ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-THPV040

NEW MACHINE LEARNING MODEL APPLICATION FOR THE AUTOMATIC LHC COLLIMATOR BEAM-BASED ALIGNMENT

G. Azzopardi[*,1], G. Ricci[1,2]
[1] CERN, Geneva, Switzerland,
[2] Sapienza Università di Roma

Abstract

A collimation system is installed in the Large Hadron Collider (LHC) to protect its sensitive equipment from unavoidable beam losses. An alignment procedure determines the settings of each collimator, by moving the collimator jaws towards the beam until a characteristic loss pattern, consisting of a sharp rise followed by a slow decay, is observed in downstream beam loss monitors. This indicates that the collimator jaw intercepted the reference beam halo and is thus aligned to the beam. The latest alignment software introduced in 2018 relies on supervised machine learning (ML) to detect such spike patterns in real-time. This enables the automatic alignment of the collimators, with a significant reduction in the alignment time. This paper analyses the first-use performance of this new software focusing on solutions to the identified bottleneck caused by waiting a fixed duration of time when detecting spikes. It is proposed to replace the supervised ML model with a Long-Short Term Memory model able to detect spikes in time windows of varying lengths, waiting for a variable duration of time determined by the spike itself. This will allow for further speeding up the automatic alignment.

INTRODUCTION

The CERN Large Hadron Collider (LHC) is the largest particle accelerator in the world, built to accelerate and collide two counter-rotating beams towards the unprecedented design center-of-mass energy of 14 TeV [1]. The LHC is susceptible to beam losses which can damage the state of superconductivity of its magnets [2]. A multi-stage collimation system, consisting of 123 collimators [3], is installed in the LHC. Each collimator consists of two parallel absorbing blocks, referred to as jaws, inside a vacuum tank. The collimators must be aligned with the beam by symmetrically positioning the jaws on either side. This provides a 99.998 % cleaning efficiency of halo particles, preventing any LHC damage [4]. Each year of LHC operation begins with a commissioning phase which involves aligning all collimators and ensuring the correct settings for nominal operation [5].

This paper presents an analysis of first-use performance of the latest alignment software, that makes use of machine learning. A comparison with the previous alignment software resulted in identifying a bottleneck that restricts alignment efficiency. This is followed by a detailed analysis of a Long-Short Term Memory model that can be introduced to further improve the performance of the new software.

* gabriella.azzopardi@cern.ch

BACKGROUND

Recap. of Collimator Alignments

Collimation alignment at the LHC is essential for beam performance and is based on different beam-based techniques developed for the specific LHC conditions [6]. While the new generation of collimators feature a design with embedded beam position monitors for a rapid alignment to the circulating beam [7], most of the LHC collimators do not have this feature. For the latter, the alignment relies on dedicated Beam Loss Monitoring (BLM) devices positioned outside the beam vacuum, immediately downstream from each collimator [4].

Collimator jaws are moved towards the beam with a step precision of 5 μm, and the BLMs are used to detect beam losses generated when halo particles impact the collimator jaws. The recorded losses are proportional to the amount of beam intercepted and are measured in units of Gy/s. Collimators are aligned with respect to a reference halo cut generated with the primary collimators. A collimator jaw is considered aligned when a movement produces a clear beam loss spike in the BLM [8]. The observation time to evaluate the quality of the signal and to assess if the spike corresponds to a correct alignment can vary from < 1 s to > 10 s depending on the machine conditions and beam properties. Aligning collimators with BLMs is referred to as the beam-based alignment (BBA), which involves aligning collimators one by one, by moving one jaw at a time towards the beam.

Before moving each jaw, the losses produced by the previous alignment must have decayed in order to decrease possible cross-talk effects between the collimators, whereby the BLM losses at a specific collimator are affected by the signal produced by other collimators around the LHC [9]. A complete alignment campaign at the LHC requires moving each collimator jaw several times, which can produce more than 1000 observation spikes. Therefore, improving the time needed to classify these spikes has a direct impact on the system's alignment time.

