for all prompts, use lowercase letters.
prompt_visualization_style={'timeline? y/n ','facility display? y/n'};
dlg_title_visualization='visualization ';
visualization_style=inputdlg(prompt_visualization_style,
dlg_title_visualization);
%string comparison for visualization method choice
str_comparison_vis_timeline=strcmp(visualization_style{1},'y');
str_comparison_vis_facility=strcmp(visualization_style{2},'y');

if str_comparison_vis_timeline==1 && str_comparison_vis_facility==1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%visualization with both timeline and facility.

%initializaing and assuring that there is no variable conflict.
clear all
clc

%Location for visualization axes variables.
%Variables are hardcoded to avoid overlapping  and location of color
boxes.
left=0.52;
bottom=0.12;
width=0.42;
height=.84;
%minimum value for background image.
min_x=0;
min_y=0;

%colors and linestyles for legend.
colors=['r','b','g','c','r','b','g','c','r','b','g','c'];
linestyles = cellstr(char('-','-','-','-','-.','-.','-.','-.','-*','-
*','-*','-*'));

%Excel data file prompt.
%The Excel file must follow the template. There is no need to add the
%extension.
file_name=inputdlg('what is the excel file name? ','data file');
[data_points,strings]=xlsread(file_name{1});
[length_colums,length_row]=size(data_points);

%time vector.
time_v=data_points(4:end,4);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%prompt time interval.
%time start must be within the overall measurement time interval.
%time end must be within the overall measurement time interval.
%time step must be smaller than the overall measurement interval span.
%the time interval must have at least 2 values that are not NaN.
prompt_time_interval={'time start ','time end','time step'};
dlg_title_time_interval='time interval ';
size_time_interval=inputdlg(prompt_time_interval,
dlg_title_time_interval);
size_time_interval=str2double(size_time_interval);
time_interval=size_time_interval(1):size_time_interval(3):size_time_in
terval(2);

% Create figure.
%Figure is set to full screen size, changing this value would affect
%position of elements within the figure.
figure1 = figure;
set(figure1,'Units','Normalized','OuterPosition',[0 0 1 1]);
set(figure1,'Color',[1 1 1]);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create axes 1 for visualization.
axes1 = axes('Parent',figure1,...
    'Position',[left bottom width height]);
box(axes1,'off');

%prompt size along axes.
%values can be negative but the minimum must be smaller than the
maximum.
%these values must be entered by the user if the user wishes to change
the background image.
prompt_size={'minimum x axis ','maximum x axis','mimimum y
axis','maximum y axis'};
dlg_title_size='size ';
size_background=inputdlg(prompt_size, dlg_title_size);
size_background=str2double(size_background);

%prompt for background image change.
background_change=inputdlg('Do you wish to change the background
image? y/n','background');
str_comparison_background=strcmp(background_change,'y');

if str_comparison_background==1
%image background prompt.
%image is flipped and the direction of y-axis is set to normal in
order to avoid reversed background image.
background=inputdlg('Background:name of the file with extension?
','background');
img=imread(background{1});
```

```
imagesc([size_background(1) size_background(2)], [size_background(3)
size_background(4)], flipdim(img,1));
set(gca,'ydir','normal')
hold on
else
end

%save axis position and ratio when adding the colorbar.
p=get(gca,'position');
colorbar;
set(gca,'position',p);

%colorbar range change prompt.
color_bar_change=inputdlg('Do you wish to change the extremums of the
colorbar? y/n','colorbar');
str_comparison_colorbar=strcmp(color_bar_change,'y');

% Create colorbar and colorbar range.
format long
colormap('jet(255)');
colorbar('peer',axes1);

%if loop for colorbar range .
%color bar maximum value must be greater or equal to data maximum
value.
%color bar minimum value must be smaller or equal to data minimum
value.
if str_comparison_colorbar==1
prompt_colbar={'minimum value','maximum value'};
title_colbar='colorbar extremums';
colbar_size=inputdlg(prompt_colbar,title_colbar);
colbar_size=str2double(colbar_size);
caxis([colbar_size(1) colbar_size(2)]);
else
colbar_size=[min(min(data_points)) max(max(data_points))];
caxis([colbar_size(1) colbar_size(2)]);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create axis timeline.
%location is hardcoded to avoid overlapping.
axes2 = axes('Parent',figure1,...
    'Position',[0.04 bottom 0.46 0.6]);
box(axes2,'off');
hax=axes2;

for i=1:data_points(3,1)
     %save time data in array.
     B{i}=data_points(4:end,(5+((i-1)*2)));
     %time interval data.
     [t_start_val pos]=min(abs(size_time_interval(1)-B{i}));
     [t_end_val pos_end]=min(abs(size_time_interval(2)-B{i}));
     %new time array.
```

```
     C{i}=B{i}(pos):size_time_interval(3):B{i}(pos_end);
    %save data in array and interpolate intermediary NaN values.

