Summer Student projects reports

Student: Eradzh Rakhmatov
Branch of Lomonosov Moscow State University, Dushanbe, Tajikistan
Supervisors: Yisel Martinez Palenzuela
Sebastian Rothe

Internship dates: Jun 26 - Aug 18 2017

1 Introduction

Throughout my stay as a Non Member State Summer Student at the European Organization for Nuclear Research (CERN) I have been working on the projects described in the following pages.

2 Extraction of MR-TOF data.

My task was to automate the process of extracting device parameters from the CERN Logging Database. Currently, there are two ways to do it. First, to use Java API and optionally a generic Graphical User Interface (called Timber). Second way to extract the data is using a pytimber package, which is a Python library that allows to access the data stored in the CERN Logging Database.

Thus, there was an excel file with missing device parameters values and my task was to extract the missing data.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	#	Data taken	beamtime	Data evaluated	Filename	A/q	Isotope	target number	target current	line current	proton current	triggered on p+?	stacks after p+	Number of peaks	half life / s	MR-ToF revolutions	trapping time / us
2	1	06/21/2015 03:18:37.037	IS535	30.01.2017	Cu-run103	79	79Ga					n		3	2.847	1000	21376.125
3	2	06/21/2015 03:18:37.037	IS535	30.01.2017	Cu-run103	79	65Cu12C1H1H					n		3	stable	1000	21376.125
4	3	06/21/2015 03:18:37.037	IS535	30.01.2017	Cu-run103	79	79Cu					n		3	0.22	1000	21376.125
5	5	06/21/2015 03:38:38.930	IS535	30.01.2017	Cu-run104	79	79Ga					n		3	2.847	1000	21376.125
6	6	06/21/2015 03:38:38.930	IS535	30.01.2017	Cu-run104	79	65Cu12C1H1H					n		2	stable	1000	21376.125
7	7	06/21/2015 03:40:47.353	IS535	30.01.2017	Cu-run105	79	79Ga					n		3	2.847	1000	21376.125
8	8	06/21/2015 03:40:47.353	IS535	30.01.2017	Cu-run105	79	65Cu12C1H1H					n		3	stable	1000	21376.125
9	9	06/21/2015 05:49:12.254	IS535	30.01.2017	Cu-run108	79	79Rb					n		4	1374	1000	21376.125
10	10	06/21/2015 05:49:12.254	IS535	30.01.2017	Cu-run108	79	79Ga					n		4	2.847	1000	21376.125
11	11	06/21/2015 05:49:12.254	IS535	30.01.2017	Cu-run108	79	65Cu12C1H1H					n		4	stable	1000	21376.125
12	12	06/21/2015 05:49:12.254	IS535	30.01.2017	Cu-run108	79	79Cu					n		4	0.22	1000	21376.125
13	13	06/21/2015 07:08:18.619	IS535	30.01.2017	Cu-run112	79	79Rb					y	6	5	1374	700	14961.875
14	14	06/21/2015 07:08:18.619	IS535	30.01.2017	Cu-run112	79	79Ga					y	6	5	2.847	700	14961.875
15	15	06/21/2015 07:08:18.619	IS535	30.01.2017	Cu-run112	79	65Cu12C1H1H					y	6	5	stable	700	14961.875
16	16	06/21/2015 07:08:18.619	IS535	30.01.2017	Cu-run112	79	79Cu					y	6	5	0.22	700	14961.875
17	17	06/21/2015 15:27:10.945	IS535	30.01.2017	Cu-run130	79	79Rb					y	15	5	1374	1000	21374.125
18	18	06/21/2015 15:27:10.945	IS535	30.01.2017	Cu-run130	79	79Ga					y	15	5	2.847	1000	21374.125
19	19	06/21/2015 15:27:10.945	IS535	30.01.2017	Cu-run130	79	65Cu12C1H1H					y	15	5	stable	1000	21374.125
20	20	06/21/2015 15:27:10.945	IS535	30.01.2017	Cu-run130	79	79Cu					y	15	5	0.22	1000	21374.125
21	21	06/22/2015 14:18:09.966	IS535	30.01.2017	Cu-run167	80	80Rb					y	8	3	33.4	700	15057
22	22	06/22/2015 14:18:09.966	IS535	30.01.2017	Cu-run167	80	80Ga					y	8	3	2.03	700	15057
23	23	06/20/2015 20:28:01.299	IS535	30.01.2017	Cu-run091	78	78Ge					y	20	4	5280	1000	21238.9
24	24	06/20/2015 20:28:01.299	IS535	30.01.2017	Cu-run091	78	78Rb					y	20	4	1056	1000	21238.9
25	25	06/20/2015 20:28:01.299	IS535	30.01.2017	Cu-run091	78	78Ga					y	20	4	5.09	1000	21238.9
26	26	06/20/2015 20:28:01.299	IS535	30.01.2017	Cu-run091	78	78Cu					y	20	4	0.335	1000	21238.9

Figure 1: Yield MR-TOF file with missing device parameters values

The script works in the following way:

1. Timestamps are read from *timestamp.txt*
2. Device parameters are read from *parameters.txt*
3. Extracted data is written to *currentTimestamp.csv* file with standard deviations in percentage (the percentage standard deviation makes of values at zero point timestamp) that are computed using parameter values fluctuations in the window of 5 minutes margin around timestamp.