Semi-Automatic Beam-based Alignment

Since 2011, LHC collimators have been aligned using a semi-automatic procedure. This involves having the user to select the collimator to align, including the required settings and BLM threshold. The collimator will then automatically move towards the beam until the BLM losses exceed the threshold selected. At this point, the collimator automatically stops moving and the user must determine whether the collimator is aligned or not by classifying the loss spike recorded in the BLM signal.

Fully-Automatic Beam-based Alignment

The fully-automatic alignment was introduced and used in 2018 for all collimator alignments. As the name suggests, this fully-automates the entire procedure by automating the user's tasks in the semi-automatic alignment [10]. The three main user tasks have each been replaced with dedicated algorithms, such that:

- The collimator to align is automatically selected to avoid cross-talk (if any) [11].
- The BLM threshold (to stop the jaw movement during alignment) is automatically selected based on the real-time losses detected at the collimator [12].
- The BLM loss spikes are automatically classified using supervised machine learning into two classes; alignment spike or spurious spike [13]. This is possible by waiting a fixed duration of time to extract the features required for classification, from the BLM loss spike. Based on experience, the classification is set to wait 4 s at injection and 6 s at flat top.

These algorithms are developed as individual modules within the fully-automatic alignment software package [14], allowing for any improvements/upgrades. This paper focuses on the upgrade of the machine learning module.

ANALYSIS OF SEMI- AND FULLY-AUTOMATIC BBA

Data were collected from collimator alignments performed using the semi-automatic alignment software in 2016 injection commissioning and the latest parallel fully-automatic alignment software in 2018, during a dedicated beam test replicating injection commissioning conditions [15]. The logged data includes the alignment of 75 collimators in 2016 and 77 collimators in 2018, both at a frequency of 1 Hz. Table 1 lists the details of the two alignment campaigns, resulting in the fully-automatic procedure able to align the collimators at injection in one third of the time required by the semi-automatic one [16].

One can observe that the moving time of the collimator jaws during both campaigns is approximately 38 % of the total time (±10 % assuming a 0.5 s error on each jaw alignment due to the 1 Hz logging precision). This indicates that the fully-automatic alignment can be further sped up by decreasing the waiting time (62 %). The main contribution to the waiting time is the spike classification which is currently set to wait a fixed 4 s, at injection, for the BLM signal to decay before classifying it.

Decay Time Analysis

The actual time required for the BLM signal to decay at injection and flat top was analysed to identify possible gains in the overall alignment time by reducing the duration of

Table 1: Details of two alignment campaigns at injection using the semi-automatic alignment in 2016 and the fully-automatic alignment in 2018.

	Semi-Automatic	**Fully-Automatic**
Collimators	75	77
Total time	2h 31m 59s	49m 17s
Moving time	58m 13s	18m 14s
Total alignments	1903	637
Moving time	38.3 %	38.0 %
Alignments/Coll	25.37	8.27

the decay observation. The decay time was analysed for 1550 alignment spikes collected during 2016-2018 alignment campaigns, which includes 719 alignment spikes at injection and 831 at flat top.

The decay rate is modelled as an exponentially falling distribution, with optimal losses achieved after 6 half-lives, as shown in Figure 1. As a result the optimal decay time is the latency required for the BLM losses to fall to 1/64 of the maximum value. Figure 2 displays the distribution of the decay latency at the two machine states such that the mean decay time is 0.61 s at injection and 2 s at flat top.

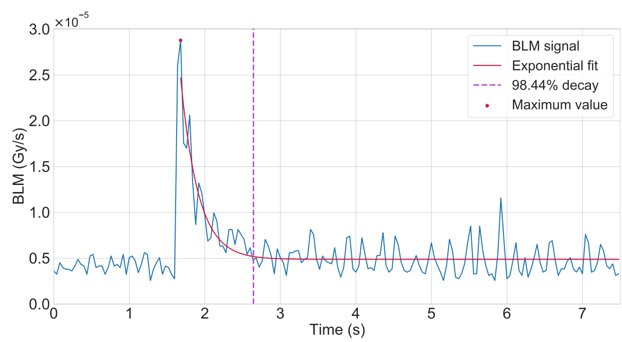

Figure 1: An example of a short decay observed in the BLM signal of an alignment spike at flat top.

