Article Identifier
DOI: 10.22032/dbt.58903
URN: urn:nbn:de:gbv:ilm1-2023isc-096:6

60th ILMENAU SCIENTIFIC COLLOQUIUM
Technische Universität Ilmenau, 4 – 8 September 2023
URN: urn:nbn:de:gbv:ilm1-2023isc:1

USING GEOMETRIC ALGEBRA TO CREATE DIFFERENTIABLE MODELS FOR OPTIMIZING CAMERA-BASED OPTICAL METROLOGY SYSTEMS

Simon Hartel, Christian Faber

Hochschule Landshut – University of Applied Sciences

ABSTRACT

In the design process of *camera-based optical metrology systems* numerous intricate and seemingly distinct optimization tasks emerge. A frequently occurring but crucial task in design or calibration is to optimize the spatial degrees of freedom of system components. Of course, modelling the poses of rigid bodies is long solved using rotation matrices and translation vectors, but when it comes to optimizing, this choice of model gets quite tedious to handle. Useful concepts such as homogeneous coordinates or (dual) quaternions have been introduced to overcome this, which however – lacking a unified framework – can quickly become difficult to maintain.

As an alternative, in this contribution it is shown how the unifying methods of *geometric algebra* can be used as an advantage for *gradient-based optimization* of camera-based optical metrology and imaging systems – and how this can be done in a generalized way for seemingly different objectives with respect to system design and calibration.

1. INTRODUCTION AND STATE OF THE ART

Camera-based optical metrology systems such as stereo vision, fringe projection, deflectometry or light sectioning (Fig 1) are well-established and widely used in industrial applications for quality control and measurement of free-form surfaces.

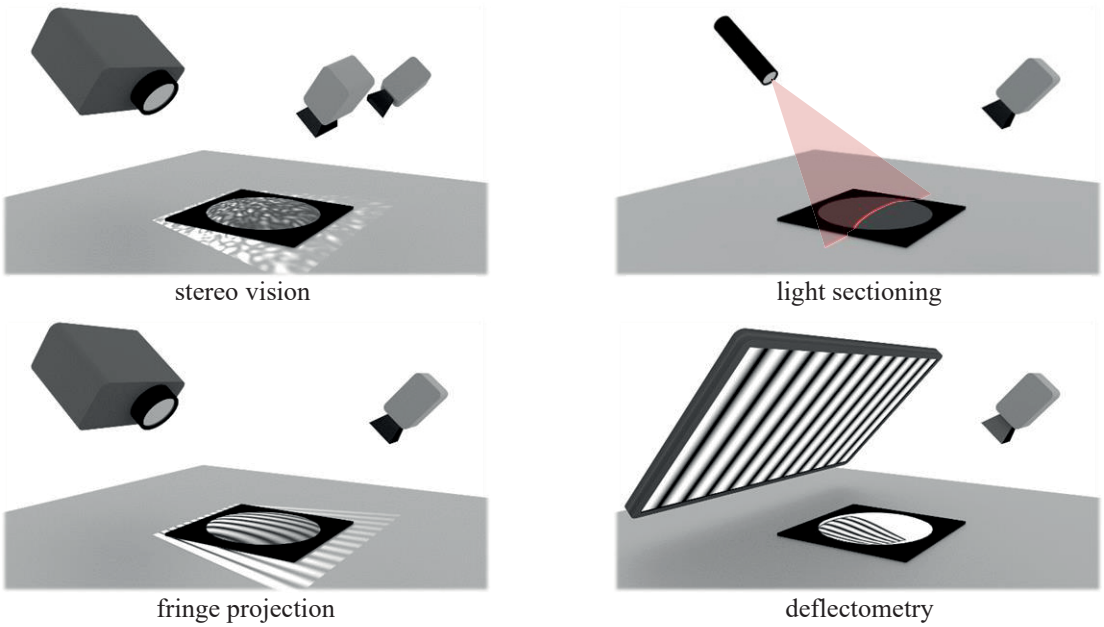

Fig 1: Examples for camera-based optical metrology systems.

However, when designing these systems, many complex and, at first glance, very different optimization tasks arise – for example, finding the best location of a customer specific specimen under test relative to a given sensor or, to a greater extent, finding an optimized geometry of the sensor itself for a specific measurement scenario. In addition, once the sensor is built, calibration parameters must be determined as accurately as possible – which can also be regarded as and obtained by the result of an appropriate optimization process.

A common yet important task when trying to reach these objectives that are often treated independently at the moment is to optimize the spatial degrees of freedom of a rigid body, e.g. the pose of a camera, a light source, a robot end-effector, or a specimen under test. Of course, modelling the pose of a rigid body is a long-solved problem and classically achieved by a rotation matrix and a translation vector, but when it comes to optimizing with respect to a given merit function, this choice of model gets quite tedious to handle, since rotation and translation are modelled differently and can suffer from gimbal lock and other needs for exception handling or optimization constraints that are hard to implement. To overcome this, useful concepts such as homogeneous coordinates [1] or dual quaternions [2,3,4,5] have been introduced in the past, which however due to their very distinct nature can quickly become extensive and difficult to maintain, especially in combination with optimizing complex models or intricate digital twins.

As an alternative, in this contribution it is shown how the unifying methods of *geometric algebra* can provide a consistent framework containing all the methods mentioned above, and can therefore be used as an advantage for *gradient-based optimization* of camera-based optical metrology and imaging systems. It is proposed how this can be done in a generalized way for seemingly different objectives with respect to system design, specimen placement, and calibration.

