ISBN: 978-3-95450-247-9 ISSN: 2673-5490 doi: 10.18429/JACoW-IPAC2024-THPG36

CONTINUOUS POSITION ESTIMATION FOR THE FULL REMOTE ALIGNMENT SYSTEM OF THE HIGH LUMINOSITY LHC UPGRADE

Jürgen Gutekunst [*], Hélène Mainaud Durand, Guillaume Kautzmann,
Francis Klumb, Mateusz Sosin, CERN, Geneva, Switzerland

Abstract

The Full Remote Alignment System (FRAS) is a remotely controlled alignment system that comprises almost one thousand permanent sensors distributed along the 200 m of equipment that will be installed in the frame of the High Luminosity LHC (HL-LHC) project on either side of the ATLAS and CMS detectors. The sensors, along with their electronics and a system of motorized actuators, will be used to remotely adjust the relative positions of the components in real time, with no human intervention needed in the irradiated tunnel environment. In this contribution we describe the design and the implementation of the position estimation algorithm which is a core-component of the FRAS. This algorithm will process the data provided by all the sensors to determine exact positions and orientations of the associated components in real-time. The position estimation module is designed as a reusable C++ library and builds on the existing CERN Logiciel Général de Compensation (LGC), a modular least-square software. It will be fully integrated into the FRAS software stack and is entirely file-less during operation. We will demonstrate its performance in a realistic case study and showcase its ability to provide position updates on a much higher frequency than the required 1Hz.

INTRODUCTION

With the High Luminosity-LHC upgrade of the Large Hadron Collider (LHC) an increase of the instantaneous luminosities of a factor up to 5 is projected. This will be achieved mainly by replacing the focusing quadrupoles in the Long Straight Section (LSS) areound the ATLAS and CMS experimental Interaction Points (IPs). The FRAS implemented on the majority of elements to be installed in the LSS will allow:

- correction of misalignment after component installation and compensation of continuous ground movement, allowing the reduction of required corrector strengths,
- reduction of human intervention in the irradiated tunnel environment and optimization of physics time.

To this end, a position monitoring system with accuracies of 0.2 mm in vertical and 0.45 mm in radial direction is put into place. The support systems for the equipment, mainly the Universal Adjustment Platform (UAP) for components up to 2 tons and HL-LHC motorized jacks for heavier components [1, 2], will then provide the adjustment with micrometric resolution.

[*] juergen.gutekunst@cern.ch

This report describes the design and implementation of the estimation calculation module which has the crucial role to provide precise information on the current position and orientation of the involved components with minimal delay. The software library is also aimed to be a first step towards standardization and reproducibility in view of the growing demand for automated alignment solutions.

We start with the description of the basic mathematical background of the underlying least square estimation, before discussing the main features and the dynamic workflow of the developed module. The Single Component Testbench (SCT) in which the monitoring module is tested in combination with other FRAS components is also presented.

POSITION ESTIMATION: MATHEMATICAL BACKGROUND

The position estimation process is based on the statistical assumption that the observations $L \in \mathbb{R}^{n_L}$ and the true parameters $x \in \mathbb{R}^{n_x}$ are related via functional mathematical observation models. More specifically, the observations are assumed to be error-affected realizations, normally distributed with known covariances Σ around the predicted model response of the true parameter values: $L \sim \mathcal{N}(F(x), \Sigma)$. The optimal parameter estimate is then characterized by the maximum-likelihood principle as the solution of the least square optimization problem

$$P_L : \min_{x \in \mathbb{R}^{n_x}} \big\| \underbrace{\Sigma^{-0.5} r(x, L)}_{=:g(x,L)} \big\|^2 , \tag{1}$$

that seeks to minimize the weighted sum of the squared residuals $r(x, L) = F(x) - L$.

