


Article

Multi Sensor Extrinsic Calibration using an Extended Set of Pairwise Geometric Transformations

Vitor Santos ^{1*}, Daniela Rato ¹, Paulo Dias ² and Miguel Oliveira ¹

¹ DEM/IEETA; University of Aveiro; {vitor,danielarato,mriem}@ua.pt

² DETI/IEETA; University of Aveiro; paulo.dias@ua.pt

* Correspondence: vitor@ua.pt; Tel.: +351 927 992 318

Version November 14, 2020 submitted to *Sensors*

Abstract: Systems composed of multiple sensors for exteroceptive perception are becoming increasingly common, such as mobile robots or highly monitored spaces. However, to combine and fuse those sensors to create a larger and more robust representation of the perceived scene, the sensors need to be properly registered among them, that is, all relative geometric transformations must be known. This calibration procedure is challenging since, traditionally, human intervention is required in variate extents. This paper proposes a nearly automatic method where the best set of geometric transformations among any number of sensors is obtained by processing and combining the individual pairwise transformations obtained from an experimental method. Besides eliminating some experimental outliers with a standard criterion, the method exploits the possibility of obtaining better geometric transformations between all pairs of sensors by combining them within some restrictions to obtain a more precise transformation, hence a better calibration. Although other data sources are possible, in this approach, 3D point clouds are obtained by each sensor which correspond to the successive centers of a moving ball its field of view. The method can be applied to any sensors able to detect the ball and the 3D position of its center namely, LIDARs, mono cameras (visual or infrared), stereo cameras and TOF cameras. Results demonstrate that calibration is improved when compared to methods in previous works that do not address the outliers problem and, depending on the context, as explained in the results section, the multi pairwise technique can be used in two different methodologies to reduce uncertainty in the calibration process.

Keywords: calibration, multi-modality, extrinsic parameters, point cloud, transformation path, Chauvenet criterion, singular value decomposition, ATLASCAR.

1. Introduction

Modern robots count on a wealth of sensors for many essential operations that need perception such as representation, obstacle avoidance, planning, guidance, localization and most of the tasks generically related to navigation and safety. Sensors now cover a wide scope of principles and modalities, hence, it is no longer unexpected to see altogether monocular (both visual and infrared) and stereo cameras along with, structured-light based or TOF 3D cameras, and LiDAR (2D and 3D) mounted on some more complex systems like autonomous cars.

One of the first challenges before using such complex robotic systems is to calibrate all these sensors so their data or deduced conclusions can be merged and reported to a common coordinate frame for the algorithms to apply on a rich set of sensor data. This redundancy is necessary to give robustness and also to cover potential variations of data rate that may ultimately jeopardize single sensor or single modality based perception.

Reporting all sensors to a common frame can be simply stated as having the knowledge of where (translation and orientation) is each sensor coordinate frame relatively a common reference, normally

35 one associated or easily related to the external world being perceived. These parameters are known,
 36 for each sensor, as its extrinsic parameters.

37 Numerous works exist to calculate the extrinsic parameters of cameras or image based or image
 38 reducible sensors. External devices such as chessboards, charuco boards, or others are used to create a
 39 known real world pattern of points or geometric feature which are easily traceable on the respective
 40 images. Using these real world references and the knowledge of the projection mechanisms (through
 41 the so-called intrinsic parameters of each device), it is possible to deduce the extrinsic parameters of a
 42 sensor in relation to the target being scanned.

43 In theory, having such a marker (chessboard or similar) shown to all sensors at once would allow
 44 for the computation of the extrinsic parameters for all sensors and the problem would be solved:
 45 that marker (chessboard or similar) should only be placed in a known and interesting or convenient
 46 position to the robot and all sensors would be easily localized relatively to a common reference frame.

47 The problem is that not all sensors may be able to perceive the target in ideal conditions and not
 48 all sensors are able to detect the target with the same representation, or may not even detect the target
 49 at all. Additionally, there are uncertainties in the process.

50 An illustration of a real historical setup that will serve as base for developments ahead is shown in
 51 Figure 1 where four sensors (three LiDARS and one camera) generate a point cloud each, all obtained
 52 from the same temporal scenario (the center of a ball in several positions) but on their own coordinate
 53 frames. It is clear that the relative positions of the sensors must be determined in order to fuse or
 54 combine the four point clouds.

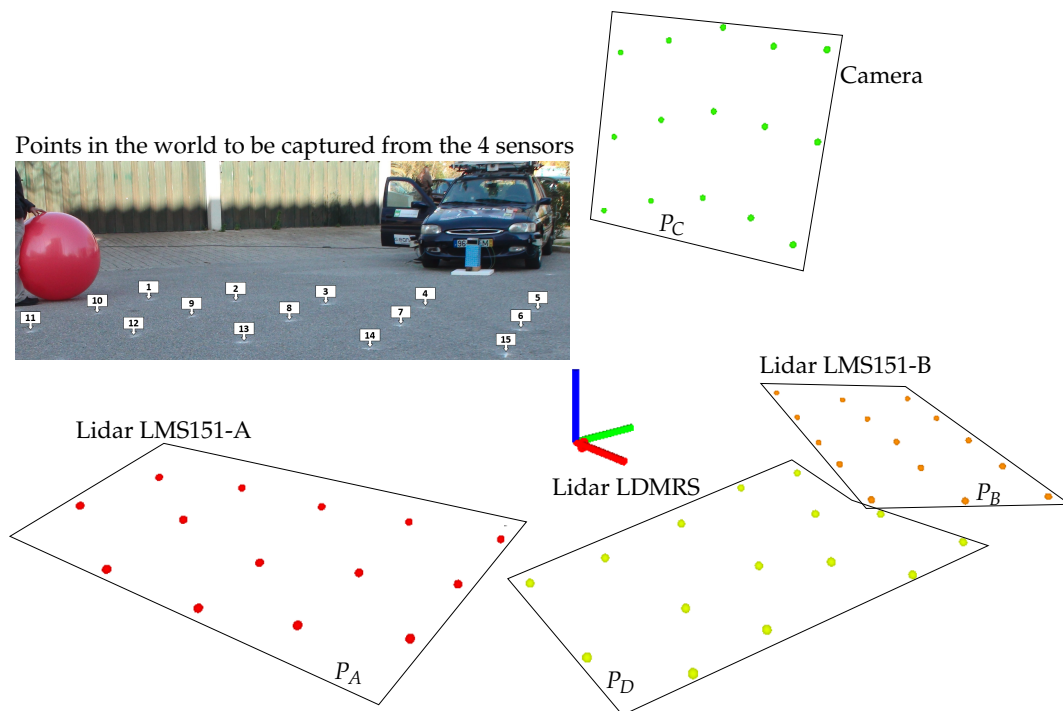


Figure 1. Example of four unregistered point clouds in ATLASCAR1 setup (adapted from [1]).

55 The main contribution of this paper is a technique to perform the extrinsic calibration of
 56 multiple sensors by eliminating outliers in the experimental process and by combining individual
 57 pairwise transformations. It improves a previously developed technique based on pairwise geometric
 58 transformations obtained from different point clouds (one for each sensor) generated with the
 59 successive center points of a moving ball.

60 The paper is divided in the following main sections: the related work, the proposed approach that
 61 includes the main algorithms described in detail, results from simulated and real data experiments,
 62 and final conclusions and future perspectives.

63 2. Related Work

64 Extrinsic calibration is a basic requirement in multi sensor platforms where data needs to be
65 represented in a common reference frame for data fusion and subsequent analysis. This calibration
66 procedure estimates the transformation between all sensors to align all data-sets in the same coordinate
67 system. Most calibration procedures are based in calibration patterns to ensure a robust and accurate
68 detection of points/objects by all sensors. Some examples of calibration patterns are chessboards [2–4],
69 fiducial markers [5–7], spherical objects [1,8–10] or cylindrical objects [11].