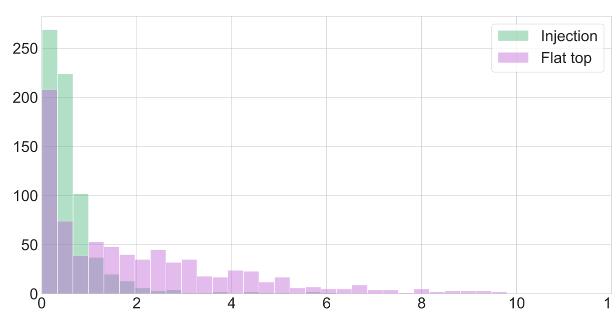

Figure 2: BLM signal decay time distributions at injection and flat top.

This highlights the fact that a dynamic adjustment of the observation time can speed up the overall alignment, as it is

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-THPV040

not necessary to consistently wait 4 s at injection or 6 s at flat top. As an example, Figure 1 displays an alignment spike at flat top with a short decay that lasts 0.96 s. Moreover, this will allow for adjusting the observation window in the case of longer decays, although they do not happen frequently.

LSTM-RNN FOR SPIKE CLASSIFICATION

In order to continuously classify spikes in real-time and automatically detect their decay, the proposed solution is to train a Long Short-Term Memory (LSTM) - Recurrent Neural Network (RNN) [17]. This may enable the spike classification after a variable duration and minimize the waiting time of the automatic collimator alignment.

The entire data set gathered during 2016-2018 consists of 2973 samples collected when the BLM losses exceeded the predefined threshold, i.e. the moment when the observation of the BLM signal starts. The collected data samples were individually analysed and labelled into the classification classes; 1550 alignment spikes and 1423 spurious spikes. Each sample is a 7.5 s time series of length 188 and contains two signals; the BLM signal logged at a frequency of 25 Hz and the collimator jaw positions logged at a frequency of 1 Hz.

The input used to train the LSTM combines the two signals in each sample by scaling the BLM signal with the collimator position in sigma at the time the threshold was exceeded. Z-Score ($z = \frac{x-\mu}{\sigma}$) is then used as the normalization technique to re-scale the features such that they have the properties of a standard normal distribution.

The network architecture was developed using the deep learning library Keras [18], with TensorFlow [19] as the back-end, see Figure 3. It consists of two LSTM layers, followed by a dropout and dense layer. The model was trained over 75 epochs using an Adam optimizer [20] with a learning rate of $3e^{-4}$. Binary cross-entropy is used as the loss function.

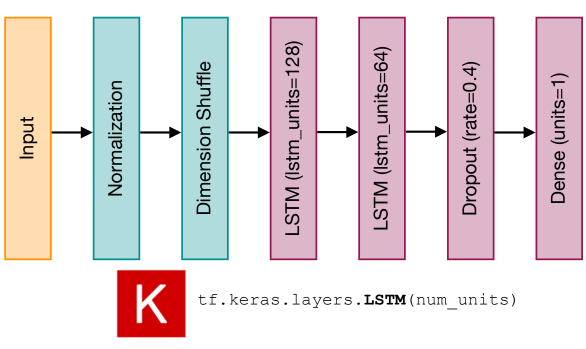

Figure 3: The LSTM network architecture. Two layers containing 128 and 64 hidden neurons, respectively, process the learning data produced by the preprocessing unit. Following is a dropout layer with rate of 0.4, then a dense layer with one neuron and sigmoid activation function.

The results of the two LSTM layers were outputted solely on the final time step, such that each layer generated a 2D array: Layer 1 output size: 188 × 128, Layer 2 output size: 64 × 1. Following this, the dropout layer outputs a 1D array of size 64, and finally the dense layer outputs the final probability used for spike classification.

In order to ensure the correct alignment of collimators, false detection of an alignment spike is more grievous than not detecting an alignment spike. Therefore, precision is used as the main performance metric to avoid false positives [13].

