

ONE ADJUSTMENT ENGINE, MANY APPLICATIONS: LGC AS A VERSATILE SURVEY TOOLBOX AT CERN

G. Kautzmann*, A. V. Burtonwood, M. Carotta, J. Gutekunst,
F. Klumb, P. Sainvitu, J.-F. Segers, R. Singh, CERN, Geneva, Switzerland

Abstract

Since the mid-1980s, LGC (Logiciel Général de Compensation) has been CERN's primary software for the least-squares adjustment of survey networks and alignment processes. Historically operated as a self-contained executable, it has evolved into an adjustment engine embedded within a diverse range of survey workflows. The evolution was driven by both operational needs and a deliberate strategy to open LGC beyond its original context. This strategy involved exposing the computation core as a reusable library, developing machine-readable outputs, documenting stable interfaces, and encouraging external applications. Documentation improvements and active community outreach have been integral to this effort. On the technical side, two developments stand out: a JSON serialisation format and an application programming interface (API), which together expose LGC as a callable library and provide structured access to its results.

This paper presents several domains in which LGC is now used programmatically: field acquisition, graphical survey environments, live monitoring systems, post-processing scripts, and simulation. All were built outside LGC's C++ codebase. Where a given integration required additions to LGC itself, those were made generally available to all users. The JSON serialisation format has further given rise to standalone tools such as dynamic result reports and geographic visualisation interfaces. Some were contributed by people without any surveying or software development background, illustrating that neither use nor contribution requires deep knowledge of LGC's internals. With LGC publicly available under an open-source licence since early 2026, these examples document CERN's current practice and invite the broader community — surveyors, scientists, and developers alike — to use, adapt, and extend the LGC toolbox.

INTRODUCTION

Surveyors at CERN rely on LGC (Logiciel Général de Compensation) to compute the 3D positions of points and objects through least-squares adjustment of diverse geospatial observations [1]. Since its initial development in the mid-1980s, LGC has evolved from a Fortran 77 program into a modern C++ application supporting more than twenty observation models. These models range from traditional instrumentation — total stations, laser trackers, and levelling systems — to more specialised tooling, for instance horizontal-offset stretched-wire measurements and inclinometers. They extend further still to continuously monitoring sensors such as hydrostatic levelling systems (HLS) and

wire-positioning sensors (WPS). Beyond this range of observation models, LGC offers considerable flexibility in how a network is set up — constraints linking points and parameters, an extensive set of tunable computation options and stochastic models, and hierarchies of reference systems that structure a network as nested local frames rather than a single flat coordinate system [2]. This variety and granularity let LGC serve setups from simple, single-station campaigns to highly complex, multi-instrument networks, and meet needs across accelerator alignment and beyond. The history, key functionalities, and open-source transition of LGC were presented at IWAA 2024 [3] and documented in the user manual [4]. A companion paper at this conference tells the broader, organisation-wide story of CERN Survey's software [5].

As continuous monitoring systems and automated processing chains have multiplied across CERN's accelerator complex, the demand has grown for LGC to be invoked repeatedly and without supervision. Increasingly, that invocation comes not from a person running a single command, but from other software. Meeting this demand meant rethinking how LGC could be accessed: not only as a program a surveyor runs, but as a component other programs can call.

This paper focuses on the growing use of LGC not as a single-shot executable but as an adjustment *engine* embedded within broader workflows. Two mechanisms, already introduced in prior work [3, 6], underpin that shift: a machine-readable output format and a documented, callable programming interface. Neither requires access to LGC's internals, and the applications presented here — old and new alike, several built by people from outside the maintainer team — are a sample of what they enable, brought together here as working examples for others to build on. The paper ends with a brief look at LGC's transition to open source, and at what these same mechanisms could offer an accelerator complex growing towards the scale of the Future Circular Collider (FCC) [7].