A{i}=data_points(B{i}(pos)+3:size_time_interval(3):B{i}(pos_end)+3,(6+
((i-1)*2)));
     nan_int=isnan(A{i});

A{i}(nan_int)=interp1(C{i}(~nan_int),A{i}(~nan_int),time_interval(nan_
int));
     %save legend.
     L{i}=char(strings(3,(6+((i-1)*2))));

hold on
%location of squares and measurement point references.
%changing size of boxes may cause overlapping.
box{i}=annotation(figure1,'rectangle',...
    [left+data_points(1,(6+((i-1)*2)))*width-0.01
bottom+data_points(2,(6+((i-1)*2)))*height-0.02 0.02 0.04]);

measurement_nbr{i}=annotation(figure1,'textbox',...
    [left+data_points(1,(6+((i-1)*2)))*width-0.02
bottom+data_points(2,(6+((i-1)*2)))*height-0.02 0.02 0.04],...
    'String',num2str(i),...
    'FitBoxToText','off',...
    'EdgeColor','none');

%ploting data on timeline axis.
plot(C{i},A{i},[linestyles{i} colors(i)],'LineWidth',1.2);
end

%adding legend to the timeline.
legend(L,'Location','northoutside','Fontsize',12,'FontWeight','bold');
legend('boxoff');
ylabel(strings(5,1),'Fontsize',12,'FontWeight','bold');
%size of time array.
s_time=size(B{1}(pos):size_time_interval(3):B{1}(pos_end));
s_time=s_time(2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

iteration=1;
%image saving for-loop.
%saves the image after rendering each column.
for j=1:s_time;
%for loop saving the line plot visualization.
for i=1:data_points(3,1);
%save rgb value corresponding to each data value.
d_perc=(A{i}(j)-colbar_size(1))/(colbar_size(2)-colbar_size(1));
format long
cmap=colormap('jet(255)');
x_rgb=d_perc*255;
x_rgb=round(x_rgb);
%if loop assuring the initialization.
```

```
if x_rgb==0;
    x_rgb=1;
else
end
%if loop changing the rgb value to white if the data is NaN.
if isnan(x_rgb)==1
    y_rgb{i}=[1 1 1];
else
y_rgb{i}=cmap(x_rgb,:);
end
end

%for loop saving the facility visualization.
for i=1:data_points(3,1);

%change the facecolor of each square in axis 1 and the value of the
data.
box{i}=annotation(figure1,'rectangle',...
    [left+data_points(1,(6+((i-1)*2)))*width-0.01
bottom+data_points(2,(6+((i-1)*2)))*height-0.02 0.02 0.04],...
    'FaceColor',y_rgb{i});
var_value{i}=annotation(figure1,'textbox',...
    [left+data_points(1,(6+((i-1)*2)))*width+0.01
bottom+data_points(2,(6+((i-1)*2)))*height-0.02 0.04 0.04],...

'String',strcat(num2str(A{i}(j)),strings(5,3)),'FontWeight','bold',...
    'FitBoxToText','on',...
    'EdgeColor','none',...
    'Tag','values');

end
%location of timeline marker (vertical black line).
mov_point= time_interval(j);
l=line([mov_point mov_point],get(hax,'YLim'),'Color',[0 0
0],'Linewidth',3);

%saving the timer string for title.
T = time_interval(j);
timer=['Time:',num2str(T),strings(6,3)];
timer=strcat(timer(1),timer(2),timer(3));

%Textbox for title.
%A textbox was chosen over the common "title" command for overlapping
%reasons.
annotation(figure1,'textbox',...
    [0.2 0.924 0.2 0.058],...
    'String',timer,...
    'HorizontalAlignment','center',...
    'FontSize',22,...
    'FontName','Arial',...
    'FitBoxToText','off',...
    'LineStyle','none',...
    'Tag','Title_figure');
```

```
%saving the figure.
export_fig(gcf,sprintf('movie_%d.png',iteration),'-r100');
%deleting all data that are "hold on" command sensitive using Tags.
iteration=iteration+1;
delete(l);
delete(findall(gcf,'Tag','values'));
delete(findall(gcf,'Tag','Title_figure'));
hold off
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%timeline plot option only.

elseif str_comparison_vis_timeline==1
%timeline only.
clear all
clc

%colors and linestyles for legend.
colors=['r','b','g','c','r','b','g','c','r','b','g','c'];
linestyles = cellstr(char('-','-','-','-','-.','-.','-.','-.','-*','-
*','-*','-*'));

%Excel data file prompt.
file_name=inputdlg('what is the excel file name? ','data file');
[data_points,strings]=xlsread(file_name{1});
[length_colums,length_row]=size(data_points);
j=1:(length_colums-3);
i=1:data_points(3,1);

%time vector.
time_v=data_points(4:end,4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%time interval.