The results are collected over a 10-fold cross-validation randomly stratified 30 times, to handle lucky splits. The results are displayed in Figure 4, highlighting that the train and test loss curves stabilized with a minimal gap between the final values, thus indicating that a good fit has been found after 75 epochs. Overall, the model obtained an average precision of 94 % on the testing sets (and 94 % accuracy).

This precision was calculated by evaluating the classification probability at the end of the available sample, whereby a classification score with a probability larger than 50 % is classified as an alignment spike. Further analysis will aid to determine the best moment to predict the spike class and the ideal probability threshold, to possibly improve the model's precision.

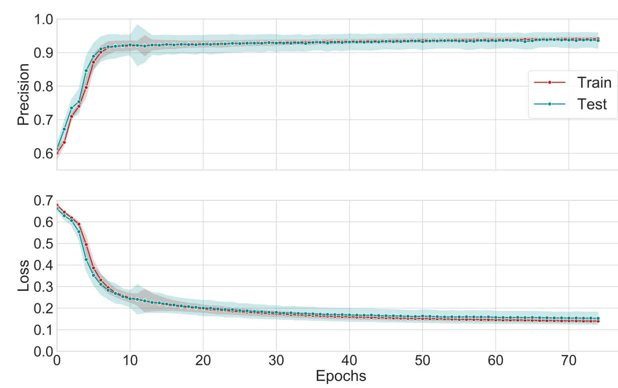

Figure 4: Loss and precision curves obtained by the LSTM on the training and testing data, in terms of mean and standard deviation.

Spike Classification Analysis

The trained LSTM model was used to continuously classify each sample at each time step, starting from the moment when the collimator stopped moving, until the end of the 7.5 s window.

An overview of the classification probabilities obtained for the two spike classes at injection is displayed in Figure 5. A clear distinction can be made between the spike classes at a latency of ~1.5 s, at which point the probability gradient for alignment spikes falls below 0.2. Taking a closer look at the classification probabilities of the two classes at a latency of 1.5 s in Figure 6, one can observe that there is no overlap at 80 % probability. Moreover, Figure 7 displays the latency required to obtain the maximum probability for each spike class. One can observe that the ~98% of spurious spikes obtain their maximum probability within the first 1.5 s, whilst

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-THPV040

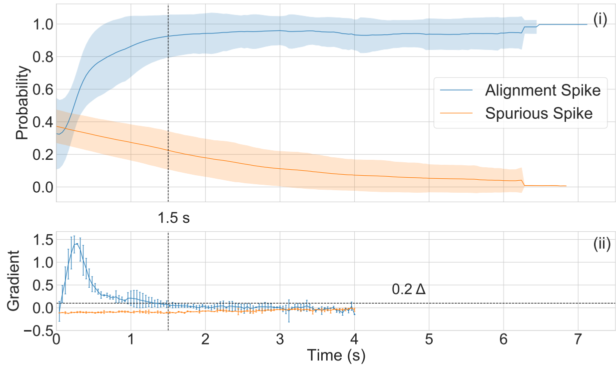

Figure 5: Spike classification results displayed as (i) the evolution of the classification probabilities in time for the two machine states, and (ii) the gradient of change in the classification probabilities. The two plots are displayed in terms of mean and standard deviation.

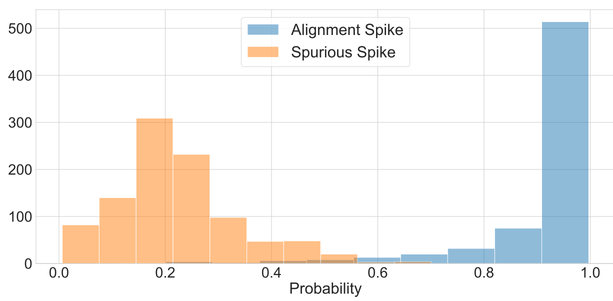

Figure 6: The probabilities obtained by the two spike classes at 1.5 s latency after the collimator stopped moving.

all spurious spikes remain below the 80 % classification probability threshold.