Iterative Solution: Gauß–Newton (GN) Method

As the functional model F in general is nonlinear, problem P_L as defined in (1) is a nonlinear least square problem and requires an iterative solution process. For this purpose, the GN method is used which generates a sequence of parameter iterates $x_{k+1} = x_k + \Delta x$ by solving linearized versions of problem P_L:

$$\min_{\Delta x \in \mathbb{R}^{n_x}} \|W_k + A_k \Delta x\|^2 , \tag{2}$$

where $W_k := g(x_k, L)$ and $A_k := \frac{\partial g}{\partial x}(x_k, L)$. Problem 2 is a linear least-square problem and its solution Δx^* is characterized by the normal equation $N_k \Delta x^* - A_k^T W_k = 0$ which can be computed using a Cholesky decomposition of the normal matrix $N_k = A_k^T A_k$.

$$\begin{tikzcd}
L^{i-1} \arrow[d] & L^{i} \arrow[d] & L^{i+1} \arrow[d] \\
P_{L^{i-1}} \arrow[d] \arrow[r, dotted] & P_{L^{i}} \arrow[d] \arrow[r, dotted] & P_{L^{i+1}} \arrow[d] \\
x^*(L^{i-1}) \arrow[ur, "ini"] & x^*(L^{i}) \arrow[ur, "ini"] & x^*(L^{i+1})
\end{tikzcd}$$

Figure 1: The sequence of measurements $(L^i)_{i\in\mathbb{N}}$ gives rise to a sequence of estimation problems. The solution of problem P_{L^i} is used to initialize the iterative solution process for problem $P_{L^{i+1}}$.

Monitoring Aspect

A particular aspect in the monitoring context is that the problems P_L do not arise as isolated instances of optimization problems but rather as members of a parametrized sequence of problems. Under certain mild regularity conditions on the mathematical model F, the solution $x^*(L)$ of P_L can be interpreted as a continuous function of the measurements L [3]. This is exploited in the warm-start initialization strategy as illustrated in Fig. 1.

For the perturbed problem $P_{L+\Delta L}$, the linearization

$$g(x_k + \Delta x, L + \Delta L) \approx W_k - \Sigma^{-0.5}\Delta L + A_k \Delta x$$

allows to compute the corresponding correction Δx^* as a function of ΔL:

$$\Delta x^*(\Delta L) = N_k^{-1}(-A_k^T W_k + A_k^T \Sigma^{-0.5}\Delta L). \qquad (3)$$

If x_k is a good approximation of $x^*(L)$, the term $N_k^{-1}A_k^T W_k$ becomes negligible and (3) provides linearized parameter estimate updates for the perturbed problem without significant computational effort: no evaluation of the functional model is necessary, only two matrix-vector multiplications and one solution operation that can reuse the already available Cholesky decomposition of N_k are needed. This highlights the further potential of the GN method for providing fast parameter estimates even for bigger systems in the future.

CONTINUOUS POSITION ESTIMATION LIBRARY

The monitoring library is designed on top of LGC, a modular, object-oriented position estimation software developed at CERN. We give a brief description of the standard LGC workflow which naturally motivates the development of a monitoring specific extension of LGC. We follow up with an overview of the features of the novel LGC monitoring library and its intended workflow.

Static Estimation Workflow of LGC

LGC is the main position estimation software [4, 5] used by CERN surveyors and offers a great flexibility in terms of defining measurement configurations and observation types and instruments. The standard workflow of LGC is file-based, with an input file containing the measurement configuration and the associated observation values being passed as argument to a command line call of LGC. After the translation into the abstract mathematical form, the problem is solved iteratively as described in the previous section. When the termination criterion is met, a file containing all results (estimated parameters, statistical data etc.) is generated, shown in Fig. 2.

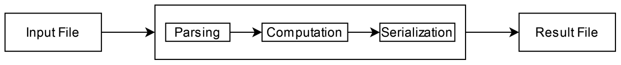

Figure 2: Static workflow of LGC.

It is apparent that in the monitoring context the measurement configuration is static and only the observation values evolve in subsequent problems. Therefore replicating the static workflow in the dynamic monitoring environment will produce unnecessary overhead with repetitive input file parsing. Furthermore the computational environment for the FRAS application prohibits the direct writing of result files and demands a more dynamic way of result extraction.