2. GENERALIZED FRAMEWORK FOR MODEL-BASED OPTIMIZATION TASKS

Rather than having to create a new optimization model for each of the objectives mentioned above, the seemingly different optimization tasks turn out to be closely related and can be generalized by the gradient-based optimization framework proposed in [6], given a *differentiable* model of the system behavior.

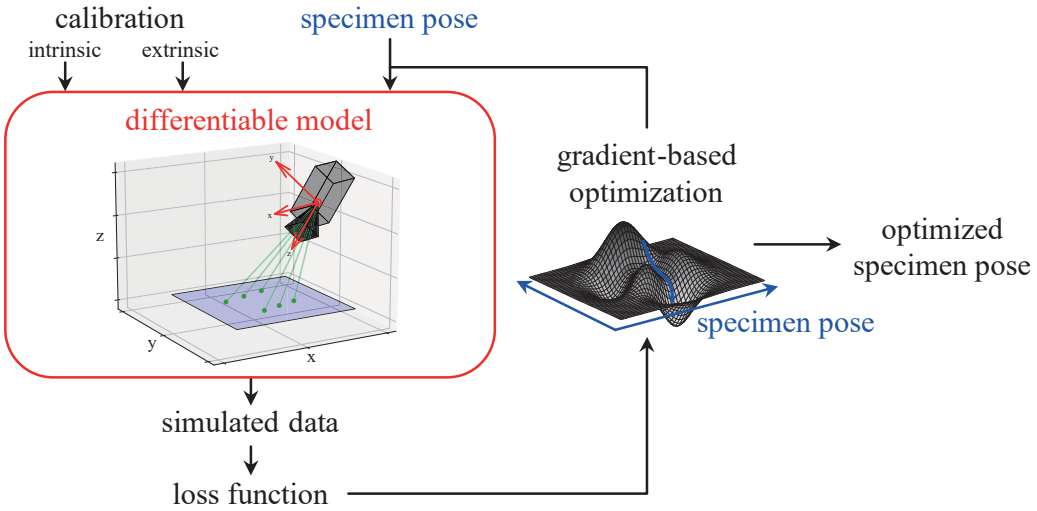

Fig 2: Framework for generalized gradient-based optimization of camera-based optical metrology systems (in this example applied for optimizing the pose of the specimen for a fixed sensor geometry) [6].

In the specific configuration shown in Fig 2, the *pose* of a for instance planar specimen under test, observed by a metrology system indicated as a camera, is optimized with respect to a

certain loss function depending on the actual measurement principle. However, by literally "connecting" the gradient-based optimization to the extrinsic calibration parameters describing the positions and orientations of the measuring system's components, rather than to the pose of the specimen as shown in Fig 2, the geometry and layout of the sensor itself can be optimized for a given measurement scenario.

Alternatively, by changing the loss function to compare simulated and real measurement data of a given (this time fixed) sensor, the same optimization framework can even be used directly for system calibration (Fig 3). Note that the differentiable model of the metrology system itself does not need to be altered at all despite the substantial difference of the various optimization tasks. Only the parameters to be optimized as well as the specific loss function are changed.

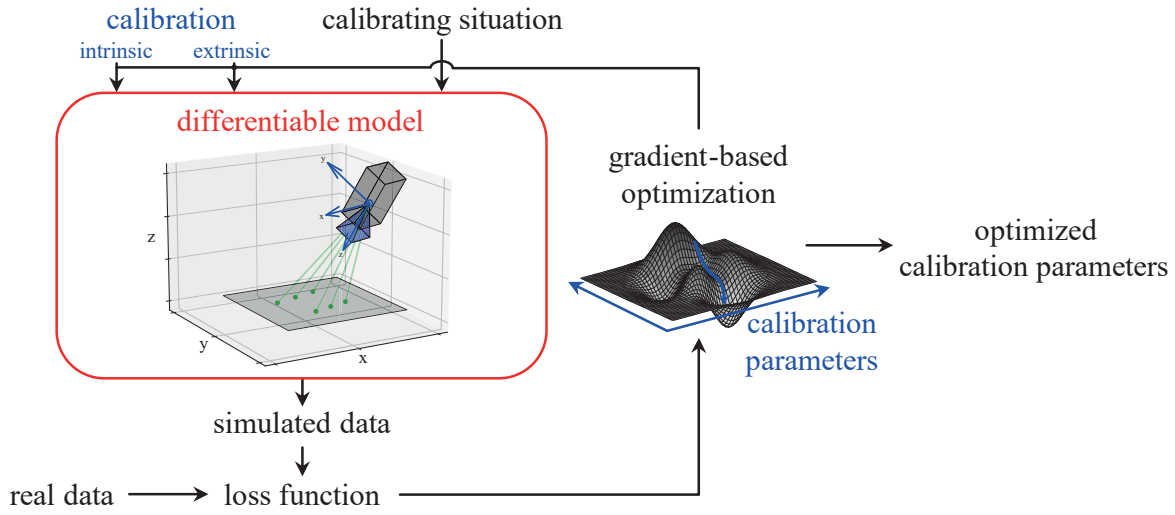

Fig 3: The same framework (and model) now applied for calibration of a (given and in this case fixed) camera-based optical metrology system. Only the parameters "connected to the optimizer" and the loss function were changed [6].