As a result, the LSTM can be set to make a classification once the probability gradient decreases below 0.2, which, as shown in Figure 5, occurs within a latency of ~1-1.5 s, at injection. If the probability at this point is below 80 %, then the BLM losses can be classified as a spurious spike, and the next alignment can begin. When the losses form an alignment spike, the next step is to determine the optimal BLM decay time to begin the next alignment. In this case an exponential function can be fit (as shown in Figure 1) to determine when ~98.5 % of the BLM signal decays (6 half-lives), which on average would have already decayed (mean of 0.61 s).

An analogous analysis was performed at flat top, showing similar results, i.e. classifications can rely on the 80 % probability threshold when the probability gradient decreases below 0.2, which at this machine state occurs within a latency of ~1.5-2 s.

Classification Results

Classifying the BLM signals as proposed by the presented analysis increases the classification precision on the data set to 98 % (with 90 % accuracy). In addition, the time taken for the LSTM to classify the data set into the two spike classes resulted in a mean latency of 1.07 s at injection and 1.54 s at flat top, whereby 88 % of the observation spikes were classified within these times. Table 2 summarises the classification results at the two machine states, indicating a factor of 4 speed-up compared to the present implementation using supervised ML with fixed observation times.

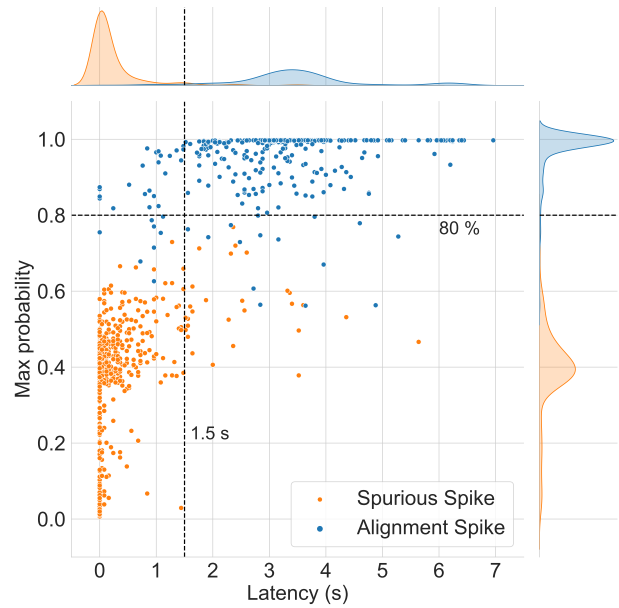

Figure 7: The distribution of the maximum probabilities achieved by the two spike classes and the required latency. The majority of alignment spikes obtain a probability above 80 % whereas all spurious spikes remain below.

Table 2: The latency results obtained when classifying the data set using the LSTM model. The classification was started at 1 s for injection and 1.5 s for flat top.

State	Start time	Mean	Stand. dev.	Max.
Injection	1 s	1.07 s	0.1 s	1.72 s
Flat top	1.5 s	1.54 s	0.06 s	2.04 s

Results with Ion Beams A data set was collated with 254 samples (192 alignment spikes, 62 spurious spikes) from the ion run in 2018 during the collimator alignment campaign in collisions. The mean decay time for the BLM losses to fall to 1/64 of the maximum value is 1.08 s.

An analogous analysis was performed with ion beams resulting in a similar environment for classifications, such that an 80 % probability threshold and 0.2 probability gradient after a minimum of 2 s latency, are ideal. The LSTM model trained on proton beams was used to classify this data set and obtained 97 % precision (and 87 % accuracy). The mean latency required to classify these samples is 2.08 s.

A summary of the results obtained at the different machine configurations analysed in this paper, is collected in Table 3.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-THPV040

Table 3: Results obtained at different LHC configurations.