LGC AS A TOOLBOX

Classical Operating Mode

In its classical mode, LGC operates as a standalone executable, available for both Windows and Linux. A formatted text input file — containing computation options, frame hierarchy definitions, a priori parameter values, constraints, and observations with their parametrised stochastic model — is passed via the command line, and the software writes results to text output files. This workflow remains the primary mode for one-off survey campaigns, largely unchanged since the introduction of LGC2 around 2016. Several purpose-built

* guillaume.kautzmann@cern.ch

field-acquisition and monitoring applications rely on this same classical mode for automated use, preparing LGC's input files and invoking it as a subprocess or standalone executable without a person triggering each run.

SurveyPad [8], the unified graphical interface for CERN's suite of survey data-processing applications, follows the same classical pattern. It hosts a dedicated plugin for LGC — pointing to an LGC executable installed separately on the user's machine — and invokes it via the same command-line call. SurveyPad itself is also publicly available, though not yet released as open source. For LGC specifically, SurveyPad offers quality-of-life features such as syntax highlighting of the input file and navigation between input and output files. The input file itself is still prepared by the user, whether by hand in SurveyPad's built-in text editor or with the help of pre-processing tools, including data pulled directly from the CERN Survey Database via GEODE [9]. SurveyPad also serves as a distribution platform: anyone can develop a script against its scripting framework and publish it as an interactive addition to SurveyPad itself. Its built-in scripting support treats Python as the preferred language for such pre- and post-processing workflows [6]. Delivered as a bare script, such a tool would have to be installed, configured, and launched from a command line. Packaged as a SurveyPad addition instead, it arrives with its dependencies resolved and its parameters exposed through the interface, within reach of a much wider set of users — including those with no software background.

Enabling Mechanisms

Two technical developments have progressively transformed LGC into an embeddable engine: JavaScript Object Notation (JSON) serialisation and the application programming interface (API). Individually, each is a generic building block rather than a solution to any single problem. Because neither requires access to LGC's internals, building on them is open to anyone willing to write against a documented interface, inside or outside the small team of LGC maintainers.

JSON serialisation. Since version 2.6.0 (June 2023), LGC can export its full data structure — input parameters, adjustment results, and derived statistics — in JSON format. This machine-readable output greatly simplifies downstream processing, and has directly enabled a set of standalone tools built entirely outside LGC's C++ codebase. Derived quantities that would otherwise require access to that codebase are pre-computed and included in the export. Full documentation of every JSON field is available in LGC's online user manual [4].

Application programming interface. LGC has also been built as a shared library: a Dynamic Link Library (DLL) on Windows, a Shared Object (SO) on Linux. Both expose an API that other applications can call directly, or through its Python wrappers pyLGC. The API exposes methods to load an LGC input file, modify observations or parameter states programmatically, trigger the least-squares computation, and retrieve results and statistics, all without spawning a separate process [6, 10]. The API has been used intern-

ally since late 2023, initially to meet the requirements of the Full Remote Alignment System (FRAS) for the High Luminosity LHC (HL-LHC) upgrade, and subsequently for a small number of internal proof-of-concept applications. A stable, production release followed with LGC 2.11.0 in June 2026 [4]. The current release exposes only a subset of the planned functionality, and continued development is anticipated to expand it. Fully file-less operation, with all configuration provided through direct API calls rather than an input file, remains a longer-term direction.

Computation Speed Improvements

A prerequisite for embedding LGC in continuous monitoring systems is that computation must complete well within the measurement cycle. Significant optimisation work was carried out on LGC's computation kernel in preparation for FRAS [10], yielding substantial reductions in adjustment time. For the Single Component Testbench configuration used to validate FRAS, the resulting library sustains approximately 100 estimation cycles per second, against a required refresh rate of 1 Hz. This two-order-of-magnitude margin leaves room for larger or more complex networks. These improvements are available to all LGC users from version 2.7.0 (July 2024) onwards.