%prompt change interval.
prompt_time_interval={'time start ','time end','time step'};
dlg_title_time_interval='time interval ';
size_time_interval=inputdlg(prompt_time_interval,
dlg_title_time_interval);
size_time_interval=str2double(size_time_interval);
time_interval=size_time_interval(1):size_time_interval(3):size_time_in
terval(2);

% Create figure.
figure1 = figure;
set(figure1,'Units','Normalized','OuterPosition',[0 0 1 1]);
set(figure1,'Color',[1 1 1]);
```

```
% Create axis timeline.
axes2 = axes('Parent',figure1);
box(axes2,'off');
hax=axes2;

for i=1:data_points(3,1)
 %save time data in array.
     B{i}=data_points(4:end,(5+((i-1)*2)));
     %time interval data.
     [t_start pos]=min(abs(size_time_interval(1)-B{i}));
     [t_end pos_end]=min(abs(size_time_interval(2)-B{i}));
     C{i}=B{i}(pos):size_time_interval(3):B{i}(pos_end);
    %save data in array.

A{i}=data_points(B{i}(pos)+3:size_time_interval(3):B{i}(pos_end)+3,(6+
((i-1)*2)));
     nan_int=isnan(A{i});

A{i}(nan_int)=interp1(C{i}(~nan_int),A{i}(~nan_int),time_interval(nan_
int));
%save legend.
     L{i}=char(strings(3,(6+((i-1)*2))));
%ploting data on timeline axis.
plot(C{i},A{i},[linestyles{i} colors(i)],'LineWidth',1.2);

hold on
end

legend(L,'Location','westoutside','Fontsize',12,'FontWeight','bold');
legend('boxoff');
ylabel(strings(5,1),'Fontsize',25);

%size of time array.
s_time=size(B{1}(pos):size_time_interval(3):B{2}(pos_end));
s_time=s_time(2);

iteration=0;
for j=1:s_time;
    %location of timeline.
mov_point=time_interval(j);
l=line([mov_point mov_point],get(hax,'YLim'),'Color',[0 0
0],'LineWidth',3);

%saving figure.
T = time_interval(j);
title(strcat('Time:', num2str(T),
's'),'FontSize',25,'FontName','Arial');

export_fig(gcf,sprintf('movie_%d.png',iteration),'-r100');
iteration=iteration+1;

delete(l);
hold off
```

```
end

else
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Facility visualization.

clear all
clc
%location for visualization axis.
left=0.24;
bottom=0.066;
width=0.55;
height=.85;
%minimum values for background image.
min_x=0;
min_y=0;

%Excel data file prompt.
file_name=inputdlg('what is the excel file name? ','data file');
[data_points,strings]=xlsread(file_name{1});
[length_colums,length_row]=size(data_points);
j=1:(length_colums-3);
i=1:data_points(3,1);
%time vector.
time_v=data_points(4:end,4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%time interval.

%prompt change interval.
prompt_time_interval={'time start ','time end','time step'};
dlg_title_time_interval='time interval ';
size_time_interval=inputdlg(prompt_time_interval,
dlg_title_time_interval);
size_time_interval=str2double(size_time_interval);
time_interval=size_time_interval(1):size_time_interval(3):size_time_in
terval(2);

% Create figure.
figure1 = figure;
set(figure1,'Units','Normalized','OuterPosition',[0 0 1 1]);
set(figure1,'Color',[1 1 1]);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create axes 1 for visualization.
axes1 = axes('Parent',figure1,...
    'Position',[left bottom width height]);
box(axes1,'off');

%prompt size along axes.
```

```
prompt_size={'minimum x axis ','maximum x axis','mimimum y
axis','maximum y axis'};
dlg_title_size='size ';
size_background=inputdlg(prompt_size, dlg_title_size);
size_background=str2double(size_background);