	Proton beams		Ion beams
	Injection	Flat top	Collisions
Decay time	~0.61 s	~2 s	~1.08 s
Prob. threshold	80 %	80 %	80 %
Prob. gradient	0.2	0.2	0.2
Class. latency	~1.07 s	~1.54 s	~2.08 s
Class. precision	**98 %**		**97 %**

THEORETICAL IMPROVEMENT OF ALIGNMENT TIME WITH LSTM

The time performance of the automatic BBA using supervised machine learning is displayed in Table 1, taking into consideration the following assumptions [11]:

- A clear alignment spike is achieved the first time the threshold is exceeded.
- Two clear alignment spikes are achieved after ~10 steps per jaw.
- The primary collimator was aligned in a previous alignment and both jaws achieved an alignment spike the first time the threshold is exceeded.

Table 4: Theoretical minimum time required to align a collimator [11].

Step	Action	Time (s)
1	Move both jaws to 4 mm	~8
2	Wait for losses to decay	x
3	Classification delay	1
4a 4b	Align Left Jaw Align Right Jaw	$2*(0.1 + x + 1)$ $2*(0.1 + x + 1)$
5a 5b	TCP before (Left Jaw) TCP before (Right Jaw)	$0.1 + x + 1$ $0.1 + x + 1$
6	TCP after (Left + Right)	$2*(0.1 + x + 1)$
Total	**17.8 + 9x**	@Inj $x >= 4$ @FT $x >= 6$

The proposed LSTM model is capable of dynamically classifying alignment spikes of varying lengths in real-time. Therefore, this will decrease the time waiting for the losses to decay (x_1) to an average of 1.07 s at injection and 2 s for alignment spikes at flat top. This indicates that the theoretical minimum time required to align a single collimator at injection is 27.43 s, assuming that every spike is an alignment spike. Therefore, on average, aligning 79 collimators at injection would require a minimum of 36.1 mins.

On the other hand, if the collimator jaws encounter one spurious spike in Steps 1, 4a and 4b, then an additional $3 * (0.1 + x_0 + 1)$ seconds are required per collimator, i.e. 1.07 s at injection and 1.54 s for spurious spikes at flat top. This results in an additional average of 6.51 s at injection, increasing the average time required in this case to 44.7 minutes.

Table 5: The average theoretical minimum time to sequentially align LHC collimators, calculated using Table 4.

Case studied	Supervised ML	LSTM-RNN
1 coll @Inj *+1 spurious spike*	53.8 s 69.1 s	27.43 s 33.94 s
79 colls @Inj *+1 spurious spike*	70.84 mins 90.98 mins	36.12 mins 44.69 mins
1 coll @FT *+1 spurious spike*	71.8 s 93.1 s	35.8 s 43.72 s
79 colls @FT *+1 spurious spike*	94.53 mins 122.58 mins	47.14 mins 57.56 mins

Table 5 summarises the theoretical minimum time required to sequentially align the collimator cases discussed, at injection and flat top. In 2018, the automatic alignment was upgraded to align the collimators in the two beams in parallel, resulting in 79 collimators aligned in 50 minutes at injection [15, 16]. Therefore the possible introduction of LSTM can theoretically align the collimators in ~24.56 minutes, speeding-up the automatic alignment by ~50 %.

CONCLUSION

The 123 LHC collimators are aligned automatically using supervised ML, provided by the latest fully-automatic software introduced in 2018. This paper analysed first-use performance of this software and identified a bottleneck caused by the fixed observation window used by the ML model to classify the BLM loss signal. This classification determines if the collimator jaws reached the correct alignment position.

In this paper a Long-Short Term Memory model was trained to continuously classify BLM signals. This allows classifying the losses into a spike class within 1-2 s of a collimator stopping its movement, once the rate of change in classification probabilities is below 0.2. Following each alignment spike classification, the suggestion is to fit an exponential function to the losses to determine whether the next alignment can begin. On average, the losses at injection would have already decayed (mean of 0.61 s), whereas decays at flat top may require a longer time (mean of 2 s).