In the following, it is shown how to realize and implement such a multi-purpose model of the geometric system behavior that is particularly well suited to provide this flexibility and ease of use. It may be differentiated, for example, automatically or numerically, e.g., using platforms driven by the current development in artificial intelligence (see section 5.5), or even analytically (see section 6).

3. MODELLING THE SENSOR WITH PLANE-BASED GEOMETRIC ALGEBRA

Recently the use of *plane-based* [7,8,9,10] (or similar: *projective* [11]) *geometric algebra* (*3D-PGA*) was proposed in computer science as a unifying and elegant approach for the representation of Euclidean space.

The basic idea behind *geometric algebra* (*GA*) [12], also known as *Clifford algebra*, is that both geometric *primitives* (like points, directions, lines, or planes) and *transformations* (like translations, rotations, reflections) become elements of an algebra that are directly amenable to computation. These algebras are generated by so-called *geometric numbers*[1]:

$$e_+^2 = 1, \qquad e_-^2 = -1, \qquad e_0^2 = 0\,. \tag{1}$$

[1] Which are – completely analogous to the "invented" imaginary unit i in the case of complex numbers – themselves *not* part of the underlying real number field, but rather added as new entities in order to form a new algebra.

The *signature* $\mathbb{R}_{p,q,r}$ of the algebra indicates the set of used geometric numbers: p of the type e_+, q of e_- and r of e_0 elements [7,8].

Fundamental for GA is the *geometric product*. For two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ it can be written as [13,14]

$$\boldsymbol{uv} = \boldsymbol{u} \cdot \boldsymbol{v} + \boldsymbol{u} \wedge \boldsymbol{v} \ . \tag{2}$$

In this case, the geometric product can be split into a (symmetric) *inner* and an (antisymmetric) *outer* product, denoted as "·" respectively "∧". For two vectors, the inner product corresponds to the classical scalar product and produces a scalar, while the outer (also called *wedge*) product spans a corresponding oriented plane segment, called a *bivector*. The resulting sum of such elements with multiple grades (like scalar, vector, bivector, trivector, …) is called a *multivector*.

Note the similarity to complex numbers, which are generated from $i^2 = -1$ and consist of a sum of real and imaginary part. In fact, complex numbers can themselves be considered as a low-dimensional geometric algebra ($\mathbb{R}_{0,1,0}$) or as a subalgebra of higher-dimensional ones (e.g. the even subalgebra of $\mathbb{R}_{2,0,0}$) [15].

By virtue of the geometric product, any analytic function (i.e. function that can be expressed as a power series like *sqrt()*, *exp()*, *log()* …) can be generalized to and therefore computed for multivectors, providing a very powerful mathematical tool.

Based on the geometric product, the so-called *sandwich product*

$$\boldsymbol{mX\tilde{m}} \tag{3}$$

can be introduced[2], which is essential for transformations: The operator, in general a multivector $\boldsymbol{m}$, is multiplied from the front and (reversed) from the back to the multivector $\boldsymbol{X}$ (the operand). This structure is used in GA for applying all kinds of transformations to any primitive.

As stated above, the specific geometric algebra with signature $\mathbb{R}_{3,0,1}$ called 3D-PGA [7,8,9,10] has emerged in recent years as a unifying approach for the representation of Euclidean space. Despite its small signature[3], it naturally incorporates powerful concepts such as projective geometry or duality and allows unification of primitives and transformations. In plane-based geometric algebra, primitives exhibit the following forms ($e_1^2 = e_2^2 = e_3^2 = 1,\ e_0^2 = 0$, $e_i e_j =: e_{ij} = -e_{ji}$ and "$*$" denotes the *duality* operator) [7,8,9,10]:

$$\textbf{plane } (ax + by + cz + d = 0): \quad \boldsymbol{p} = ae_1 + be_2 + ce_3 + de_0 \qquad (\textit{vector}), \tag{4}$$

$$\textbf{line } (\text{Plücker } a, b, c, d, e, f): \quad \boldsymbol{l} = ae_{01} + be_{02} + ce_{03} + de_{12} + ee_{31} + fe_{23} \qquad (\textit{bivector}), \tag{5}$$

$$\textbf{point}^{\textbf{4}}\ (x|y|z): \quad \boldsymbol{P} = xe_{032} + ye_{013} + ze_{021} + e_{123} = (xe_1 + ye_2 + ze_3 + e_0)^* \qquad (\textit{trivector}). \tag{6}$$

[2] Analogous to a group acting on itself via the inner automorphism forming the group conjugation.

[3] Resp. (relatively) low dimension when compared with other approaches like *conformal geometric algebra*, see [9,13,16,17,18,19].

[4] The definition of the point can be best understood either as the dual of the plane defined by its component coordinates or as the "meet" of the three planes defined by $x = x_{point}$, $y = y_{point}$ and $z = z_{point}$.

Referring to (6), note that the point e_{123} is the origin. A point without e_{123} denotes a *direction* (*ideal*, "at infinity") rather than a finite position. Since primitives are homogeneous, a (non-ideal) primitive $\boldsymbol{X}$ (not "at infinity") is normalized by

$$\widehat{\boldsymbol{X}} = \frac{\boldsymbol{X}}{\|\boldsymbol{X}\|} = \frac{\boldsymbol{X}}{\sqrt{\boldsymbol{X}\bar{\boldsymbol{X}}}}. \tag{7}$$

As a result, geometric constructions formulated in 3D-PGA become extremely elegant, can be implemented by concise programming code and are advantageous in many aspects, since Plücker coordinates, dual numbers, homogeneous coordinates, or even dual quaternions are inherently embraced.