70 Many calibration systems are described in the literature, however there is no general solution
71 multiple sensor calibration. Pairwise calibrations between sensor is often used due to its simplicity,
72 since the calibration step does not require a global optimization. This pairwise approach must consider
73 all possible combinations between sensor modalities in the pair. These sensor combinations have
74 been addressed in the literature: RGB to RGB camera calibration [1,12–16]; RGB to depth camera
75 (RGB-D cameras) calibration [10,17–21]; camera to 2D LIDAR [1,20,22–27]; 2D LIDAR to 3D LIDAR
76 [11]; camera to 3D LIDAR [27–29]; and camera to radar [30].

77 To adapt the pairwise approach to the case of complex robotic systems that contain many sensors
78 of different modalities, several pairwise calibrations must be combined in a sequential transformation
79 procedure based on graphs where one sensor calibrates with another and then relates to a another
80 sensor, and successively. Another approach is to define one sensor as the reference and report all the
81 remainder to it. In this case, the graph of transformations is a one level pyramid with the reference
82 sensor on top and all other sensors below. This methodology is the one adopted in [1] to calibrate all
83 the sensors on-board the ATLASCAR autonomous vehicle [31] relatively to a reference sensor.

84 The problem of multi-sensor calibration can also be solved using simultaneous optimization as in
85 Liao et al. [32] that use a joint objective function to calibrate simultaneously three RGB cameras and a
86 RGB-D camera with good results in the calibration accuracy. An approach to estimate simultaneously
87 temporal offsets and spatial transformations is presented in [33]. This approach can be used for any set
88 of sensors (for example cameras and LIDAR), as its does not consider unique properties of specific
89 sensors. It also does not require the usage of calibration patterns for the LIDAR, as the planes present
90 in the scene are used for that purpose. Another relevant work occurs in [34] that proposes a calibration
91 for the sensors onboard a PR2 robot. The process uses the sensor uncertainty and is based on bundle
92 adjustment.

93 In [35], an optimization procedure is implemented which, in addition to estimate the poses of
94 the sensors, also estimates the poses of the calibration patterns. This enables the definition of errors
95 to be formulated using sensor to calibration pattern tandems, rather than the classic sensor to sensor
96 pairwise combinations. As a result, the problem of the exploding number of sensor combinations is
97 avoided, since the number of combinations do not explode with the increase in the number of sensors,
98 which makes it possible to consider all available data during the optimization.

99 In most of the mentioned approaches, there is still the need for some sort of user interaction, or
100 to provide first guess estimates for optimization based algorithms, which sometimes may be slow to
101 converge, despite the fact that that slowness may not be relevant for a offline process, which is the case
102 for many calibration procedures.

103 Overall, it appears pertinent to devise a solution of a nearly automatic mechanism with very little
104 intervention of human operation, desirably with sleek performance. One solution, already exploited
105 by other authors ([1]), is to use a simple target that must be easily detectable by all sensors, from a wide
106 range of viewpoints and perspectives, be it based on images, range maps, or even simple 2D range
107 profiles. Pereira et al. [1] used a large size ball that is easily detected by cameras and LiDARs (including
108 TOF and structured light devices). Those works have been later extended ([9]) to monochromatic and
109 infrared based images using Deep Learning techniques to detect the ball in the image with greater
110 robustness and accuracy. These solutions, however, besides still relying on pairwise approaches and
111 do not address the problem of outliers, which this papers addresses by proposing an extended solution

112 that combines the pairwise transformation and also automatically filters out outliers generated in the
 113 acquisition process.

114 The work clearly follows the line of [1] and [9] but it integrates and extends the concepts to multi
 115 pairwise procedures. Compared to existing works, the proposal that is going to be detailed in the
 116 paper, shows the following advantages: i) it can be applied to any sensor that is capable of detecting
 117 the 3D position of a moving target (ball), and not only cameras; ii) it can take all transformation paths
 118 into account or a subset of paths; iii) it is faster than iterative methods with deterministic duration and
 119 computational cost; iv) in certain conditions (explained further), it allows to reduce calibration errors
 120 in a straightforward algorithmic process.

121 3. Proposed Approach

122 Figure 1 and its associated diagram in Figure 2 illustrate the acquisition of the same scene by four
 123 different and arbitrarily positioned sensors (A, B, C, D), generating four point clouds corresponding
 124 to several positions of a known object seen by all sensors at different locations in the scene like the
 125 center of a ball, for example (P_A, P_B, P_C, P_D). For the sake of clarification, these point clouds are not
 126 actually the usual point clouds that express the geometry of a scene; they are instead a set of points
 127 that represent the time-lapsed position of a moving object in space, in this case the successive sparsely
 128 distributed positions of a ball center. But, theoretically, these two types of point clouds have similar
 129 formats and can be manipulated with common available tools.

130 Simple 3D data set matching, or more sophisticated registration techniques readily available in
 131 many libraries (PCL – Point Cloud Library or open3D, for example), can be used to obtain the position
 132 of sensors B, C and D relative to sensor A, which is assumed to be the reference of the system: this
 133 is translated by the three geometric transformations ${}^A\mathbf{T}_B, {}^A\mathbf{T}_C, {}^A\mathbf{T}_D$. In summary, for a set of four
 134 sensors (four point clouds), three matching operations are performed, that is, take one sensor (A) as
 135 the reference, and determine the position of the remainder three in relation to it.

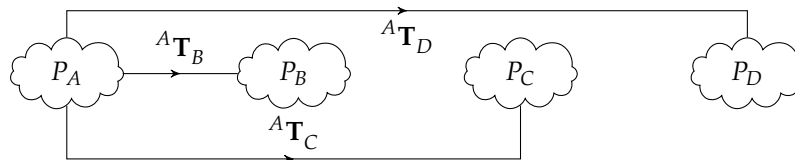


Figure 2. Example of four distinct point clouds showing the transformations of B, C and D relative to frame A. See the setup in Figure 1.

136 However, there are other transformations that can be obtained among the remainder sensors
 137 using their own point clouds besides sensor A. Figure 3 rearranges the layout and shows all possible
 138 transformation paths to obtain ${}^A\mathbf{T}_D$; a similar reasoning can be done to obtain any of the other
 139 transformations in diverse representations of equivalent transformation paths.

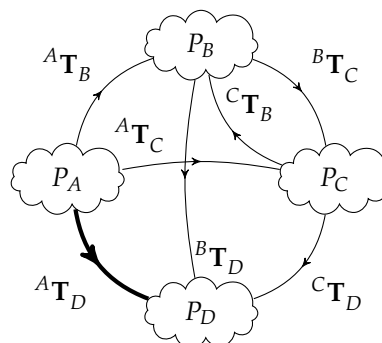


Figure 3. Rearrangement of example from Figure 2 of four distinct point clouds, but now showing all the paths for transformations from A to D, passing also through B and C.