This work allows for classifying BLM signals independent of whether the losses decayed or not, thus solving the bottleneck in the fully-automatic alignment process. Overall, this research could decrease the alignment time by ~50 %. The LSTM is readily available to be incorporated into the alignment software for testing during the LHC Run 3.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-THPV040

REFERENCES

[1] O. Brüning, *et al.*, "LHC design report", vol. 1, CERN, Geneva, Switzerland, 2004.

[2] C. Bracco, "Commissioning Scenarios and Tests for the LHC Collimation system", Ph.D. thesis, Ecole Polytechnique Federale de Lausanne, 2009.

[3] G. Azzopardi, *et al.*, "LHC Collimation Controls System for Run III Operation", in *Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems*, Shanghai, China, 2021, paper THPV012, this conference.

[4] G. Valentino, "Fast automatic beam-based alignment of the LHC collimator jaws", Ph.D. thesis, Univeristy of Malta, 2014.

[5] M. Lamont, "The LHC from commissioning to operation", in *Proceedings of International Particle Accelerator Conference*, San Sebastian, Spain, paper MOYAA01, pp. 11–15, 2011.

[6] R. W. Aßmann, *et al.*, "Expected performance and beam-based optimization of the LHC collimation system", in *Proceedings of the European Particle Accelerator Conference*, Lucerne, Switzerland, pp. 1825 – 1827, 2004.

[7] G. Valentino, *et al.*, "First operational experience with embedded collimator BPMs in the LHC", in *Proceedings of the International Particle Accelerator Conference*, Busan, Korea, paper WEPMW034, 2016.

[8] E. B. Holzer, *et al.*, "Beam loss monitoring system for the LHC", in *IEEE Nuclear Science Symposium Conference Record, 2005*, vol. 2, pp. 1052–1056, 2005. doi:10.1109/NSSMIC.2005.1596433

[9] G. Azzopardi, B. M. Salvachua Ferrando, G. Valentino, "Data-driven cross-talk modeling of beam losses in LHC collimators", in *Physical Review Accelerators and Beams*, vol. 22, no. 8, 2019.

[10] G. Azzopardi, G. Valentino, B. M. Salvachua Ferrando, S. Redaelli, A. Muscat, "Software Architecture for Automatic LHC Collimator Alignment using Machine Learning", in *Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems*, New York, NY, USA, paper MOCPL04, pp. 78–85, 2019.

[11] G. Azzopardi, "Automation of the LHC Collimator Beam-Based Alignment Procedure for Nominal Operation", Ph.D. thesis, Univeristy of Malta, 2019.

[12] G. Azzopardi, *et al.*, "Automatic Beam Loss Threshold Selection for LHC Collimator Alignment", in *Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems*, New York, NY, USA, paper MOPHA010, pp. 20–213, 2019.

[13] G. Azzopardi, G. Valentino, A. Muscat, B. M. Salvachua Ferrando, "Automatic spike detection in beam loss signals for LHC collimator alignment", in *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 934, pp. 10–18, 2019.

[14] G. Azzopardi, "The Automatic LHC Collimator Beam-Based Alignment Software Package", in *Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems*, Shanghai, China, paper WEPV016, this conference, 2021.

[15] G. Azzopardi, B. M. Salvachua Ferrando, G. Valentino, S. Redaelli, "Operational Results on the Fully-Automatic LHC Collimator Alignment", in *Physical Review Accelerators and Beams*, vol. 22, pp. 1208–1211, 2019.

[16] G. Azzopardi, B. M. Salvachua Ferrando, G. Valentino, S. Readelli, A. Muscat, "Operational Results of LHC collimator alignment using machine learning", in *Proceedings of the International Particle Accelerator Conference 2019*, Melbourne, Australia, 2019.

[17] S. Hochreiter, J. Schmidhuber, "Long short-term memory", in *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] F. Chollet, *et al.*, "Keras", 2015. Available at: https://github.com/fchollet/keras.

[19] M. Abadi, *et al.*, "Tensorflow: A system for large-scale machine learning", in *12th Symposium on Operating Systems Design and Implementation*, 2016.

[20] D. P. Kingma, J. L. Ba, "Adam: A method for stochastic optimization", arXiv preprint arXiv:1412.6980, 2014.