	A	B	C	D	E
1	Timestamp	YGPS.TARGET-HEAT:I_MEAS	STD in %	YGPS.LINE-HEAT:I_MEAS	STD in %
2	06/20/2015 20:28:01.299000	604.17	0	273.6	0
3	04/21/2015 15:27:10.945000	453.7	0	343.1	0
4	06/30/2015 20:28:01.299000	604.17	0	333.3	0.210021
5	06/20/2016 20:28:01.299000	432.9	0	302.8	1.88E-14
6					

Figure 2: Final *currentTimestamp.csv* file

2.1 SWAN and Pytimber

Although, I have installed python and all the necessary packages to my computer, it is also possible to run the python script on SWAN (Service for Web based Analysis). SWAN is a platform to perform interactive data analysis in the cloud so there is no need to install necessary packages locally.

The way pytimber works is rather simple.

```
1  import pytimber
2  ldb = pytimber.LoggingDB()
3  data=ldb.get('YGPS.LINE-HEAT:I_MEAS', '2017-08-17 10:00:01.299')
4  print(data)
5
```

Figure 3: Extracting data using pytimber

The code above returns the python dictionary where the key is a device parameter with lists of timestamps and values as key values.

```
{'YGPS.LINE-HEAT:I_MEAS': (array([  1.50295677e+09]), array([ 331.9]))}
```

Figure 4: Extraction result

3 Pressure readout from XGS-600 Gauge Controller

The pressure readout was implemented in two ways.

3.1 Using labview

As shown Fig.5, the labview program commicates with gauge controller through RS-232 port and writes a command #000F to it to dump all pressure values. Afterwards, the values are read and displayed with timestamps in the respective windows. This labview program will be used in the near future by David

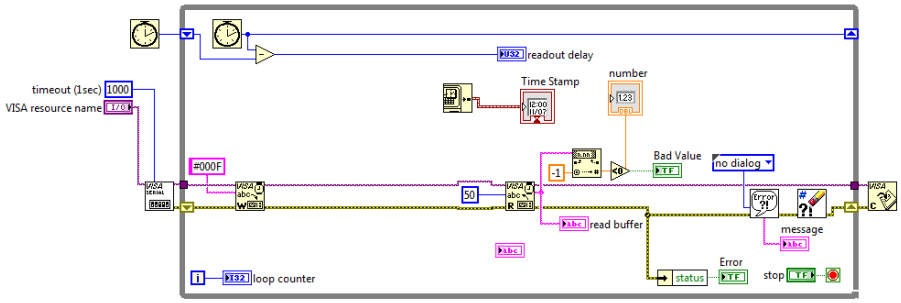

Figure 5: Labview block diagram

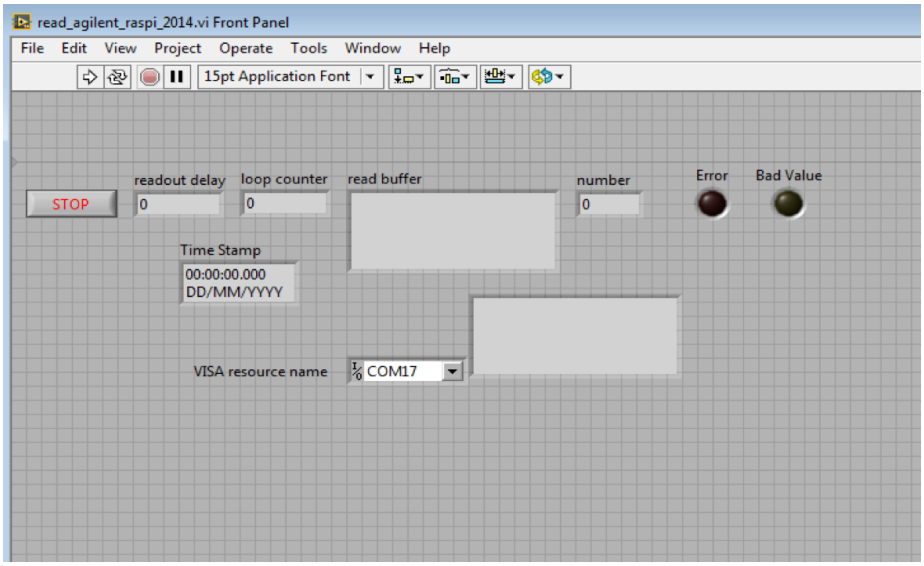

Figure 6: Labview front panel

Leimbach, who is doing his master thesis at CERN, in his experiment to measure electron affinity of At. Also, it can potentially be used in offline ISOLDE.