140 We call a transformation path ${}^X\mathcal{T}_Y$, from frame X to frame Y , a sequence of geometric
 141 transformations derived from the transformation graph. An additional superscript k is used to
 142 distinguish different transformation paths between the same coordinate frames ${}^X\mathcal{T}_Y^k$. For example, as
 143 shown in Figure 3, there are five transformation paths (different ways) to obtain ${}^A\mathbf{T}_D$, after ${}^A\mathbf{T}_B$, ${}^B\mathbf{T}_C$,
 144 ${}^A\mathbf{T}_C$, ${}^C\mathbf{T}_D$, ${}^B\mathbf{T}_D$ and ${}^C\mathbf{T}_B$, reminding though that ${}^C\mathbf{T}_B = ({}^B\mathbf{T}_C)^{-1}$, that are as follows:

- 145 1. the direct measurement of ${}^A\mathbf{T}_D$ from the data matching algorithm: ${}^A\mathcal{T}_D^0 = {}^A\mathbf{T}_D$
- 146 2. ${}^A\mathcal{T}_D^1 = {}^A\mathbf{T}_B {}^B\mathbf{T}_D$;
- 147 3. ${}^A\mathcal{T}_D^2 = {}^A\mathbf{T}_C {}^C\mathbf{T}_D$;
- 148 4. ${}^A\mathcal{T}_D^3 = {}^A\mathbf{T}_B {}^B\mathbf{T}_C {}^C\mathbf{T}_D$;
- 149 5. ${}^A\mathcal{T}_D^4 = {}^A\mathbf{T}_C {}^C\mathbf{T}_B {}^B\mathbf{T}_D = {}^A\mathbf{T}_C ({}^B\mathbf{T}_C)^{-1} {}^B\mathbf{T}_D$

150 It is certain that these geometric transformations are independent because the point clouds were
 151 created from the point of view of different sensors. As an hypothetical situation: if for example
 152 the P_D point cloud has poorer quality (due perhaps to more noisy acquisition settings) the ${}^A\mathbf{T}_D$
 153 transformation would exhibit some larger uncertainties; so, combining this transformation with other
 154 transformations that involve other (expectantly) more precise point clouds will improve a final version
 155 of ${}^A\mathbf{T}_D$, but involving of course the P_D point cloud. As described ahead, a better estimate of ${}^A\mathbf{T}_D$ can
 156 be obtained by the combination of part or all of the five listed results (${}^A\mathcal{T}_D^0, {}^A\mathcal{T}_D^1, {}^A\mathcal{T}_D^2, {}^A\mathcal{T}_D^3, {}^A\mathcal{T}_D^4$). The
 157 0-th transformation path ($k = 0$) is the actual direct transformation from sensor A to D in the current
 158 example.

159 With these transformation paths it is possible to calculate a geometric transformation that is some
 160 type of combination of all the transformation paths, expectantly with a smaller uncertainty than the
 161 single original transformation.

162 To ease the interpretation of equations and algorithms, the following nomenclature convention is
 163 adopted to describe geometric transformations from sensor i to sensor j :

- 164 • ${}^i\mathbf{T}_j$ - The real transformation (usually unknown)
- 165 • ${}^i\tilde{\mathbf{T}}_j$ - The "measured" transformation (after the common localization of a unique object)
- 166 • ${}^i\hat{\mathbf{T}}_j$ - The estimated (calculated, hopefully improved) transformation.
- 167 • ${}^i\hat{\mathbf{T}}_j^k$ - A derived transformation resulting from some transformation path ${}^i\mathcal{T}_j^k$.

168 The estimated transformation is actually calculated using some sort of mean or combination of
 169 multiple derived transformations which are the result of their associated transformation paths. It is
 170 clear then, that the concept of transformation path (which results in derived transformations) is useful
 171 in a numeric approach because independent transformations can be obtained experimentally with,
 172 possibly, different levels of uncertainty, and their manipulation can provide an averaging or smoothing
 173 of those uncertainties. As stated earlier, a derived transformation (as one sample k of a larger set) is
 174 represented by ${}^n\hat{\mathbf{T}}_m^k$, which means the k -th sample of the derived transformations from sensor n to m .

Each derived transformation, for example between sensor 0 and sensor n , can be obtained in
 several ways, depending on how many steps the accumulated concatenation of transformations
 (transformation path) is done, like the following example:

$$\begin{aligned}
 {}^0\hat{\mathbf{T}}_n^1 &= {}^0\tilde{\mathbf{T}}_1 {}^1\tilde{\mathbf{T}}_n \\
 {}^0\hat{\mathbf{T}}_n^2 &= {}^0\tilde{\mathbf{T}}_2 {}^2\tilde{\mathbf{T}}_n \\
 {}^0\hat{\mathbf{T}}_n^3 &= {}^0\tilde{\mathbf{T}}_3 {}^3\tilde{\mathbf{T}}_n \\
 &\dots \\
 {}^0\hat{\mathbf{T}}_n^k &= {}^0\tilde{\mathbf{T}}_p {}^p\tilde{\mathbf{T}}_q \dots {}^m\tilde{\mathbf{T}}_n.
 \end{aligned} \tag{1}$$

175 Those derived transformations can be combined (averaged) and compared, or even merged, with
 176 the actual measurement (${}^0\tilde{\mathbf{T}}_n$) to provide a result with more confidence than the actual measurement
 177 itself, and is given in the general case by:

$${}^0\hat{\mathbf{T}}_n = \text{COMBINE} \left({}^0\hat{\mathbf{T}}_{n'}, {}^0\hat{\mathbf{T}}_{n'}^1, {}^0\hat{\mathbf{T}}_{n'}^2, {}^0\hat{\mathbf{T}}_{n'}^3, \dots, {}^0\hat{\mathbf{T}}_{n'}^k \right), \quad (2)$$

178 where $\text{COMBINE}()$ represents the function to average, merge, weight, or otherwise combine samples
 179 of a transformation between two sensors, although originated from distinct transformation paths;
 180 henceforward, the terms "average" and "combine" for geometric transformations will be used
 181 interchangeably. The algorithm for this multi pairwise approach is detailed further in section 3.2.

182 In the process, translations and rotations are expected to be combined separately and outliers are
 183 to be taken into account, as well as the confidence of each sample, should it be available or known,
 184 for example, based on the number of points present in each point cloud: assuming that, the more the
 185 points, the better is the estimation of the geometric transformation, which may be debatable, mainly
 186 because of poor detections or the presence of outliers.

187 Resuming to the example of Figure 3, there are $6 = \frac{4 \times 3}{2}$ direct feed forward transformations
 188 (pairwise independent relations) that can be obtained from the four measured point clouds. These six
 189 transformations have more information than only the three related to a single reference frame.

190 In summary, the idea is to define one sensor from the set of N sensors to be the reference (usually
 191 the one easier to place or locate in the overall reference frame of the world), naming it A, whichever it
 192 might be, and obtain the relative position of the remainder $(N - 1)$ sensors relatively the reference.
 193 Each of these $(N - 1)$ transformations is now to be obtained as an "averaging" of a number of separate
 194 geometric transformations that represent the same frame relations.

195 This implies that, whenever needed, matching operations have to be done both ways, like ${}^C\mathbf{T}_D$
 196 and ${}^D\mathbf{T}_C$; but, as ${}^D\mathbf{T}_C = ({}^C\mathbf{T}_D)^{-1}$, in principle, inverting a matrix can be performed instead of a
 197 second point cloud matching; nonetheless, most of the times, point cloud matching algorithms, namely
 198 when probabilistic approaches are used, can perform differently when source and target point clouds
 199 are swapped, and therefore, the safest option is to calculate both and pick the best instead of simply
 200 inverting matrices.

201 For a set of N sensors, a global overview of the operations can be summarized as:

- 202 • Acquire N point clouds of a reference object in several positions, one from each sensor;
- 203 • Perform at most $N(N - 1)$ point cloud matching operations or, assuming that ${}^n\hat{\mathbf{T}}_0$ will not be
 204 used, perform $(N - 1) + (N - 1)(N - 2) = (N - 1)^2$ point cloud matching operations;
 - 205 – Alternatively, perform only $(N - 1) + \frac{(N-1)(N-2)}{2} = \frac{N(N-1)}{2}$ cloud matching operations,
 206 and $\frac{(N-1)(N-2)}{2}$ inversions of transformations (in reverse directions of the previous point).
 - 207 In this case, less than $\frac{N(N-1)}{2}$ inversions are necessary because no inverse transformations
 208 are made to the reference sensor.
- 209 • Each of the $N - 1$ sensors has a set of transformation paths (connecting to the reference sensor)
 210 with different lengths (1, 2, 3, ...) where the length of some transformation path $\mathcal{L}({}^X\mathcal{T}_Y)$ is the
 211 number of transformations that compose it, as enumerated next, where $\mathcal{P}(n, r)$ is the permutation
 212 (arrangements without repetition) of n elements taken in groups of r elements:
 - 213 – length 1: $\mathcal{P}(N - 2, 0) = (N - 2)! / (N - 2)! = 1$, one path;
 - 214 – length 2: $\mathcal{P}(N - 2, 1) = (N - 2)! / (N - 3)! = (N - 2)$ paths;
 - 215 – length 3: $\mathcal{P}(N - 2, 2) = (N - 2)! / (N - 4)! = (N - 2)(N - 3)$ paths;
 - 216 – ...
 - 217 – length r : $\mathcal{P}(N - 2, r - 1)$ paths (for $r < N$);

with a total number of paths Q_N for each of the $N - 1$ sensors given by:

$$Q_N = \sum_{r=1}^{N-1} \mathcal{P}(N - 2, r - 1), \quad (3)$$

218 where N is the number of sensors, r is the length of transformation path, and \mathcal{P} means the
 219 mathematical permutation, as stated above.