APPLICATIONS

The domains described below cover a broad cross-section of the survey workflow — from in-field data acquisition to long-term monitoring, post-processing, and prospective simulation. Together they illustrate the three modes of operation built on the building blocks introduced in LGC as a Toolbox, above. The classical executable lies behind field acquisition and cryomodule monitoring, the API behind FRAS, blunder detection, and simulation, and the JSON export behind graphical visualisation. They were chosen to span the range of what is possible, not to provide a complete inventory of everything LGC is used for at CERN.

Field Acquisition

TSUNAMI [11] is CERN's in-house in-field measurement and alignment software, used by surveyors alongside commercial alternatives. It is organised into guided modules, which walk a surveyor step by step through a given measurement, and advanced modules, which offer free use of the underlying acquisition and computation tools. A field session can also launch SurveyPad [8] directly. During field campaigns, LGC acts as TSUNAMI's computation engine, serving two ends. One is the positioning work, such as determining the 3D position of an instrument within the accelerator network. The other is the set of checks confirming that a measurement is free of outliers and correctly taken before the surveyor leaves the site. These tasks draw on polar (laser tracker), levelling, stretched-wire offset, and inclinometer (roll) measurements. TSUNAMI prepares input files in a format directly readable by LGC. Once the computation is complete, LGC can produce files that insert point

positions into the CERN Survey Database via its interface GEODE [9]. Its JSON export is now the primary means by which TSUNAMI extracts and presents the results a surveyor needs. TSUNAMI applies those checks using LGC's own residuals and a-posteriori variance factor to confirm each computation is sound, and runs a further check on the raw observations to catch gross errors. In this integration, LGC is invoked as a subprocess automatically by TSUNAMI, without the surveyor manually running an LGC command. The tight coupling between field acquisition and on-the-fly adjustment shortens the feedback loop for surveyors working on site.

Continuous Monitoring Systems

Two systems at CERN rely on LGC for continuous position estimation: FRAS, currently in its qualification phase ahead of installation, and MATHILDE (Monitoring and Alignment Tracking for HIE-ISOLDE), running in its classical mode since 2016.

FRAS. The system is designed to estimate the positions of accelerator components for the HL-LHC upgrade continuously, at up to 1 Hz [10]. Measurements come from several sensor types — WPS, HLS, Frequency Scanning Interferometry (FSI) for absolute distance measurement, and inclinometers — fed to LGC at regular intervals without human intervention [10, 12]. The position of the cold mass of each HL-LHC inner-triplet component is monitored inside its vacuum vessel, and referred back to the wider frame of the alignment network linking the components together. Where a deviation is found, an operator can trigger a physical realignment through the Universal Adjustment Platform [13] and motorised jacks installed under the equipment. The speed improvements described in LGC as a Toolbox, above, were developed specifically to meet FRAS's performance and software-stability requirements. To avoid any risk of downtime, FRAS integrates a dedicated branch of the library, kept alongside the mainline release. FRAS is not yet installed on the accelerator: most of the work to date has been qualification and validation on several testbenches, with installation planned before 2030. Once operational, FRAS will represent the first large-scale deployment of the LGC API in a monitoring environment, embedded within CERN's Front-End Software Architecture [14], a real-time control framework. The API exposes finer-grained control over the underlying computation, available both in the mainline production library — the Windows DLL and Linux SO, and their Python wrapper pyLGC [4] — and in FRAS's dedicated Linux branch. Individual observations can be updated on the fly, reweighted to reflect a change in sensor behaviour, or deactivated in case of sensor failure. Individual parameters can likewise be fixed to a known value or released again [10].