Dynamic Workflow of the LGC Monitoring Library

To provide a more streamlined workflow reflecting the dynamic nature of a continuous stream of estimation problems, a novel monitoring library was developed as an extension of LGC. It is designed as a class in C++ that allows easy instantiation of a "monitor" object which then in turn offers a lightweight file-less interface for updating measurements, triggering computations and extracting results.

We now will outline the main features of the library including the instantiation and the workflow during the continuous monitoring process, as shown in Fig. 3.

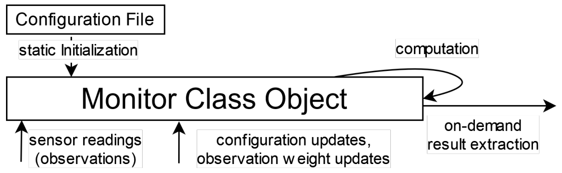
Configuration File
static Initialization
computation
Monitor Class Object
on-demand result extraction
sensor readings (observations)
configuration updates, observation weight updates

Figure 3: Dynamic workflow of the monitoring library.

Initialization

The monitor object is initialised once by loading a configuration file that contains all the static configuration data, i.e. sensor names, instruments used, observations with identifiers. Internally, this defines the function $g(x, L)$ that characterizes problem P_L.

Manipulating Observations and Parameters

Observations The essential requirement for continuous monitoring is the ability to update observation values provided by the sensors on-the-fly using unique observation identifiers for each sensor. Beside this, the class also offers more granular control with the observation weights capable of being changed and, in case of sensor failure, the possibility to deactivate specific observations such that they are not anymore influencing the least-square adjustment process.

Parameters In some cases, e.g. if a movement of a component gets blocked, it can become necessary to fix a previously estimated parameter to a user-specified value. For this purpose a “freeze” method can be called with the parameter name and a value which fixes the corresponding variable accordingly. An “unfreeze” method allows to include a previously frozen parameter again as decision variable if desired.

Computation and Result Extraction

Computation The least-square computation is triggered by the adjust() method. All observation updates, including frozen parameters, adapted weights etc. that are supplied up until the time of the call of the method are considered in the subsequent computation. By default, the solution of the previous estimation problem is taken as the initial value for the GN iterations, as illustrated in Fig. 1. The adjust() method returns a status boolean indicating the success or failure of a computation.

Result Extraction After computation, the estimated parameters can be extracted by calling the getEstimate() method with the parameter identifier as argument. Additional statistical data such as the estimated precision for each parameter and the global sigma a posteriori can also be extracted. It is also possible to extract the actual residual for each observation. To prevent an unintentional result extraction without a previous call of the adjust() method (e.g. because a measurement update was applied after the preceding computation) all the result extraction methods are protected via a status boolean that monitors the data validity, ensuring the result data remains up-to-date.

Implementation Aspects

To ensure minimal external dependencies and a clean separation of interface and implementation, the library is designed using the Pointer to Implementation (PImpl) method [6]. The only dependency needed for communicating measurement data and results is the linear algebra library Eigen [7], which is header-only. The library is distributed in the form of a shared C++ library that includes a header file exposing all the available methods as well as a detailed documentation of the interface. The cross-platform design of LGC (Windows and Linux) carries over to the monitoring library. It can be conveniently incorporated into other C++ applications through linkage via CMake.

APPLICATION IN THE SINGLE COMPONENT TESTBENCH

The SCT is a mock-up installation of a single LHC inner triplet quadrupole fitted with all the hardware and software solutions foreseen for the FRAS. It serves as important validation milestone before the more comprehensive Inner Triplet (IT) magnet string test [8] and allows the study of the collective behavior of the complete acquisition, control and command chain.

The SCT configuration relies on Wire Position Sensors (WPS) and Hydrostatic Leveling Sensors (HLS) that measure the quadrupole position [9], see Fig. 4. In the SCT mock-up production environment the library has been actively operated from within the Front End Software Architecture (FESA) real-time control framework [10] with a frequency of one estimation cycle per second for several weeks. The position estimation results are extracted and fed back through the FESA system and form the basis for the motor commands.

Tests to assess the performance with artificially generated perturbed measurements on a 16 GB, 11th Gen Intel Core i5 machine show that library is able to treat the SCT configuration at around 100 estimation cycles per second, demonstrating its efficiency in a controlled test environment.