- Perform $(N - 1)$ averaging operations of geometric transformations to obtain all ${}^0\hat{\mathbf{T}}_n$.

For the illustrated case of $N = 4$, for each sensor relatively to the reference sensor, there is one transformation path with length 1, two paths with length 2 and two paths with length 3, yielding a total number of transformation paths given by: $(4 - 1) \times (1 + 2 + 2) = 15$. Table 1 shows the number of transformation paths for all the possible path lengths in a set of several sensors (from 3 to 10).

N	Lengths of transformation paths $\mathcal{L}(\mathcal{T})$ for each sensor in the set									Full total
	1	2	3	4	5	6	7	8	9	
3	1	1								4
4	1	2	2							15
5	1	3	6	6						64
6	1	4	12	24	24					325
7	1	5	20	60	120	120				1956
8	1	6	30	120	360	720	720			13699
9	1	7	42	210	840	2520	5040	5040		109600
10	1	8	56	336	1680	6720	20160	40320	40320	986409

Table 1. Number of paths of the several lengths of the transformation paths for a given number of sensors N , and the full total of transformation paths in each set of sensors. The full total in each row is obtained by $(N - 1) \times \sum_r \mathcal{L}_r(\mathcal{T})$ or similarly, $(N - 1) \times Q_N$ with Q_N obtained from (3).

The number of paths grows exponentially with the number of sensors. Just as an indication, 11 sensors imply a total of nearly a million transformation paths, and 20 sensors would generate more than 10^{16} paths! For those cases where many transformation paths exist, a solution can be to limit the number of transformation paths and not use all of them, since the principle applies independently of the number of the transformation paths to be "averaged". More transformation paths should reduce the uncertainty but it is expected that after a certain number, that reduction may become negligible, hence no longer useful. This statement is a generalization of the concept of uncertainty propagation in an averaging process of N samples of some variable; if N is large, using $N + 1$ samples is not expected to reduce much further the uncertainty of the averaged result.

Possible strategies to limit the number of transformation paths to combine, for each sensor, include the following:

1. Use minimum path lengths, but ensuring all combinations of that path length — this require path lengths of 2, but all of them are necessary to involve all sensors in the estimations of each ${}^0\hat{\mathbf{T}}_n$. This would require $(N - 2)$ transformation paths to be combined with ${}^0\hat{\mathbf{T}}_n$ for each sensor.
2. Use a maximum path length which ensures that all sensors are involved as well, but no more than one path would be needed to cover all sensors. For each sensor, the estimation (averaging) would be done only with ${}^0\hat{\mathbf{T}}_n$ and ${}^0\hat{\mathbf{T}}_n^{k_L}$, where k_L is the index that corresponds to one of the maximum path lengths.
3. Use a minimum number of transformation paths, but at least as large as the number of sensors in the set and, again, ensuring that all sensors are involved. Resorting to Table 1, this would require a path length of 3 (or just 2 for three sensors). The number of paths of length 3 to average for each sensor would be N , from all the possible available in the 4-th column of Table 1.

These possibilities are presented as heuristic alternatives to the usage of all transformation paths that, besides not being sure to be needed, would be impractical to apply for large values of N . But values of N up to 6 are absolutely reasonable to use all transformation paths and keep the fast performance. There is no formal demonstration of which is the choice that produces best results, and a compromise solution with a large applicability would be to use all the transformation paths of length up to 3. Even large number of sensors would require only some hundreds of transformation paths. In this approach it is assumed that it is always possible to establish transformation path between any pair

254 of sensor with any viable path lengths. In case that turns out impossible (not all sensors share enough
255 overlap of the field of view), a smaller number of paths has to be used.

256 3.1. Pairwise Matching Algorithm for Arrays of Points

257 To calculate the pairwise best fit transformation $[\mathbf{R}|\mathbf{t}]$ between two sets of points A and B , the
258 classical technique based on [36] is used. These sets of points are sorted and there is a one-to-one
259 correspondence between the points, but the expression "point cloud" will be used instinctively as "set
260 of points". The point clouds are first relocated around the origin (by subtracting them their respective
261 centroids), and Singular Value Decomposition (SVD) is applied to the 3×3 covariance matrix of the
262 coordinates of points (in the format 3×1), as expressed with equations (4) and (5):

$$\mathbf{A}_C = \mathbf{A} - \text{centroid}_A \quad \text{and} \quad \mathbf{B}_C = \mathbf{B} - \text{centroid}_B; \quad (4)$$

$$\mathbf{H} = \mathbf{A}_C \cdot \mathbf{B}_C^T \quad \text{and} \quad [\mathbf{U} \quad \mathbf{\Sigma} \quad \mathbf{V}] = \text{SVD}(\mathbf{H}). \quad (5)$$

263 The rotation matrix and the translation vector between the point clouds are consequently
264 calculated using the expressions in (6):

$$\mathbf{R} = \mathbf{V} \cdot \mathbf{U}^T \quad \text{and} \quad \mathbf{t} = \text{centroid}_B - \mathbf{R} \cdot \text{centroid}_A. \quad (6)$$

265 3.2. The Multi Pairwise Algorithm

266 For the sake of simplicity, it is considered from now on that the sensors and their point clouds are
267 numbered starting on zero (S_0 with P_0 , S_1 with P_1 , etc.), and that the reference sensor is sensor zero, as
268 illustrated in Figure 4, where, as an example, what is sought is the position of sensor 3 (S_3) relatively to
269 sensor 0 (S_0), that is, to obtain ${}^0\hat{\mathbf{T}}_3$. In the figure, reverse transformations to S_0 are not shown because
270 they are not to be used in practice when calculating the derived transformations, that is, ${}^n\tilde{\mathbf{T}}_0$ will not
271 be part of any transformation path.

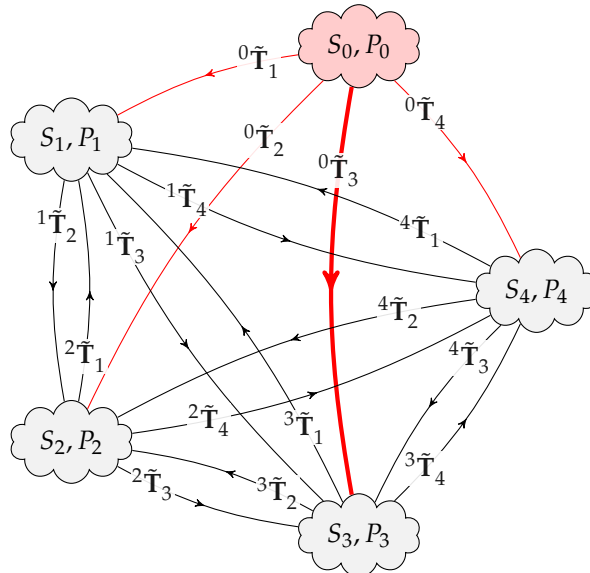


Figure 4. Example of five distinct sensors and their point clouds where all transformation paths from S_0 to S_3 , passing also through S_1 , S_2 and S_4 , can be established. In the figure, transformations from S_0 to the other sensors are actually the ones who require improved estimates based on the several transformation paths and associated derived transformations, that is, calculate ${}^0\hat{\mathbf{T}}_n$ using the several ${}^j\tilde{\mathbf{T}}_i$ present in the diagram and obtained experimentally. Both ${}^i\tilde{\mathbf{T}}_j$ and ${}^j\tilde{\mathbf{T}}_i$ may be obtained using the same registration technique or, to save time, just by performing algebraic inversion: ${}^i\tilde{\mathbf{T}}_j = ({}^j\tilde{\mathbf{T}}_i)^{-1}$.