Thus, 3D-PGA can be used beneficially for concise and flexible modelling of camera-based metrology and imaging systems, including geometric optics and ray tracing [20,21,22]. As a simple example, camera-based metrology systems are commonly simulated by their inverted light ray path, called *sight rays* – given by the chief rays associated to each single pixel in the image plane. Intersecting the sight rays $\boldsymbol{r}$ of a camera, which for instance are known from the camera calibration (see section 5), with a planar object $\boldsymbol{p}$ is simply done by multiplying using the outer product (Fig 4):

$$\boldsymbol{Q} = \boldsymbol{r} \wedge \boldsymbol{p}\ . \tag{8}$$

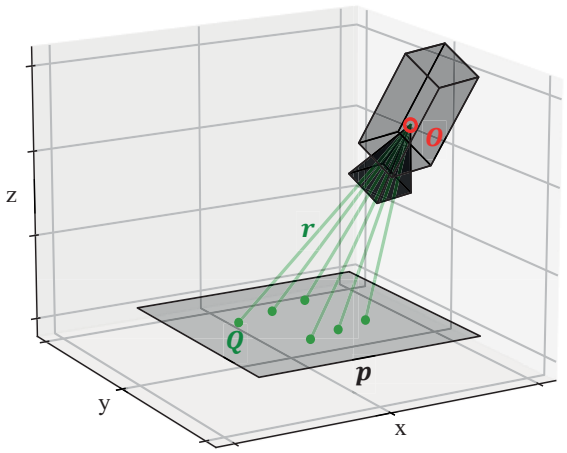

Fig 4: Intersection of sight rays with a planar object [21].

Analogous (or better said "dual") to this, the points $\boldsymbol{Q}$ and optical center of the camera $\boldsymbol{O}$ can be joined by the so-called *regressive product* ("$*$": *dual*), creating the sight ray $\boldsymbol{r}$:

$$\boldsymbol{r} = \boldsymbol{O} \vee \boldsymbol{Q} \coloneqq (\boldsymbol{O}^* \wedge \boldsymbol{Q}^*)^*. \tag{9}$$

This way, both the outer and regressive product have a direct geometric interpretation and are often referred as *"meet"* respectively *"join"*.

Please note that when using these operations, no exception handling is required in the code whatsoever: the "meet" of two parallel planes will automatically be an "ideal" element denoting a line "at infinity" that still can be used for subsequent computations, e.g. be "rotated around" – see section 4. For a more complete introduction and definition of both GA and 3D-PGA see the listed references.

4. GRADIENT-BASED POSE OPTIMIZATION WITH 3D-PGA

In 3D-PGA, rigid body motions are described by so-called *motors*. Given a rotation axis $\boldsymbol{a} = r_x e_{23} + r_y e_{31} + r_z e_{12}$ (line through the origin), with angle of rotation $\|\boldsymbol{a}\| = \sqrt{r_x^2 + r_y^2 + r_z^2}$, see (7), and a translation axis $\boldsymbol{d} = t_x e_{01} + t_y e_{02} + t_z e_{03}$ (ideal line at infinity), distance of translation $\|\boldsymbol{d}\|_\infty = \sqrt{t_x^2 + t_y^2 + t_z^2}$, both represented by bivectors (5), the transformations can be uniformly written as

$$\boldsymbol{t_{rot}} = e^{\boldsymbol{a}/2} \quad \text{and} \quad \boldsymbol{t_{trans}} = e^{\boldsymbol{d}/2}\ . \tag{10}$$

As mentioned above, the exponential function is defined by its power series using the geometric product (the same known e.g. from complex numbers or quaternions). Since in PGA transformations are built from consecutive reflections (according to the *Cartan-Dieudonné theorem*) [7,8,9,10,11], $\boldsymbol{t_{rot}}$ and $\boldsymbol{t_{trans}}$ are both *bireflections*, i.e., two consecutive reflections – hence the factor ½ in (10). As in (10) can clearly be seen, rotation and translation are unified in 3D-PGA and have six parameters (three each for rotation and translation axis) for six spatial degrees of freedom of a rigid body. The transformations can simply be combined by the geometric product to the motor

$$\boldsymbol{m} = \boldsymbol{t_{trans}} \boldsymbol{t_{rot}} \tag{11}$$

and applied to *any* primitive by (3). In fact, this is isomorphic to (dual) quaternions, which turn out to be simply the even subalgebra of 3D-PGA. Thus, motors exhibit no *gimbal lock* and are inherently easy to interpolate, average, combine and invert. To make modelling of the pose even more concise, inferring from the *Mozzi-Chasles theorem* [23,24], the most general rigid body motion in 3D-PGA is a *screw motion*, which is a combined rotation and translation "along" *one* transformation axis (*quadreflection)* [25]. Thus, (11) can be written as [26]

$$\boldsymbol{m} = e^{\boldsymbol{b}} \quad \Leftrightarrow \quad \boldsymbol{b} = \ln \boldsymbol{m} \tag{12}$$

with the bivector, given by the transformation axis (line $\boldsymbol{b}$), in general of the form[5] (5)

$$\boldsymbol{b} = a\boldsymbol{e_{01}} + b\boldsymbol{e_{02}} + c\boldsymbol{e_{03}} + d\boldsymbol{e_{12}} + e\boldsymbol{e_{31}} + f\boldsymbol{e_{23}} \, . \tag{13}$$

The gradient-based optimization stated in section 2 can directly be "connected" to these bivector coefficients a, b, c, d, e, f for pose optimization. In summary, motors are ideal for optimization purposes, lead to interpretable gradients, and are at the same time versatile and intuitive to use.