3.2 Using python on raspberrypi

Python script has been written to read presssure values from RS-232 port and plot it online over timestamp using plot.ly api. In addition, the script also writes the data with timestamp to the local text file whose name is *timestamp.txt*

I tried to make changes to raspberrypi's configurations to run the python script automatically everytime it is rebooted, however, the issue encountered was that Plot API does not automatically fill in the API keys, which are needed to plot data online, as it should. Therefore, currently, after each reboot the user has to remotely connect to raspberrypi and change API keys that are generated on the plot.ly website.

This is currently being used in CaO production experiment implemented by Joao Pedro Ramos.

```
07/08/2017 14:59:14    0.0   11.33
07/08/2017 14:59:18    0.0   11.33
07/08/2017 14:59:21    0.0   11.31
07/08/2017 14:59:24    0.0   11.3
07/08/2017 14:59:27    0.0   11.29
07/08/2017 14:59:30    0.0   11.27
07/08/2017 14:59:33    0.0   11.26
07/08/2017 14:59:36    0.0   11.25
07/08/2017 14:59:39    0.0   11.23
07/08/2017 14:59:42    0.0   11.21
07/08/2017 14:59:45    0.0   11.19
07/08/2017 14:59:48    0.0   11.16
07/08/2017 14:59:52    0.0   11.15
07/08/2017 14:59:55    0.0   11.6
07/08/2017 14:59:58    0.0   4.354
07/08/2017 15:00:01    0.0   4.374
07/08/2017 15:00:04    0.0   3.368
07/08/2017 15:00:07    0.0   2.988
07/08/2017 15:00:10    0.0   2.997
07/08/2017 15:00:13    0.0   3.002
07/08/2017 15:00:16    0.0   3.015
07/08/2017 15:00:19    0.0   2.14
07/08/2017 15:00:23    0.0   1.049
07/08/2017 15:00:26    0.0   0.4045
07/08/2017 15:00:29    0.0   0.1709
07/08/2017 15:00:32    0.0   0.09057
07/08/2017 15:00:35    0.0   0.04998
07/08/2017 15:00:38    0.0   0.02256
07/08/2017 15:00:41    0.0   0.003275
07/08/2017 15:00:44    0.0   0.0001333
07/08/2017 15:00:47    0.0   0.0001333
07/08/2017 15:00:50    0.0   0.0001333
07/08/2017 15:00:54    0.0   0.0001333
07/08/2017 15:00:57    0.0   0.0001333
```

Figure 7: *timestamp.txt* file as created on raspberry pi

4 Visual target documentation

Tecnichians who work in target and ion source production often take photos of the targets because there are always differences in a way each one is produced.

The idea was to automate the process of taking pictures by running a script on raspberrypi that would control 2-3 web cameras connected to it, which are fixed on a tripod, and take photos every few seconds. At the same time, the target can be placed on a slowly rotating table. Afterwards, the pictures can be saved to easily accessable public folder.

To take the idea further, it is possible to push the pictures to Autodesk Remake software to create 3D image from them. If accuracy is high enough, the model can be used to evaluate the distance between arbitrary elements of the target when necessary.

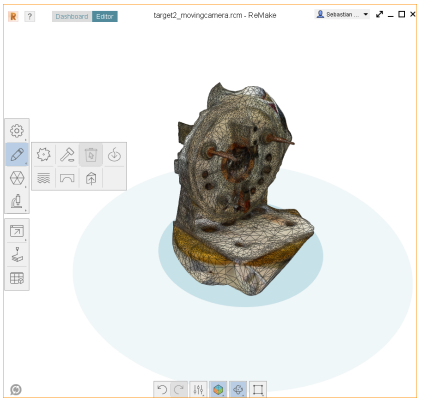

Figure 8: First version of 3D model

On the Fig.8, we can see the first version of the 3D model of the target, which was created by taking photos with only one web camera. This result can be improved by using more cameras, taking more pictures and organizing a more favourable surrounding.

5 Acknowledgements

I would like to thank Sebastian Rothe for his guidance throughout the internship, Yisel Martinez Palenzuela for showing me around CERN the first few days, as well as Joao Pedro Ramos and David Leimbach for helping me to develop a better understanding of their experiments.