272 The major steps of the algorithm were given earlier, but they can be detailed more specifically as
 273 follows:

- 274 • Define one sensor as the reference, S_0 , to which all other sensors will be reported/located.
- 275 • The actual calibration procedure is to obtain the list of $N - 1$ geometric transformations ${}^0\hat{T}_n$ for
 276 $n \in \{1, 2, \dots, N - 1\}$
- 277 • Acquire experimentally N point clouds P_n from the N sensors;
- 278 • Since the reference sensor is not to be part of any transformation path because it is never the
 279 destination of any transformation, perform $(N - 1) + (N - 1)(N - 2) = (N - 1)^2$ point cloud
 280 matching operations, that is, obtain all pairs of geometric transformations ${}^n\tilde{T}_m$ between sensor n
 281 and sensor m , where $n, m \in \{0, 1, \dots, N - 1\}$ and $m \notin \{0, n\}$. There is a partial alternative to
 282 this step described earlier but it is omitted here to keep the procedure shorter.
- 283 • Define which strategy is adopted to establish the set of transformation paths to use in the average
 284 calculation. As example, variant 1 from the list presented earlier is chosen, meaning to pick all
 285 transformation paths with length 2, which is the minimum length, as proposed.
- 286 • For each sensor, perform the COMBINE calculation of the results obtained in the previous step.

287 In terms of pseudo-code, the main procedure could be presented as shown in Algorithm 1.

Algorithm 1: Multi pairwise sensor calibration

```

// Nomenclature:  $tT[n][m] \leftrightarrow {}^n\tilde{T}_m$ ,  $hT[n][m] \leftrightarrow {}^n\hat{T}_m$ ,  $Tp[0][m][k] \leftrightarrow {}^0\hat{T}_m^k \Leftarrow {}^0\mathcal{T}_m^k$ 
Input :Sensors  $S[n]$ , with  $n \in \{0, 1, \dots, N - 1\}$ 
Output:Estimated geometric transformations  $hT[0][m]$ , with  $m \in \{1, \dots, N - 1\}$ 
// Obtain the point clouds from the  $N$  sensors
1 for  $n = 0$  to  $N - 1$  do
2   |  $P[n] \leftarrow \text{AcquirePointCloud}(S[n])$  // Function that returns a point cloud of ball centers
3 end
// Compute the geometric matching between all pairs of point clouds
4 for  $n = 0$  to  $N - 1$  do
5   | for  $m = 1$  to  $N - 1$  do
6     | if  $n == m$  then
7       |  $tT[n][m] \leftarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$  // Identity transformation
8     | else
9       | // If the reverse has already been matched, no need to repeat: just invert!
10      | // However, a new match could be done if different results are expected.
11      | if  $tT[m][n]$  already exists then
12        |  $tT[n][m] \leftarrow \text{invert}(tT[m][n])$  // Invert geometric transformation
13      | else
14        |  $tT[n][m] \leftarrow \text{MatchPointClouds}(S[n], S[m])$  // PCL-based or other similar functions
15      | end
16    | end
17  | end
// For each sensor, create the derived transformations and calculate the mean transformation
18 for  $m=1$  to  $N-1$  do
19   |  $Tp[0][m][0] \leftarrow tT[0][m]$ 
20   | for  $k = 1$  to  $N - 2$  // The limit  $(N - 2)$  covers all paths with length 2 (see Table 1)
21     |  $Tp[0][m][k] \leftarrow \text{DerivedTransformation}(tT, 0, m, k, 2, N)$ 
22   | end
23   |  $hT[0][m] \leftarrow \text{MeanTransformation}(Tp[0][m][\ ])$  // For all  $k$ 
24 end

```

288 When ranging the transformation path number from $k = 0$ to some limit $k = k_L$, an order is
 289 expected, where $k = 0$ corresponds to a path with a single transformation (${}^n\mathcal{T}_m^0 = {}^n\hat{\mathbf{T}}_m$) and the
 290 following values of k correspond to an increasing amount of accumulated transformations. For
 291 example, when $N = 5$, for each of the four sensors relatively to the reference sensor, the following
 292 number of path lengths correspond to the indicated values of k :

Path lengths for $N = 5 \rightarrow$	1	2	3	4
Number of paths \rightarrow	1	3	6	6
Values of index $k \rightarrow$	0	1,2,3	4,5,6,7,8,9	10,11,12,13,14,15

294 Two of the most relevant functions from Algorithm 1 are `DerivedTransformations()` and
 295 `MeanTransformation()`. The latter can have several formulations as described further (being the
 296 SVD approach the most straightforward to implement), but the first needs a little more explanation.
 297 The function requires the indices of the transformations from n to m of length L for path k , and use
 298 those indices to compose (post-multiply) the respective transformations $tT[i][j]$. The function and
 299 associate parameters are stated as follows:

300 Function $T = \text{DerivedTransformation}(tT, n, m, k, L, N)$

301	T	The return value: the derived transformation path $Tp[0][m][0]$
302	tT	Array of transformation matrices for all pairs of sensors
303	n	Starting sensor (usually 0, but could be extended to be any)
304	m	Ending sensor (any, except sensor 0, but could be extended)
305	k	Number of the transformation path
306	L	Length of the transformation paths to use
307	N	Total number of sensors in the problem

308 According to what was stated earlier, for this call, the parameters k and L are redundant, but the
 309 function can be prepared for both approaches in the calling: if k is valid ($k \geq 0$) it has priority over L
 310 (which is then ignored or potentially used to confirm that there is no discrepancy between the desired k
 311 and the corresponding L); on the other hand ($k < 0$, which is an invalid index), the first path of length
 312 L could be used and the appropriate value of k is assumed to perform the operation. For example,
 313 in the previous case, for $N = 5$, the following call $T = \text{DerivedTransformation}(tT, 0, 4, 7, 2, 5)$ could
 314 trigger an alert because the value used for k ($= 7$) corresponds to a path length of $L = 3$ and not 2 as
 315 stated in the call! Still, as the path length of 3 remains compatible with the indicated number of sensors
 316 ($= 5$), the calculation could be done, and the proper transformation for $k = 7$, $L = 3$ and $N = 5$ would
 317 be returned.

318 To enable all this checking, the function must be able to assess the entire set of values for k for a
 319 given number of sensors N and for each path length L . That is given generically as shown next, where
 320 the number of path lengths ranges from 1 to $N - 1$ and k is indeed function of N and L :

L	1	2	3	...	$N - 1$
$k(N, L)$	0	1	$(N - 2) + 1$...	$\prod_{r=2}^{L-1} (N - r) + 1$
$k(N, L)$	0	2	$(N - 2) + 2$...	$\prod_{r=2}^{L-1} (N - r) + 2$
$k(N, L)$	0	\vdots	\vdots	\vdots	\vdots
$k(N, L)$	0	$(N - 2)$	$(N - 2) + (N - 2)(N - 3)$...	$\prod_{r=2}^{L-1} (N - r) + \prod_{r=2}^L (N - r)$

In a more compact form, the previous statements can be summarized as:

$$k(N, L) \in \{K_{min}, \dots, K_{max}\} = \left\{ \prod_{r=2}^{L-1} (N - r) + 1, \dots, \prod_{r=2}^{L-1} (N - r) + \prod_{r=2}^L (N - r) \right\}, \quad (7)$$

knowing, of course, that these expressions are applicable for $N \geq 3$ and $L < N$, as is also verifiable in Table 1. Curiously, and it is simple to demonstrate, the following also holds:

$$K_{max} = (K_{min} - 1)(N - L + 1). \quad (8)$$

322 To perform the computation of the k -th derived transformation (result of its correspondent
323 transformation path), the function `DerivedTransformation()` needs the list of all permutations
324 (arrangements) of $(N - 2)$ sensors taken in groups of L , but only those that end in the target sensor.