MATHILDE. Since 2016, this system has monitored the positions of superconducting cavities and solenoids inside HIE-ISOLDE cryomodules, well before LGC's API existed. Small H-BCAM (HIE-ISOLDE Boston CCD Angle Monitor) cameras, mounted on metrological tables outside the cryostat, observe spherical retroreflective glass targets fixed

to the cavities and solenoids through dedicated viewports. This configuration was chosen to fit the project's specific constraints and the surveying techniques available at the time. The system allows the internal elements to be measured across the cryomodule's different states — at ambient temperature, during cooldown, and under stable cryogenic operation. Because the metrological tables carrying the cameras are themselves referenced to pillars outside the cryostat, the positions obtained are tied back into the wider survey network rather than to the cryomodule alone. A representative computation processes 227 such observations to solve for 157 free transformation parameters, taking 20 s to 30 s [15]. During cooldown, measurements are taken every 15 minutes for the first two weeks, then every 30 minutes thereafter. Position estimates are computed by MATHILDE's own software component, called MATHIS, which manages acquisition, invokes LGC as a standalone executable, and links results to the relevant CERN databases [16]. It can run unattended on a CERN Front-End Computer (FEC), a specialised embedded Linux-based computer that interfaces directly with physical accelerator equipment. Equally, it can run as a stand-alone application on an ordinary workstation for expert use. Like TSUNAMI, MATHIS predates LGC's API by several years, which is why both rely on LGC's classical operating mode rather than the API.

Together, FRAS's validated testbench performance and MATHILDE's decade of field operation show that LGC can sustain continuous, automated adjustment loops rather than being triggered manually, campaign by campaign. Either interface serves: the API for FRAS, the classical executable for MATHILDE.

Post-Processing — Automated Blunder Detection

LGCBlunderDetection [17] is a dedicated post-processing tool that automates the identification of outlier observations in survey datasets. In an ordinary least-squares adjustment, every residual contributes to the cost function through its square, so a single large blunder can dominate the solution and distort every other estimated parameter. Version 0.1.2 of the tool is built entirely in Python on top of LGC's Python API (pyLGC) [4] for LGC 2.11.0, and is packaged as a standalone script that can also be launched from within SurveyPad.

Rather than re-implementing any observation model, the tool queries the API for the same linearised least-squares system LGC itself solves internally. This includes the design matrix, the weight matrix, the constraint matrix, and the misclosure vector, together with a map from each observation back to the line of the input file it came from. The underlying geodetic model — how each instrument and observation type is formulated — therefore never needs to be recoded outside LGC. Similar tools were attempted before the API existed, but none proved sustainable. Recoding the model was only part of the difficulty: the quantities listed above were never exposed outside LGC, and a failed computation produced no output at all for a tool to work from. Using these matrices, the tool re-solves the same linearised system under

alternative loss functions instead of ordinary least squares: least-absolute-value, Huber [18], and iterative reweighting. The Huber loss uses a configurable tuning parameter, while the iterative-reweighting scheme repeatedly down-weights and re-solves any observation whose normalised residual exceeds a configurable threshold. Each is formulated as a convex optimisation problem and solved with the open-source Clarabel solver [19]. In parallel, LGC’s own native solver is queried on the same input, so its solution can be compared directly against the robust one. In practice, a surveyor typically works with the tool iteratively (Fig. 1): run the chosen solver once, then inspect the flagged residuals in the interactive report. Any observation that looks like a blunder can be followed back to its line in the input file, where it can be deactivated, down-weighted through its stochastic model, or corrected outright — a mistyped point name or target height, for instance. The computation is then re-run from the edited file, either by the tool itself or by invoking LGC directly, repeating the cycle until no further outlier stands out.

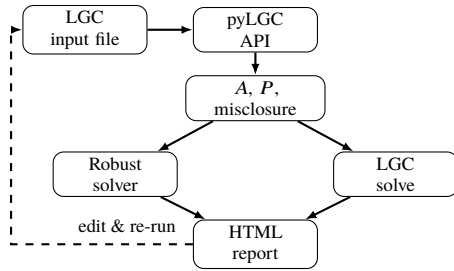


Figure 1: The LGCBlunderDetection pipeline: LGC’s Python API exposes the linearised least-squares system, solved both robustly and by LGC itself. The report leads back to the input file, where the surveyor edits the offending observation before the computation is re-run.