5. EXAMPLE: CAMERA CALIBRATION

It is important to note that the following example of camera calibration is aiming to demonstrate the proposed modelling methods for optimization in a succinct manner. It is *not* intended to be a complete or novel calibration approach. Accordingly, the calibration process, data and results are presented in an illustrative rather than rigorous way. To focus on the essential, several simplifications are made for illustration purposes, regarding to which the calibration can be straightforwardly extended following the proposed modelling and optimization approach, in particular:

- The calibration is performed by only one pose and acquisition of the calibration target. Of course, to increase the quality of the calibration, several should be used.
- Modelled and optimized intrinsic calibration parameters are reduced to exemplary ones like radial lens distortion. Tangential lens distortion, skew distortion within the image plane due to pixel clock problems, etc. are not considered in this example (but could easily be incorporated).
- The specific choice of gradient optimizer and its parameterization are not addressed here, which however may have a strong influence on the individual calibration result.

[5] Mind that due to the lack of commutativity of rotations and translations, $\boldsymbol{b}$ is in general *not* equal to $\boldsymbol{a}/2 + \boldsymbol{d}/2$. However, $\boldsymbol{b} = \boldsymbol{a}/2$ for $\boldsymbol{d} = 0$ and vice versa.

5.1 Calibration setup

For this example, the setup outlined in Fig 5 is used for calibration: A camera observes a uniquely decodable calibration target, in this case a planar display showing a phase-shifted sinusoidal fringe pattern (Fig 6) sequence. The resulting image of the *real* camera is shown in Fig 7 (camera pixels corresponding to sight rays in Fig 5 are marked).

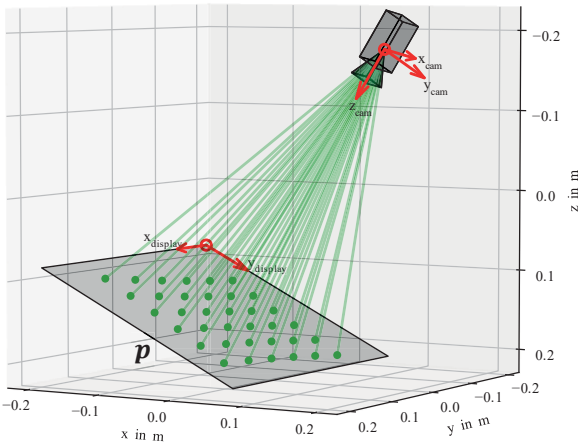

Fig 5: Model used for camera calibration.

In the following, the model of the expected system behavior shown in Fig 5 is created, which can be used to (rudimentary) simulate the system by tracing sight rays. Since the calibration parameters are at this state yet to be determined, the simulation can only be performed with (for this example quite poor) initial values for the camera calibration, resulting in a clearly discernible deviation of the fringe pattern between real (Fig 7) and simulated (Fig 8, for the following illustrations, the simulated camera image is tinted green) camera images.

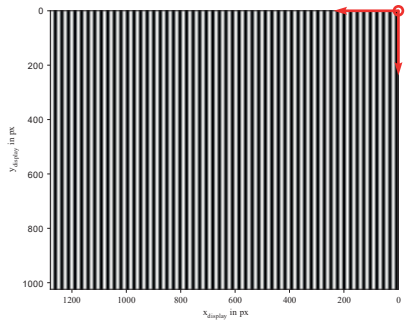

Fig 6: Phase-shifted pattern shown on display used as uniquely decodable calibration target.

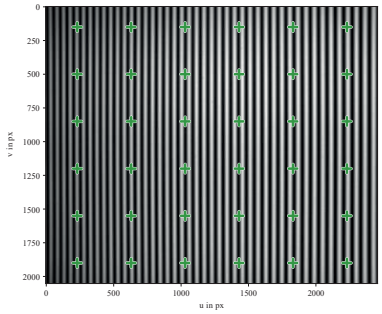

Fig 7: Real camera image (camera pixels corresponding to Fig 5 and Fig 10 are marked).

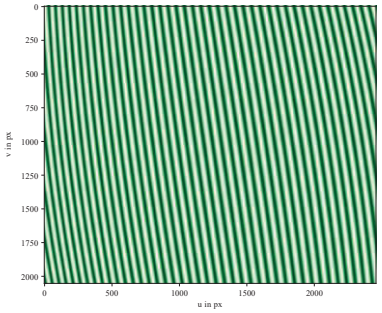

Fig 8: With start values simulated camera image (tinted green due to following illustrations).

5.2 Extrinsic calibration model

The calibration target is given as plane $\boldsymbol{p}$ (4) in world coordinates (Fig 4 and Fig 5). To describe the pose of the camera relative to world coordinates, the motor $\boldsymbol{m}$ (11,12) is used referring to section 4, with which any primitive can be transformed between world and camera coordinates by sandwiching (3). For example, the calibration target in camera coordinates is $\boldsymbol{mp\tilde{m}}$.