%prompt for background image change.
background_change=inputdlg('Do you wish to change the background
image? y/n','background');
str_comparison_background=strcmp(background_change,'y');

if str_comparison_background==1
%image background prompt.
background=inputdlg('Background:name of the file with extension?
','background');
img=imread(background{1});
imagesc([size_background(1) size_background(2)], [size_background(3)
size_background(4)], flipdim(img,1));
set(gca,'ydir','normal')
hold on
else
end

%save axis position and ratio when adding the colorbar.
p=get(gca,'position');
colorbar;
set(gca,'position',p);

%colorbar range change prompt.
color_bar_change=inputdlg('Do you wish to change the extremums of the
colorbar? y/n','colorbar');
str_comparison_colorbar=strcmp(color_bar_change,'y');

% Create colorbar and colorbar range.
format long
colormap('jet(255)');
colorbar('peer',axes1);
ylabel(colorbar,strings(5,1),'Fontsize',18);

%if loop for colorbar range .
if str_comparison_colorbar==1
prompt_colbar={'minimum value','maximum value'};
title_colbar='colorbar extremums';
colbar_size=inputdlg(prompt_colbar,title_colbar);
colbar_size=str2double(colbar_size);
caxis([colbar_size(1) colbar_size(2)]);
else
colbar_size=[min(min(data_points)) max(max(data_points))];
caxis([colbar_size(1) colbar_size(2)]);
end


for i=1:data_points(3,1)
```

```
     %save time data in array.
     B{i}=data_points(4:end,(5+((i-1)*2)));
     %time interval data.
     [t_start pos]=min(abs(size_time_interval(1)-B{i}));

     [t_end pos_end]=min(abs(size_time_interval(2)-B{i}));
   C{i}=B{i}(pos):size_time_interval(3):B{i}(pos_end);
    %save data in array.

A{i}=data_points(B{i}(pos)+3:size_time_interval(3):B{i}(pos_end)+3,(6+
((i-1)*2)));
nan_int=isnan(A{i});
A{i}(nan_int)=interp1(C{i}(~nan_int),A{i}(~nan_int),time_interval(nan_
int));

box{i}=annotation(figure1,'rectangle',...
    [left+data_points(1,(6+((i-1)*2)))*width-0.01
bottom+data_points(2,(6+((i-1)*2)))*height-0.02 0.02 0.04]);

measurement_nbr{i}=annotation(figure1,'textbox',...
    [left+data_points(1,(6+((i-1)*2)))*width-0.02
bottom+data_points(2,(6+((i-1)*2)))*height-0.02 0.02 0.04],...
    'String',num2str(i),...
    'FitBoxToText','off',...
    'EdgeColor','none');
hold on
end
%size of time array.
s_time=size(B{1}(pos):size_time_interval(3):B{2}(pos_end));
s_time=s_time(2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

iteration=0;
for j=1:s_time;
for i=1:data_points(3,1);
hold on
%save rgb value corresponding to each data value.
d_perc=(A{i}(j)-colbar_size(1))/(colbar_size(2)-colbar_size(1));
format long
cmap=colormap('jet(255)');
x_rgb=d_perc*255;
x_rgb=round(x_rgb);
if x_rgb==0;
    x_rgb=1;
else
end

if isnan(x_rgb)==1
    y_rgb{i}=[1 1 1];
else
y_rgb{i}=cmap(x_rgb,:);
end
```

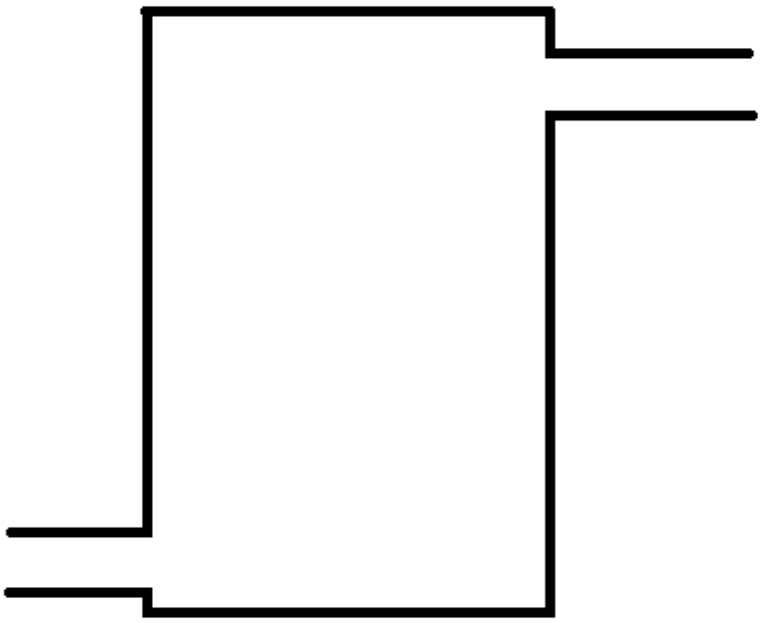
```
end

for i=1:data_points(3,1);