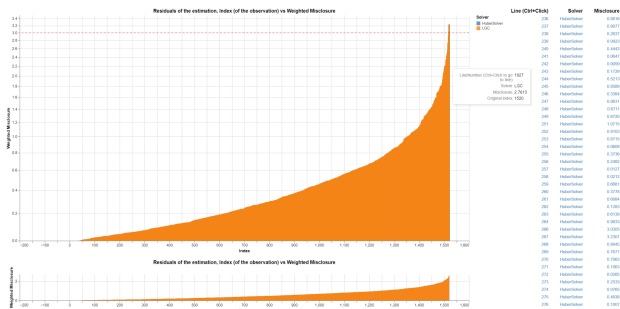


Figure 2: The interactive HTML report generated by LGCBlunderDetection, comparing LGC’s own solution against the robust solver’s for each observation.

The output is an interactive HTML report, plotting each observation’s normalised residual under both the robust and the native LGC solution side by side (Fig. 2). The report supports brushing, zooming, and sorting by either solution. Clicking either a bar in the chart or its row in the linked table jumps straight to the corresponding line of the input file. That navigation is a built-in SurveyPad behaviour rather than anything specific to this tool: any HTML page, or even

an external website, can embed such a link for SurveyPad to intercept.

Simulation and Safety Verification

LGC has long supported simulation for prospective network design, enabling sensitivity analyses before a measurement campaign is carried out. These capabilities are now exposed through the API, allowing external applications to drive simulation scenarios programmatically rather than through the classical input-file workflow.

A recent application is a kinematic simulation framework developed for the virtual commissioning of FRAS [20]. Whereas LGC normally solves for unknown positions given observations, the framework repurposes it as a forward simulation engine. The frame parameters of each component — the translations and rotations that locate it within the network — are frozen at a hypothetical, possibly displaced configuration, and LGC is asked what each sensor should then read. This reuses LGC’s residual-computation machinery, and with it every observation model already built into LGC — such as geoid corrections and wire-sag corrections for WPS — without re-implementing any of them, and without any addition to LGC itself.

Built on the LGC API and its Python bindings, the framework writes these simulated readings directly into real FRAS FECs running the control software, in place of live sensor data. This lets the same control-system logic that would run in production — including the Protection Layers, the interlock logic that stops motors when a measured deformation exceeds its threshold — be exercised without moving a real magnet, ahead of installation on the machine (Fig. 3).

Early runs alone caught several issues: a missing unit conversion, sign errors in the Protection Layer formulae, and a misconfiguration that silently disabled a Protection Layer safeguard. Each lay beyond the reach of unit testing, because each lived in the composition of components that were correct in isolation. The framework has since become a standing integration suite, exercised daily during development both to validate new features and to guard against regression, and is required to pass before any change to the control system is deployed. The framework is described in [20] as a first prototype of a FRAS digital shadow, though not yet encompassing the two-way communication a full digital twin would need.

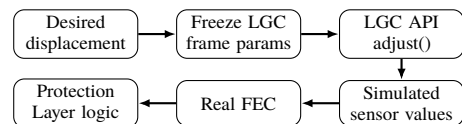


Figure 3: The kinematic simulation pipeline: LGC’s forward mode generates simulated sensor readings for a hypothetical displacement, written into FRAS’s real FECs to exercise its Protection Layer logic under test.

Another project uses the same API to optimise sensor and target placement, applying experimental-design methods to find the configuration that minimises uncertainty for a given instrumentation budget [21]. Both applications treat

LGC’s adjustment engine not merely as a tool to process real measurements, but as a predictive model that can be queried ahead of a campaign or a system’s deployment.

Graphical Visualisation — LGC2GIS

Beyond the API, the JSON serialisation has independently given rise to several loosely coupled post-processing tools, each built purely from the exported JSON. A dynamic HTML result report [3] renders adjustment statistics with interactive sorting and plotting directly from the JSON export. A set of lightweight post-processing scripts reprojects point coordinates and their associated precision ellipsoids into alternative reference frames, without requiring access to LGC’s internal frame-transformation logic. The most developed example among them is LGC2GIS, a visualisation tool that publishes LGC adjustment results to CERN’s geographic information system (GIS) [22].