5.3 Intrinsic calibration model

Assuming the widely used *pinhole camera model* [27] (Fig 9), with the *camera constant* c as calibration parameter, the imaged (world) point $\boldsymbol{Q_{real}}$ is projected onto the *image plane* $\boldsymbol{c} = e_3 - ce_0$, see (4), by

$$\boldsymbol{X} = \boldsymbol{r} \wedge \boldsymbol{c} \qquad (14)$$

with the sight ray corresponding to $\boldsymbol{Q_{real}}$

$$\boldsymbol{r} = e_{123} \vee (\boldsymbol{mQ_{real}\tilde{m}}) \,. \qquad (15)$$

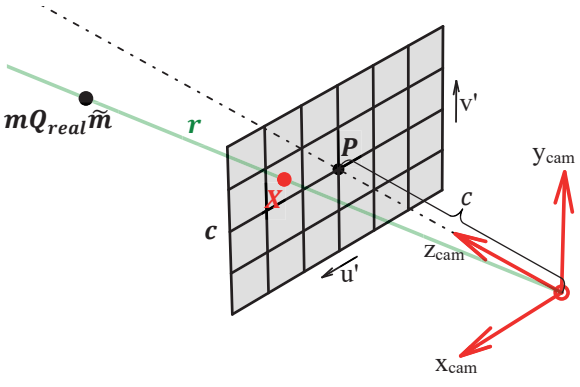

Fig 9: Pinhole camera model with image plane $\boldsymbol{c}$.

With the camera pixel size w in u- and v-direction as well as the principal point $\boldsymbol{P} = e_{12} \wedge \boldsymbol{c}$ respectively $(u'_p|v'_p)$ in camera pixels, the undistorted image point $(u'|v')$ is calculated by

$$u' = u'_p + \frac{e_1 \vee \widehat{\boldsymbol{X}}}{w_u} \qquad \text{and} \qquad v' = v'_p + \frac{e_2 \vee \widehat{\boldsymbol{X}}}{w_v} . \tag{16}$$

Finally, lens distortion (in this example for simplicity only radial distortion with coefficients k_1, k_2, k_3) is added [28,29,30,31,32]:

$$\begin{pmatrix} u \\ v \end{pmatrix} = (1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \begin{pmatrix} u' \\ v' \end{pmatrix} \quad \text{with} \quad r = \frac{1}{c} \left\| \widehat{\boldsymbol{P}} \vee \widehat{\boldsymbol{X}} \right\| . \tag{17}$$

5.4 Raytracing model

In this example, raytracing the system outlined in Fig 5 can be performed fairly simply: The sight rays $\boldsymbol{r}$ known from the intrinsic camera model in camera coordinates are traced by meeting with the calibration target $\boldsymbol{p}$ in world coordinates, resulting in (world) points $\boldsymbol{Q_{sim}}$, see (8):

$$\boldsymbol{Q_{sim}} = (\widetilde{\boldsymbol{m}}\boldsymbol{r}\boldsymbol{m}) \wedge \boldsymbol{p} . \tag{18}$$

5.5 Differentiable model and gradient-based optimization

For gradient-based optimization, the model of system behavior must be implemented in a differentiable manner, i.e., for all parameters leading into the model, the gradient with respect to the loss function must be known.

From an implementation point of view, implementing the model using platforms driven by the current development of artificial intelligence can be very advantageous, as those models are typically trained using *gradient descent*, therefore providing suitable means to calculate the required gradient (e.g. *backpropagation* for a (deep) neural network used in supervised learning). Thus, tools for (automatically) determining the gradients like *automatic differentiation* are deeply incorporated in and provided by those platforms. Furthermore, with the modularity provided by concepts like *layers*, *trainable weights*, etc., these platforms are well suited for implementing the generalized flexible framework proposed in section 2 for optimizing camera-based optical metrology and imaging systems [6].

Of course, the gradients can also be calculated otherwise, for example by further exploiting the possibilities of geometric algebra, as outlined in section 6.

5.6 Optimization

With the real measurement data $\boldsymbol{Q_{real}}$ from section 5.1 obtained by decoding the calibration target in the camera image (Fig 7) and the simulated data $\boldsymbol{Q_{sim}}$ calculated by the model of system behavior from sections 5.2 to 5.4 using equation (18), a loss function for optimizing the system's calibration parameters can for example be formulated as the root of the mean of the squared deviation (*RMS*) between the real target points $\boldsymbol{Q_{real}}$ and the simulated target points $\boldsymbol{Q_{sim}}$ corresponding to the same camera pixel:

$$loss = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \left\| \widehat{\boldsymbol{Q}}_{\boldsymbol{i,sim}} \vee \widehat{\boldsymbol{Q}}_{\boldsymbol{i,real}} \right\|^2} \tag{19}$$

For the camera pixel marked in Fig 7, this loss function is exemplarily plotted in Fig 10.

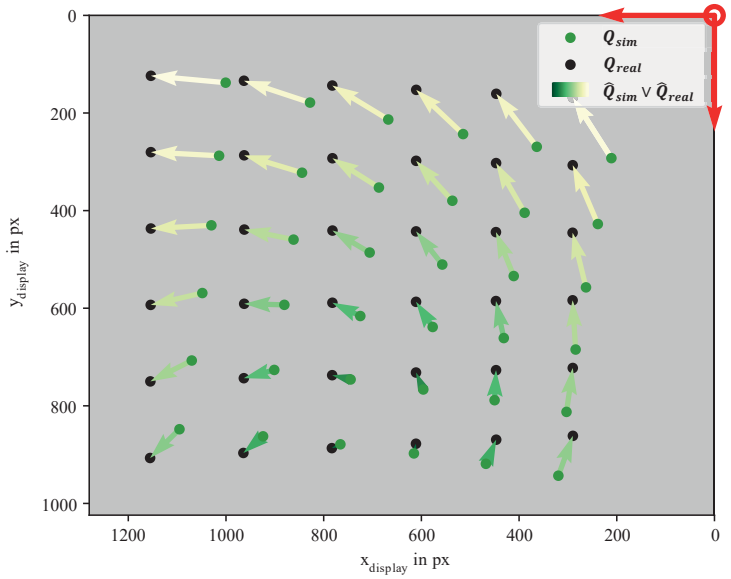

Fig 10: Loss function used for calibration and obtained from the RMS-deviation between simulated and real measurement data viewed on the calibration target $\boldsymbol{p}$.