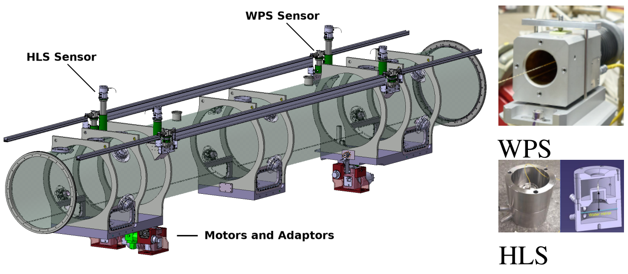

Figure 4: The SCT setup with the Twice Aperture Prototype (TAP) quadrupole equipped with hydrostatic water level sensors HLS and wire position sensors WPS for position monitoring.

CONCLUSION

A position estimation library for monitoring applications based on the least square estimation software LGC has been created. The reusable module provides a standardized way to configure and operate continuous monitoring applications facilitating its integration into other surveying projects. A lightweight interface allows on-the-fly treatment of measurement updates and result extraction that is adapted to the continuous nature of the monitoring processes. The library is now a core-component for the position estimation in the FRAS, has been integrated in its hardware and software stack foreseen for the HL-LHC upgrade, and is currently operating on a single component testbench with an update frequency of 1Hz.

ACKNOWLEDGEMENTS

The research leading to these results is supported by the HL-LHC project. Gratitude goes to the colleagues V. Rude, B. Schoefield and J. Borrego from the contributing HL-LHC project who provided valuable insights to this study.

REFERENCES

[1] P. Biedrawa *et al.*, "Full remote alignment system for the High-Luminosity Large Hadron Collider HL-LHC", presented at the 16th International Workshop on Accelerator Alignment (IWAA'22), Ferney-Voltaire, France, Oct. 2022. `https://cds.cern.ch/record/2849056/files/CERN-BE-2023-007.pdf`

[2] M. Sosin *et al.*, "Design and Study of a 6 Degree-Of-Freedom Universal Adjustment Platform for HL-LHC Components", in *Proc. IPAC'19*, Melbourne, Australia, May 2019, pp. 3720–3722. `doi:10.18429/JACoW-IPAC2019-THPGW058`

[3] J. F. Bonnans and A. Shapiro, "Optimization Problems with Perturbations: A Guided Tour", SIAM Review, vol. 40, no. 2, pp. 228–264, 1998. `doi:10.1137/S0036144596302644`.

[4] M. Jones, "An Object Oriented Approach to Processing Accelerator Alignment Measurements", presented at the 11th International Workshop on Accelerator Alignment (IWAA'10), DESY, Hamburg, Germany, Sept. 2010. `https://iwaa2010.desy.de/e107506/e107507/e113379/e119273/IWAA2010-MJ_LGC.pdf`

[5] M. Barbier, Q. Dorleat, and M. Jones, "LGC: A new revised version", presented at the 11th International Workshop on Accelerator Alignment (IWAA'16), Grenoble, France, Oct. 2016.

[6] H. Sutter and A. Alexandrescu. "The Pimpl Idiom". In *C++ Coding Standards: 101 Rules, Guidelines, and Best Practices*, pages 101–103. Addison-Wesley Professional, 2005.

[7] G. Guennebaud *et al.*, "Eigen v3". 2010. `http://eigen.tuxfamily.org`

[8] M. Bajko *et al.*, "HL-LHC IT STRING: Status and Perspectives", in IEEE Transactions on Applied Superconductivity, vol. 34, no. 5, pp. 1–6, 2024. `doi:10.1109/TASC.2024.3354217`.

[9] V. Rude *et al.*, "3D calculation for the alignment of LHC low-beta quadrupoles", Technical Report CERN-BE-2023-003, CERN, Geneva, 2022. `https://cds.cern.ch/record/2849052`.

[10] M. Arruat *et al.*, "Front-end software architecture", in *Proc. ICALEPCS'07* , Knoxville, TN, USA, Oct. 2007, pp. 310–312.