LGC2GIS consumes the JSON output of an LGC computation and converts it first to the standard GeoJSON format, and then transforms it into the ArcGIS JSON format expected by CERN’s GIS infrastructure, with metadata attached to each feature. Six layers are generated this way — points, laser-tracker observations, confidence ellipses and vertical error bars, planimetric and altimetric displacement vectors (the offset between each point’s provisional and adjusted coordinates). Each is derived entirely from statistics already present in the JSON export, such as each point’s covariance matrix, without requiring any further computation from LGC itself. All objects are first transformed from LGC’s local CERN Coordinate System into CHTRF95, Switzerland’s national geodetic reference frame, as required by CERN’s ArcGIS JSON format. The transformation itself is performed by SurveyLib [23], the same open-source library underlying LGC, exposed here through its own Python bindings. This mirrors the DLL-plus-bindings architecture already described for the LGC API. The result is then uploaded through a secure, token-authenticated API call to the central GIS database (Fig. 4).

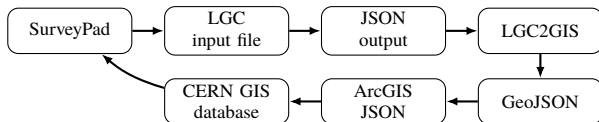


Figure 4: The LGC2GIS pipeline, from a SurveyPad-driven LGC computation to CERN’s geographic information system and back to SurveyPad’s GIS portal.

LGC2GIS then opens the GIS portal within SurveyPad; the same portal page can equally be opened separately in a standard web browser. There, surveyors and non-surveyors alike can explore a geographic view of the adjustment (Fig. 5). Every object on the map — points, observations, confidence ellipses, vertical error bars and displacement vectors alike — is clickable, opening a panel with its associated metadata. For an observation, for instance, that panel shows its measured value, sigma, and residual, colour-coded by normalised residual: grey below two, orange between two and three, and red beyond three. A direct link in the

panel opens the corresponding line of the LGC input file, through the same SurveyPad link mechanism used by the blunder-detection report. From the tool’s interface, users can also scale the ellipses, error bars, and displacement vectors for clearer visualisation. As long as the tool’s window stays open within SurveyPad, every new computation is uploaded automatically, with layer names incremented so that successive survey configurations can be compared side by side.

Three-dimensional visualisation is not yet supported; extending LGC2GIS to building-information-modelling (BIM) formats such as Industry Foundation Classes (IFC) could be a natural next step.

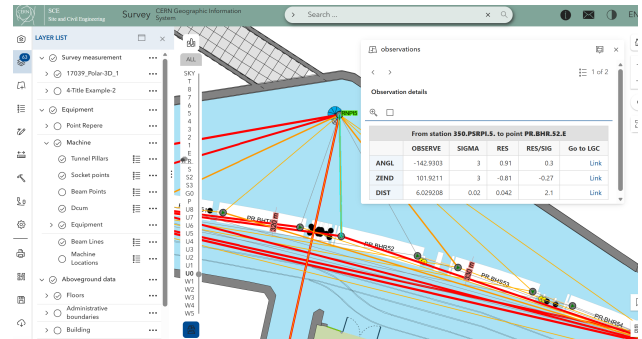


Figure 5: The CERN GIS portal opened from SurveyPad: observation lines coloured by normalised residual, with an observation’s details panel showing a direct link back to the corresponding line of the LGC input file.

Across the tools built on the JSON export and the API — automated blunder detection, LGC2GIS, and the simulation and optimal-design work — a common pattern emerges. Each was built entirely outside LGC’s C++ codebase. Each also lowers the barrier for the next person to build something new on top of LGC. Indeed, several of these tools were built by people with no background in survey software, each supported as needed by the maintainers’ experience in mathematics, surveying, and development. LGCBlunderDetection and the optimal-design work [21] are the product of mathematics students, and the kinematic simulation framework grew out of a master’s thesis in cybernetics and robotics. LGC2GIS, built by a surveying engineer, needed only modest support from the core team, and only to connect to the GIS portal’s REST API. They remain, deliberately, only a sample: those same building blocks are open to whichever application a surveyor, a student, or a developer elsewhere chooses to build next.