325 For example, five sensors taken in groups of 2, because path lengths of 2 are required, and
326 excluding sensor 0 — the reference, yield the following ordered permutations: (1 2), (1 3), (1 4), (2
327 1), (2 3), (2 4), (3 1), (3 2), (3 4), (4 1), (4 2), (4 3), but, when the target sensor to calibrate relatively
328 sensor 0 is defined (and using the example of sensor 3), only the sequences that end in 3 are desired
329 for further calculation: (1 3), (2 3), (4 3), that is, all the permutations of sensors $\{1, 2, 4\}$ in groups of 1
330 ($1 = 2 - 1 = L - 1$) are needed. So, the numbers of sensors in sequence whose transformations are
331 to be obtained and further combined are: $0 \triangleright Perm(\{1, 2, 4\}, 1) \triangleright 3$, where the symbol \triangleright denotes a
332 transformation between the associated pair of sensors or, more explicitly:

$$\begin{aligned} 333 & \bullet k = 1 \Rightarrow \{S_0 \triangleright S_1 \triangleright S_3\} \Rightarrow {}^0\hat{\mathbf{T}}_3^1 = {}^0\tilde{\mathbf{T}}_1^1 \tilde{\mathbf{T}}_3 \\ 334 & \bullet k = 2 \Rightarrow \{S_0 \triangleright S_2 \triangleright S_3\} \Rightarrow {}^0\hat{\mathbf{T}}_3^2 = {}^0\tilde{\mathbf{T}}_2^2 \tilde{\mathbf{T}}_3 \\ 335 & \bullet k = 3 \Rightarrow \{S_0 \triangleright S_4 \triangleright S_3\} \Rightarrow {}^0\hat{\mathbf{T}}_3^3 = {}^0\tilde{\mathbf{T}}_4^4 \tilde{\mathbf{T}}_3 \end{aligned}$$

336 As another example, if $N = 6$ and $L = 3$ starting in S_0 and ending on S_4 would result in the
337 following 12 sensor sequences to use: $0 \triangleright Perm(\{1, 2, 3, 5\}, (3 - 1)) \triangleright 4$ or, in expanded form: (1 2 4),
338 (1 3 4), (1 5 4), (2 1 4), (2 3 4), (2 5 4), (3 1 4), (3 2 4), (3 5 4), (5 1 4), (5 2 4), (5 3 4), still assuming, of course,
339 that sensor 0 starts all sequences.

In conclusion, when function `DerivedTransformation(tT, n, m, k, L, N)` is called, for the sake and application in this paper, it expects the following integers and limits:

$$n = 0, \quad N > 2, \quad 0 < L < N, \quad 0 < m < N \quad (9)$$

340 and tT is an array with all pairs of geometric transformations previously calculated in Algorithm 1.
341 The relevant code of the function is described in Algorithm 2.

342 However, if the length of the transformation paths is to be restricted to $L = 2$ (as proposed earlier
343 for the chosen solution to limit the number of operations), the procedures are simpler and Algorithm 2
344 can be simplified and proposed as in Algorithm 3. Hence, the function `GetPathPairSequences(N, m)`
345 from Algorithm 3 is much simpler than its general variant `GetSensorPathSequences(N, L, m)` from
346 Algorithm 2 and provides the following sequences, being $m > 1$ and $N > 2$:

$$\begin{aligned} 347 & \bullet 0 \triangleright 1 \triangleright m \\ 348 & \bullet 0 \triangleright \dots \triangleright m \\ 349 & \bullet 0 \triangleright m - 1 \triangleright m \\ 350 & \bullet 0 \triangleright m + 1 \triangleright m \\ 351 & \bullet 0 \triangleright \dots \triangleright m \\ 352 & \bullet 0 \triangleright N - 1 \triangleright m \end{aligned}$$

353 As mentioned earlier, the reference sensor is named zero and it is always the start of the
354 transformation paths for any given target sensor. Should the reference sensor be another, a renaming of
355 the sensors would be done to establish the new reference sensor, that is, the new sensor zero. Despite
356 that possibility, the methodology remains unchanged, only the sensor numbering is affected.

357 3.3. Combination of Geometric Transformations

Combining or averaging a set of transformations can be considered as averaging translations and rotation angles. Therefore, for a set of K transformation matrices given by $\mathbf{T}_i = \left[\mathbf{R}_i \mid \mathbf{t}_i \right]$, the average

Algorithm 2: Function to perform the computation of derived transformations

Input :array of transformation pairs tT , starting sensor n , target sensor m , number of the transformation path k , length of transformation path L , number of sensors N

Output: Derived geometric transformation: Tp

```

1 Function DerivedTransformation( $tT, n, m, k, L, N$ )
2   if  $n \neq 0$  then
3     | return error // Was expecting the starting sensor to be 0
4   end
5    $K_{min} \leftarrow \prod_{r=2}^{L-1} (N - r) + 1$  // From equation (7)
6    $K_{max} \leftarrow \prod_{r=2}^{L-1} (N - r) + \prod_{r=2}^L (N - r) = (K_{min} - 1)(N - L + 1)$  // From equations (7) and (8)
7   if  $k \geq K_{min}$  and  $k \leq K_{max}$  then
8     |  $seqs \leftarrow$  GetSensorPathSequences( $N, L, m$ ) // Get all paths for these parameters
9   else
10    | return error // Arguments not consistent
11  end
12   $idx \leftarrow k - K_{min}$  // Normalise index to start in 0
13   $seqk \leftarrow seqs[idx]$  // Pick the idx-th sequence
14   $T \leftarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$  // Initialise with identity transformation
15  for  $t = 0$  to  $L - 1$  do
16    |  $T \leftarrow T \times tT[seqk[t]][seqk[t + 1]]$  // Accumulate transformations by post-multiplication
17  end
18   $Tp \leftarrow T$ 
19 return  $Tp$ 

```

Algorithm 3: Simplified function to compute derived transformations with $L = 2$

Input :array of transformation pairs tT , target sensor m , number of the transformation path k , number of sensors N

Output: Derived geometric transformation: Tp

```

1 Function DerivedTransformationSimple( $tT, m, k, N$ )
2   if  $k \geq 1$  and  $k \leq (N - 2)$  then
3     |  $seqs \leftarrow$  GetPathPairSequences( $N, m$ )
4   else
5     | return error // Arguments not consistent
6   end
7    $seqk \leftarrow seqs[k - 1]$  // Pick the  $(k - 1)$ -th sequence
8    $Tp \leftarrow tT[seqk[0]][seqk[1]] \times tT[seqk[1]][seqk[2]]$ 
9 return  $Tp$ 