%change the facecolor of each square in axis 1.
box{i}=annotation(figure1,'rectangle',...
    [left+data_points(1,(6+((i-1)*2)))*width-0.01
bottom+data_points(2,(6+((i-1)*2)))*height-0.02 0.02 0.04],...
    'FaceColor',y_rgb{i});
var_value{i}=annotation(figure1,'textbox',...
    [left+data_points(1,(6+((i-1)*2)))*width+0.01
bottom+data_points(2,(6+((i-1)*2)))*height-0.02 0.04 0.04],...

'String',strcat(num2str(A{i}(j)),strings(5,3)),'FontWeight','bold',...
    'FitBoxToText','on',...
    'EdgeColor','none',...
    'Tag','values');

end
%saving figure.
T = time_interval(j);
title(strcat('Time:',num2str(T),
's'),'FontSize',25,'FontName','Arial');

export_fig(gcf,sprintf('movie_%d.png',iteration),'-r100');
iteration=iteration+1;
delete(findall(gcf,'Tag','values'));
hold off
end

end
```

APPENDIX 2. MICROSOFT EXCEL TEMPLATE

		Location x		0,3		0,4		0,5
nbr of measurements		location y		0,3		0,5		0,7
3		**time**	**t1**	**variable1**	**t2**	**variable2**	**t3**	**variable3**
variable measured	unit	**0**						
Temperature	**C**	**1**	**1**	**4**	**1**		**1**	
Time	**s**	**2**	**2**	**5**	**2**		**2**	
		3	**3**	**8**	**3**	**32**	**3**	
		4	**4**	**33**	**4**	**45**	**4**	
		5	**5**	**18**	**5**	**46**	**5**	**36**
		6	**6**	**24**	**6**	**hello**	**6**	**50**
		7	**7**	**38**	**7**	**78**	**7**	**54**
		8	**8**	**46**	**8**	**58**	**8**	**1**
		9	**9**	**4**	**9**	**62**	**9**	**45**
		10	**10**	**5**	**10**	**48**	**10**	**45**
		11	**11**	**8**	**11**	**32**	**11**	**25**
		12	**12**	**33**	**12**	**45**	**12**	**86**
		13	**13**	**18**	**13**	**46**	**13**	**36**
		14	**14**	**24**	**14**	**hello**	**14**	**50**
		15	**15**	**38**	**15**	**78**	**15**	**54**
		16	**16**	**46**	**16**	**58**	**16**	**1**
		17	**17**	**4**	**17**	**62**	**17**	**45**
		18	**18**	**5**	**18**	**48**	**18**	**45**
		19	**19**	**8**	**19**	**32**	**19**	**25**
		20	**20**	**33**	**20**	**45**	**20**	**86**
		21	**21**	**18**	**21**	**46**	**21**	**36**
		22	**22**	**24**	**22**	**hello**	**22**	**50**
		23	**23**	**38**	**23**	**78**	**23**	**54**
		24	**24**	**46**	**24**	**58**	**24**	**1**
		25	**25**	**4**	**25**	**62**	**25**	**45**
		26	**26**	**5**	**26**	**48**	**26**	**45**
		27	**27**	**8**	**27**	**32**	**27**	**25**
		28	**28**	**33**	**28**	**45**	**28**	**37**
		29	**29**	**18**	**29**	**46**	**29**	**36**
		30	**30**	**24**	**30**		**30**	**50**
		31	**31**	**38**	**31**		**31**	**54**
		32	**32**	**46**	**32**		**32**	**67**
		33	**33**	**4**	**33**		**33**	**45**

34	34	5	34		34	45
35	35	8	35		35	25
36	36	33	36		36	86
37	37	18	37	46	37	36
38	38	24	38	hello	38	50
39	39	38	39	78	39	54
40	40	46	40	58	40	68
41	41	4	41	62	41	45
42	42	5	42	48	42	45
43	43	8	43	32	43	25
44	44	33	44	45	44	86
45	45	18	45	46	45	
46	46	24	46	hello	46	
47	47	38	47	78	47	
48	48	46	48	58	48	
49	49	4	49	62	49	
50	50	5	50	48	50	

APPENDIX 3. FACILITY IMAGE USED



The facility is drawn on a 500 by 500 pixel square and saved as a .PNG file.