CONCLUSION AND OUTLOOK

LGC has evolved from a standalone executable into an adjustment engine integrated at many points across CERN’s survey workflow, from field acquisition to continuous monitoring. JSON serialisation and the API introduced a small set of generic building blocks rather than a fixed catalogue of features, and the applications described here draw on only part of what those blocks already support. Speed improvements to the computation kernel have been a key enabler

for monitoring applications that require continuous, automated adjustment cycles. Continued work in this direction is anticipated to be necessary as CERN’s accelerator complex looks towards FCC scale — a tunnel circumference of 90.7 km against the LHC’s 26.7 km, whose geodetic studies are already under way [7]. Likely areas of further work include the use of graphics processing units to accelerate the most demanding steps of the solution, refined partial-derivative formulations, and an updated geoid model.

Meeting these demands could draw on exactly the kind of external contribution this paper has explored, which is why LGC’s own transition to open source matters beyond its present-day applications. Following the strategy presented at IWAA 2024 [3], LGC is now publicly available as open-source software, hosted under the geodetic-metrology-tools organisation [24] and documented at lgc2.docs.cern.ch [4]. The GPL-3.0-or-later licence covers the source code, with documentation released separately under CC-BY-4.0. This has been accompanied by active outreach — presentations, training sessions, and direct support for early adopters. LGC has already found its first external adopter: the European Synchrotron Radiation Facility (ESRF) has integrated it into its own accelerator-alignment workflows, providing operational validation beyond CERN [6]. Community contributors could similarly adopt LGC as a pre- or post-processing component, adapt the monitoring or simulation approaches presented here to their own sensor systems, or contribute new application layers on top of the existing API. IWAA has long served as a venue for sharing alignment techniques across institutions. The authors hope that an open LGC toolbox, and in time a broader ecosystem of tools built around it [5], can become a shared resource for that same community.