```

transformation matrix has a translation vector which is given by $\hat{\mathbf{t}} = \frac{1}{K} \sum_i^K \mathbf{t}_i$ and a rotation matrix $\hat{\mathbf{R}}$ obtained by an operation of "averaging" the various \mathbf{R}_i . A few approaches can be considered for this operation of merging the various \mathbf{R}_i : quaternions, Euler angles and single value decomposition (SVD) are the most likely candidates. Quaternions and Euler angles based approaches both require a conversion between representations, but SVD does not. On the other hand, the Euler angles technique allows a true mean value calculation and also a weighted mean (in case it is necessary), making that

approach very versatile in this context. Adopting the RPY version of Euler angles (but actually any triple of Euler angles would do), that formulates like this:

$$\begin{aligned} \mathbf{M} &= \text{RPY}(\phi, \theta, \psi) = \text{rotz}(\phi) \times \text{roty}(\theta) \times \text{rotx}(\psi) \\ &= \begin{bmatrix} C\phi C\theta & C\phi S\psi S\theta - C\psi S\phi & S\phi S\psi + C\phi C\psi S\theta \\ C\theta S\phi & C\phi C\psi + S\phi S\psi S\theta & C\psi S\phi S\theta - C\phi S\psi \\ -S\theta & C\theta S\psi & C\psi C\theta \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \end{aligned} \quad (10)$$

and being $\theta \in]-\pi/2, +\pi/2[$, then, the correspondent Euler angles can be expressed as:

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \mathcal{E}(\mathbf{M}) = \begin{bmatrix} \arctan(r_{21}, r_{11}) \\ \arctan\left(-r_{31}, \sqrt{r_{32}^2 + r_{33}^2}\right) \\ \arctan(r_{32}, r_{33}) \end{bmatrix}. \quad (11)$$

If $\theta = \pm\pi/2$ ($r_{11} = r_{21} = r_{32} = r_{33} = 0$), ϕ or ψ can be arbitrary (gimbal lock). In that case, we can adopt the convention: $\psi = 0$ and $\phi = \arctan(-r_{12}, r_{22})$.

Formally, the "mean rotation" is obtained using all K samples of ϕ_i, θ_i, ψ_i in the following manner:

$$\hat{\mathbf{R}} = \text{RPY}\left(\frac{1}{K} \sum_i^K \phi_i, \frac{1}{K} \sum_i^K \theta_i, \frac{1}{K} \sum_i^K \psi_i\right) \quad (12)$$

If individual transformations have some normalised degree of confidence μ_i , where $\sum_i^K \mu_i = 1$, then rotation and translation "averages" can be obtained in the classic way:

$$\hat{\mathbf{R}} = \text{RPY}\left(\frac{1}{K} \sum_i^K \mu_i \phi_i, \frac{1}{K} \sum_i^K \mu_i \theta_i, \frac{1}{K} \sum_i^K \mu_i \psi_i\right) \quad \text{and} \quad \hat{\mathbf{t}} = \frac{1}{K} \sum_i^K \mu_i \mathbf{t}_i \quad (13)$$

An even more compact and straightforward approach, even though with slightly different results (due to the least square fit technique associated) is to use SVD. First, we calculate the decomposition using traditional tools:

$$[\mathbf{U} \quad \mathbf{\Sigma} \quad \mathbf{V}] = \text{SVD}\left(\sum_i \mathbf{R}_i\right) \quad (14)$$

and then obtain the "average rotation" $\hat{\mathbf{R}}$ using:

$$\hat{\mathbf{R}} = \mathbf{V}\mathbf{U}^T. \quad (15)$$

In case different degrees of confidence in the geometric transformations are available, we can weight the various transformations \mathbf{R}_i using the following:

$$[\mathbf{U} \quad \mathbf{\Sigma} \quad \mathbf{V}] = \text{SVD}\left(\sum_i \mu_i \mathbf{R}_i\right), \quad (16)$$

assuming that $\sum_i \mu_i = 1$, but not necessarily. Indeed, if a given \mathbf{R}_i has a larger confidence (for example the double of the remainder) it would be added twice in expression (16). In all cases, the final "average" transformation is given, of course, by: $\hat{\mathbf{T}} = [\hat{\mathbf{R}} \mid \hat{\mathbf{t}}]$.

4. Propagation of Uncertainty with a Simulated Experiment

To test the properties and potential advantages of the approach, we perform a simulation considering a set up based on the ATLASCAR2 prototype [37]. Figure 5 shows the car with four sensors placed on their own coordinate frames (F_0, F_1, F_2, F_3). The geometric transformations between

370 all pairs (${}^0\tilde{\mathbf{T}}_1, {}^0\tilde{\mathbf{T}}_2, {}^0\tilde{\mathbf{T}}_3, {}^1\tilde{\mathbf{T}}_2, {}^1\tilde{\mathbf{T}}_3, {}^2\tilde{\mathbf{T}}_3$) are measured experimentally, most likely with some errors both
 371 in translation and rotation.

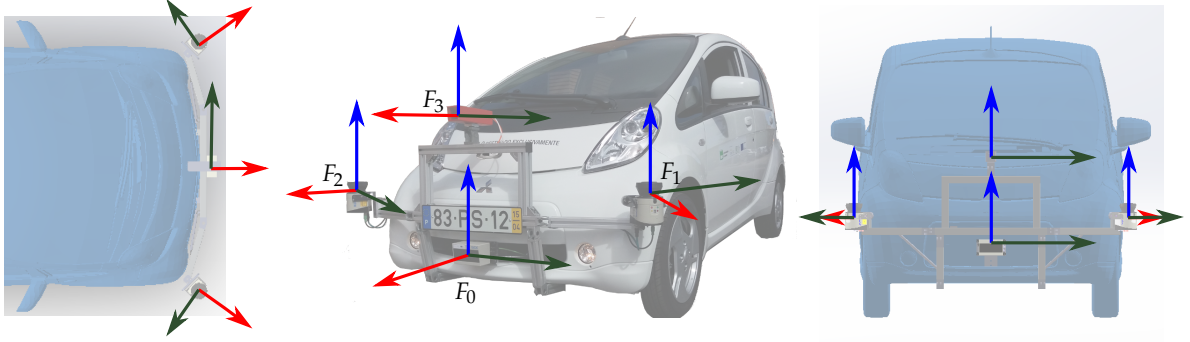


Figure 5. Example of four sensors in ATLASCAR2 vehicle.

372 This section will demonstrate with numerical examples and a systematic analysis that using
 373 combinations of multiple pairwise transformations, within certain conditions, decreases the error that
 374 occurs in each simple transformation between pairs of sensors in the car.

For this experiment we use the real relative postures in terms of position (x, y, z) , in meters, and Euler angles (ϕ, θ, ψ) , in degrees, of the remainder three sensors respectively to the reference sensor (F_0), using the notation $\{x, y, z, \phi, \theta, \psi\}$, are defined as follows:

$${}^0\tilde{\mathbf{T}}_1 \rightarrow \{-0.05, -1, 0.25, 35, 0, 0\}, {}^0\tilde{\mathbf{T}}_2 \rightarrow \{-0.05, 1, 0.25, -35, 0, 0\}, {}^0\tilde{\mathbf{T}}_3 \rightarrow \{-0.02, 0, 0.50, 0, 0, 0\}. \quad (17)$$

The approximate numerical values of the transformation matrices are then given by:

$${}^0\tilde{\mathbf{T}}_1 = \begin{bmatrix} 0.8192 & -0.5736 & 0 & -0.05 \\ 0.5736 & 0.8192 & 0 & -1 \\ 0 & 0 & 1 & 0.25 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^0\tilde{\mathbf{T}}_2 = \begin{bmatrix} 0.8192 & 0.5736 & 0 & -0.05 \\ -0.5736 & 0.8192 & 0 & 1 \\ 0 & 0 & 1 & 0.25 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^0\tilde{\mathbf{T}}_3 = \begin{bmatrix} 1 & 0 & 0 & -0.02 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.05 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (18)$$