With real (Fig 7) and simulated (Fig 8) data, the differentiable model (Fig 5) and a loss function (19), all prerequisites are in place to set up the framework shown Fig 3. To illustrate the calibration result, the simulated camera image (tinted in green, Fig 8) is superimposed on the real camera image (Fig 7) resulting in various Moiré-like patterns due to the mismatch of the imaged fringe patterns caused by the (yet) unknown calibration parameters of the model (Fig 11).

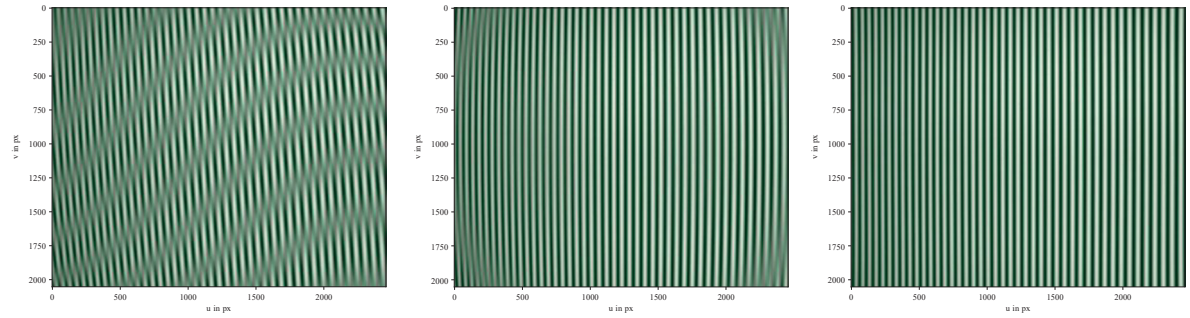

Fig 11: Superimposed real and simulated camera image (start value of optimization).

Fig 12: Result of optimization without lens distortion.

Fig 13: Result of optimization with lens distortion.

Conversely, the disappearance of these “Moiré-effects” means that the calibration parameters have been successfully optimized.

First, the gradient based optimization is literally “connected” to the bivector coefficients (12,13) of the motor of the camera, optimizing the extrinsic calibration parameters. During optimization, the optimizer tries to vary the “connected” model parameters according to the gradient (in this case only the pose of the camera) until real and simulated camera images coincide as best as possible. The optimization result is illustrated in Fig 12, where, as might be expected, obvious distortions remain at the margins and corners of the image.

Therefore, in addition to the extrinsic parameters, the lens distortion parameters (17) are “connected” for optimization according to Fig 3. Again, starting from the initial values of Fig 11, the optimization result is shown in Fig 13. Since no more “Moiré-effects” resulting from

the superposition are present, optimized calibration parameters were successfully found (Fig 14).

As mentioned at the beginning, the quality of the calibration can be enhanced without further ado by using more than one calibration target pose or by extending the model according to the proposed approach. Due to the generalization proposed in section 2, the developed model can be used directly for other optimization tasks (see Fig 2 and [6]), for example to find just these best possible poses of the calibration target for future calibrations.

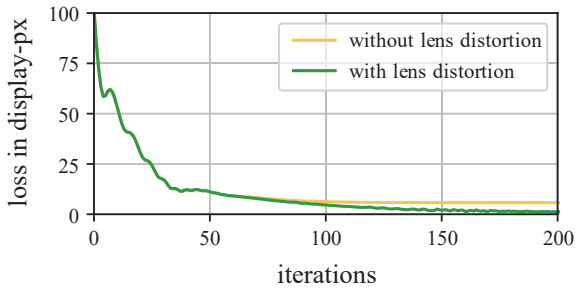

Fig 14: Loss of calibration (19) without and with lens distortion corresponding to Fig 12 and Fig 13.

6. CONCLUSION AND OUTLOOK

In conclusion, it was shown how seemingly different optimization tasks with respect to system design (optimization of system hardware) and calibration (optimization of software parameters) of camera-based optical metrology and imaging systems can be generalized using the same model of system behavior. For modelling, the unifying methods of *geometric algebra* can be used in an advantageous way, especially in combination with *gradient-based optimization*. This approach was applied, as a simple example, to calibrate a camera.

However, this beneficial dovetailing of geometric algebra and gradient-based optimization need not end here: Since 3D-PGA contains dual numbers[6], automatic differentiation and thus the determination of the model's gradients can also be performed directly within the algebra. Furthermore, due to the formal simplification provided using geometric algebra, depending on the individual model, it may even be possible to calculate the corresponding gradients to a certain extent analytically, allowing for an efficient implementation as well as a deeper understanding of the gradients, the model, and the optimization process itself.