REFERENCES

- [1] J. K. Illiffe, “Three-dimensional adjustments in a local reference system”, 1987. [doi:10.5170/CERN-1987-001.307](https://doi.org/10.5170/CERN-1987-001.307)
- [2] M. Barbier, Q. Dorleat, and M. Jones, “LGC: a new revised version”, presented at IWAA 2016, ESRF, France, Oct. 2016.
- [3] G. Kautzmann, J. Gutekunst, F. Klumb, and P. Elson, “LGC open-source: a strategy to share adjustment software and algorithmic development”, presented at IWAA 2024, SLAC, USA, Oct. 2024.
- [4] LGC2 documentation, <https://lgc2.docs.cern.ch/>
- [5] F. Klumb, J. Gutekunst, S. Huang, G. Kautzmann, and R. Singh, “Towards a sustainable and interoperable software ecosystem for survey and alignment activities at CERN”, presented at IWAA 2026, KEK, Japan, Oct. 2026.
- [6] G. Kautzmann, J. Gutekunst, F. Klumb, R. Singh, and P. Elson, “LGC: CERN’s geodetic network adjustment software goes open source”, presented at the FIG Congress 2026, Cape Town, South Africa, May 2026.
- [7] B. Weyer, B. Yang, D. Mergelkuhl, *et al.*, “Overview of the Future Circular Collider at CERN from the survey and alignment perspective”, presented at IWAA 2026, KEK, Japan, Oct. 2026.
- [8] P. Lewandowski, R. Ducceschi, and F. Klumb, “SurveyPad, a common interface to Geode & data processing tools”, presented at IWAA 2022, CERN, Switzerland, Oct. 2022.
- [9] A.-V. Naegely, I. Iliev, and A. Bensahla Talet, “GEODE: A new database interface with apex”, presented at IWAA 2016, ESRF, France, Oct. 2016.
- [10] J. Gutekunst, F. Klumb, G. Kautzmann, H. Mainaud Durand, and M. Sosin, “Continuous position estimation for the full remote alignment system of the High Luminosity LHC upgrade”, *JACoW*, vol. IPAC2024, THPG36, 2024. [doi:10.18429/JACoW-IPAC2024-THPG36](https://doi.org/10.18429/JACoW-IPAC2024-THPG36)
- [11] P. Sainvitu, P. Dewitte, J.-C. Gayde, D. Mergelkuhl, and D. Missiaen, “Tsunami, a unified in-field measurement and alignment software for experiments and accelerators at CERN Large Scale Metrology Section”, presented at IWAA 2016, ESRF, France, Oct. 2016.
- [12] V. Rude *et al.*, “3D calculation for the alignment of LHC low-beta quadrupoles”, presented at IWAA 2022, CERN, Switzerland, Oct. 2022.
- [13] M. N. Noir *et al.*, “Motorized Universal Adjustment Platform for Micrometric Adjustment of Accelerator Components”, in *Proc. 12th Int. Conf. Mech. Eng. Design Synchrotron Radiat. Equip. Instrum. (MEDSI’23)*, Beijing, China, Nov. 2023, pp. 316–318. [doi:10.18429/JACoW-MEDSI2023-THPPP026](https://doi.org/10.18429/JACoW-MEDSI2023-THPPP026)
- [14] A. Guerrero, J. J. Gras, J. L. Nougaret, M. Ludwig, M. Arnaud, and S. Jackson, “CERN front-end software architecture for accelerator controls”, in *9th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALPCS 2003)*, pp. 342–344, Oct. 2003.
- [15] G. Kautzmann, J.-C. Gayde, and F. Klumb, “HIE-ISOLDE: Commissioning and first results of the Mathilde system monitoring the positions of cavities and solenoids inside cryomodules”, presented at IWAA 2016, ESRF, France, Oct. 2016.
- [16] F. Klumb, J.-C. Gayde, and G. Kautzmann, “MATHIS Software for Controlling BCAM-Based Monitoring and Alignment Systems”, presented at IWAA 2016, ESRF, France, Oct. 2016.
- [17] M. Carotta, “New implementation of blunder detection algorithms as scripts for CERN’s in house least-square adjustment software LGC”, CERN, Geneva, Rep. EDMS 3337447v.2, 2025. <https://edms.cern.ch/document/3337447/2>
- [18] P. J. Huber, “Robust Estimation of a Location Parameter”, *The Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73–101, 1964. [doi:10.1214/aoms/1177703732](https://doi.org/10.1214/aoms/1177703732)
- [19] P. J. Goulart and Y. Chen, Clarabel: an interior-point solver for conic programs with quadratic objectives, 2024, arXiv: [2405.12762](https://arxiv.org/abs/2405.12762) [math.OA],
- [20] A. V. Burtonwood, “Kinematic simulation for virtual commissioning of particle accelerator alignment systems”, Master’s thesis, Norwegian University of Science and Technology (NTNU), Nov. 2025. <https://hdl.handle.net/11250/5481328>
- [21] B. Yang, J. Gutekunst, and M. Del Grande, “Optimal sensor and target placement via experimental design”, presented at IWAA 2026, KEK, Japan, Oct. 2026.

- [22] J.-F. Segers, G. Kautzmann, and N. Guilhaudin, “BE-GM and GIS collaboration: development of a new visualization tool for LGC compensation”, CERN, Geneva, Rep. EDMS 3414547 v.1, 2026. <https://edms.cern.ch/document/3414547/1>
- [23] SurveyLib – GitHub repository, <https://github.com/geodetic-metrology-tools/SurveyLib>
- [24] LGC2 – GitHub repository, <https://github.com/geodetic-metrology-tools/LGC2>