REFERENCES

[1] A. F. Möbius: "Der barycentrische Calcul", Verlag von Johann Ambrosius Barth, Leipzig, 1827.
[2] W. R. Hamilton: "On quaternions; or on a new system of imaginaries in algebra", The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 33(219), pp. 58-60, 1848.
[3] W. R. Hamilton: "Elements of quaternions", Longmans, Green, & Company, 1866.
[4] W. K. Clifford: "Preliminary sketch of biquaternions", Proceedings of the London Mathematical Society, 1(1), pp. 381-395, 1871.
[5] A. McAulay: "Octonions: A Development of Clifford's Bi-quaternions", University Press, 1898.
[6] S. Hartel, C. Faber: "Combining simulation and optimization: Multipurpose modelling of camera-based optical metrology systems", 124th annual meeting of the DGaO, B8, 2023.
[7] C. Gunn, S. De Keninck: "Geometric Algebra for Computer Graphics", SIGGRAPH 2019.
[8] L. Dorst, S. De Keninck: "May the Forque Be with You", SIBGRAPI 2021.

[6] Or alternatively, the algebra itself can be built over dual numbers rather than real numbers.

[9] L. Dorst, D. Fontijne, S. Mann: “Geometric Algebra for Computer Science: An Object-oriented Approach to Geometry”, Amsterdam: Elsevier, 2009.
[10] L. Dorst, S. De Keninck: “A Guided Tour to the Plane-Based Geometric Algebra PGA”, version 2.0, 2022, https://geometricalgebra.net and https://bivector.net.
[11] E. Lengyel: “Projective Geometric Algebra Done Right”, 2020, https://projectivegeometricalgebra.org.
[12] W. K. Clifford: “Applications of Grassmann’s extensive algebra”, American Journal of Mathematics, 1(4): pp. 350–358, 1878.
[13] D. Hestenes, G. Sobczyk: “Clifford algebra to geometric calculus: a unified language for mathematics and physics”, Springer Science & Business Media, 5, 2012.
[14] A. Macdonald: “Linear and geometric algebra”, 2010.
[15] G. Sobczyk: “Matrix Gateway to Geometric Algebra, Spacetime and Spinors”, 2019.
[16] D. Hestenes: “Old wine in new bottles: A new algebraic framework for computational geometry”, Geometric Algebra with Applications in Science and Engineering, pp. 3-17, 2001.
[17] A. Lasenby, J. Lasenby: “Surface evolution and representation using geometric algebra”, The Mathematics of Surfaces IX: Proceedings of the ninth IMA conference on the mathematics of surfaces, pp. 144-168, Springer, London, 2000.
[18] H. Li, D. Hestenes, A. Rockwood: “Generalized homogeneous coordinates for computational geometry”, Geometric computing with Clifford algebras: theoretical foundations and applications in computer vision and robotics, pp. 27-59, 2001.
[19] A. Lasenby, J. Lasenby, R. Wareham: “A convariant approach to geometry using geometric algebra”, 2004.
[20] T. Corcovilos: “Geometric Algebra - beyond vectors”, 2019, https://www.corcoviloslab.com.
[21] S. Hartel, C. Faber: “Shedding new light on ray tracing for optical metrology systems by using Geometric Algebra”, 123rd annual meeting of the DGaO, A21, 2022.
[22] J. Lasenby, A, Stevenson: “Using geometric algebra for optical motion capture”, Geometric algebra with applications in science and engineering: pp. 147-169, 2001.
[23] G. Mozzi: “Discorso matematico sopra il rotamento momentaneo dei corpi”, 1763.
[24] M. Chasles: “Note sur les propriétés générales du système de deux corps semblables entr’eux”, Bulletin des Sciences Mathématiques, Astronomiques, Physiques et Chemiques, 14: pp. 321-326, 1830.
[25] M. Roelfs, S. De Keninck: “Graded Symmetry Groups: Plane and Simple”, 2021.
[26] S. De Keninck, M. Roelfs: “Normalization, square roots, and the exponential and logarithmic maps in geometric algebras of less than 6D”, Mathematical Methods in the Applied Sciences, 2022.
[27] Z. Zhang: “A flexible new technique for camera calibration”, IEEE Transactions on pattern analysis and machine intelligence, 22(11): pp. 1330-1334, 2000.
[28] A. E. Conrady: “Decentred lens-systems”, Monthly notices of the royal astronomical society, 79(5): pp, 384-390, 1919.
[29] D. C. Brown: “Decentering distortion of lenses”, Photogrammetric Engineering, 32(3): pp. 444-462, 1966.
[30] D. C Brown: “Close-range camera calibration”, Photogrammetric Engineering, 37(8): pp. 855-866 (1971)
[31] R. Tsai: “A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses.”, IEEE Journal on Robotics and Automation, 3(4), pp. 323-344, 1987.
[32] C.S. Fraser: “Digital camera self-calibration”, ISPRS Journal of Photogrammetry and Remote sensing, 52(4): pp. 149-159, 1997.

[33] C. Defferara, F. Negroni, A. Sarti, S. Tubaro: “New perspectives on camera calibration using geometric algebra”, International Conference on Image Processing, 2, 2002.

FUNDING

This work was supported by the *Bavarian Ministry of Economic Affairs, Regional Development and Energy* funding the *KISSMe3D* project (DIK-2105-0028 / DIK0267/02) within the funding guideline *Digitalisierung - Informations- und Kommunikationstechnologie* (*IuK)*.

CONTACTS

Simon Hartel email: simon.hartel@haw-landshut.de

Prof. Dr. Christian Faber email: christian.faber@haw-landshut.de