We propose to determine ${}^0\hat{\mathbf{T}}_1$ by applying the algorithms described in the previous sections. Since we have four sensors in the setup ($N = 4$) we have five (see Table 1) possible transformation paths (19):

$$\begin{aligned} {}^0\hat{\mathbf{T}}_1^0 &= {}^0\tilde{\mathbf{T}}_1 \\ {}^0\hat{\mathbf{T}}_1^1 &= {}^0\tilde{\mathbf{T}}_2 {}^2\tilde{\mathbf{T}}_1 \\ {}^0\hat{\mathbf{T}}_1^2 &= {}^0\tilde{\mathbf{T}}_3 {}^3\tilde{\mathbf{T}}_1 \\ {}^0\hat{\mathbf{T}}_1^3 &= {}^0\tilde{\mathbf{T}}_2 {}^2\tilde{\mathbf{T}}_3 {}^3\tilde{\mathbf{T}}_1 \\ {}^0\hat{\mathbf{T}}_1^4 &= {}^0\tilde{\mathbf{T}}_3 {}^3\tilde{\mathbf{T}}_2 {}^2\tilde{\mathbf{T}}_1 \end{aligned} \quad (19)$$

For the purpose of this experiment, we do not use any point clouds and we assume that the transformations are already available (for example, after using the matching technique mentioned earlier). To continue the experiment, we need all the multiple pairs of transformations ${}^2\tilde{\mathbf{T}}_1, {}^3\tilde{\mathbf{T}}_1, {}^2\tilde{\mathbf{T}}_3$, etc., which are calculated by simple algebraic manipulation: for example, ${}^2\tilde{\mathbf{T}}_1 = ({}^0\tilde{\mathbf{T}}_2)^{-1} {}^0\tilde{\mathbf{T}}_1$, and so on. As it is expected, with these "perfect" geometric transformations, all the procedures presented earlier, namely the average transformation given by expression (15) and related, produce the perfect result. In other words, the following operations

$$[\mathbf{U} \quad \Sigma \quad \mathbf{V}] = \text{SVD} \left({}^0\hat{\mathbf{R}}_1 + {}^0\hat{\mathbf{R}}_1^1 + {}^0\hat{\mathbf{R}}_1^2 + {}^0\hat{\mathbf{R}}_1^3 + {}^0\hat{\mathbf{R}}_1^4 \right) \quad \text{with} \quad {}^0\hat{\mathbf{R}}_1 = \mathbf{U}\mathbf{V}^T, \quad (20)$$

375 confirm that ${}^0\hat{\mathbf{R}}_1 = {}^0\tilde{\mathbf{R}}_1$, and the same for the translation part, which is easier to calculate (a
 376 simple arithmetic mean). However, the important issue here is to study the effect of errors in the
 377 transformations and how they propagate through the operations, and the calculation of the average

378 transformation to reduce the error in the final ${}^0\hat{\mathbf{T}}_1$. For this purpose, a systematic test of uncertainty
 379 propagation was carried out using a Monte Carlo Simulation (MCS) to study the propagation of errors
 380 when multiplying the geometric transformations given in (19), and their subsequent combination.

381 The uncertainty in the transformation matrices may have origin in any of the the six variables:
 382 $\phi, \theta, \psi, t_x, t_y, t_z$. So, to apply the MCS method to study uncertainty propagation, an explicit expression
 383 for these variables was derived for all matrix multiplications in (19) using expression (11) for the
 384 angles, and the three lines of the fourth column of the resulting matrix for the translation.

To perform this operation, the nominal values presented in (17) are used with an additional term that represents the uncertainty in each variable for each of the three base matrices, yielding a total of six sources of uncertainty for each matrix involved. Follows in (21) the example of one of those transformation matrices with the six uncertainty values:

$${}^0\hat{\mathbf{T}}_1^\Delta = \begin{bmatrix} 1 & 0 & 0 & t_{x_1} + \Delta x_1 \\ 0 & 1 & 0 & t_{y_1} + \Delta y_1 \\ 0 & 0 & 1 & t_{z_1} + \Delta z_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \text{RPY}(\phi_1 + \Delta\phi_1, \theta_1 + \Delta\theta_1, \psi_1 + \Delta\psi_1). \quad (21)$$

385 For the the purpose of MCS, a new notation is introduced to identify a matrix that includes the
 386 uncertainty terms to study the uncertainty propagation: ${}^0\hat{\mathbf{T}}_2^\Delta$.

387 However, to fully simulate uncertainty propagation in the expressions of (19), intermediate
 388 matrices, such as ${}^2\tilde{\mathbf{T}}_1, {}^3\tilde{\mathbf{T}}_1, {}^2\tilde{\mathbf{T}}_3$, etc., are also needed, but they are not available because in this
 389 simulation there are no point clouds to extract the matrices from them. So, these matrices have to
 390 be derived from combinations of the others, but with their own terms of uncertainty and not the
 391 terms from the others that originate them by multiplication or inversion. This is necessary to have an
 392 unbiased experience and, of course, the consequence is to add more degrees of freedom for the MCS.

393 For example, ${}^2\hat{\mathbf{T}}_3^\Delta$ can be obtained after $({}^0\hat{\mathbf{T}}_2^\Delta)^{-1} \times {}^0\hat{\mathbf{T}}_3^\Delta$ and if it were to be studied on its own (the
 394 propagation of uncertainty that reached it), no further manipulations would be required to study this
 395 uncertainty propagation. The issue is that ${}^2\hat{\mathbf{T}}_3^\Delta$ is to be used in other expressions and for simulation
 396 purposes there should not exist unrelated terms of uncertainty with the same name! So, for this
 397 example and all the others in a similar situation, the terms of uncertainty were renamed to allow true
 398 independence during the MCS tests. The procedure was to replace the $[\Delta\phi_n, \Delta\theta_n, \Delta\psi_n, \Delta t_{x_n}, \Delta t_{y_n}, \Delta t_{z_n}]$
 399 in ${}^2\hat{\mathbf{T}}_3^\Delta$ by new terms, namely in this specific case by $[\Delta\phi_{n32}, \Delta\theta_{n32}, \Delta\psi_{n32}, \Delta t_{x_{n32}}, \Delta t_{y_{n32}}, \Delta t_{z_{n32}}]$.

400 For each of the matrix operations from (19), the MCS method was applied to the intricate analytic
 401 expressions resulting from the application of (11), where the variables depend on many sources of
 402 uncertainty: for example, matrix ${}^0\hat{\mathbf{T}}_1^{4\Delta}$ depends on 30 sources of error from the accumulated operations
 403 in this simulation. This is actually a worst case scenario not usually occurring with experimental data.

404 Both uniform and Gaussian probability density functions (PDF) were used and several uncertainty
 405 values (standard deviation) were tested for the angles and for the translation components. To ease the
 406 implementation of the process, in each trial, the same value in radians for angles and in meters for
 407 translations was used for all parameters. This option was taken for simple simulation convenience;
 408 hence, an input uncertainty of 0.1 results in 0.1 rad, which is about 5.7° , and 0.1 m for the translation
 409 parts. The parameters of the full experiment are the following:

- 410 • Number of samples for the MCS: 10^5 for each transformation path;
- 411 • Standard deviations of input uncertainties:
 - 412 – for translations (in meter): {0.01, 0.02, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5};
 - 413 – for angles: { $0.6^\circ, 1.1^\circ, 2.9^\circ, 5.7^\circ, 8.6^\circ, 11.5^\circ, 14.3^\circ, 17.2^\circ, 20.1^\circ, 22.9^\circ, 25.8^\circ, 28.6^\circ$ };
- 414 • PDF for the samples: Gaussian and uniform distributions;

415 The final results are of the same nature for all six functions for the four transformation paths (five
 416 with the direct ${}^0\hat{\mathbf{T}}_1^\Delta$). This means that the PDF of the uncertainties is preserved to the final output,
 417 namely for the Gaussian PDF (Figure 6). Curiously, uniform PDF in the input maps also as uniform

418 PDF for the variables in matrix ${}^0\hat{\mathbf{T}}_1^\Delta$ (shown in the third histogram of Figure 6), but maps to Gaussian
 419 PDF for all the other (${}^0\hat{\mathbf{T}}_1^{1\Delta}$, ${}^0\hat{\mathbf{T}}_1^{2\Delta}$, etc.).

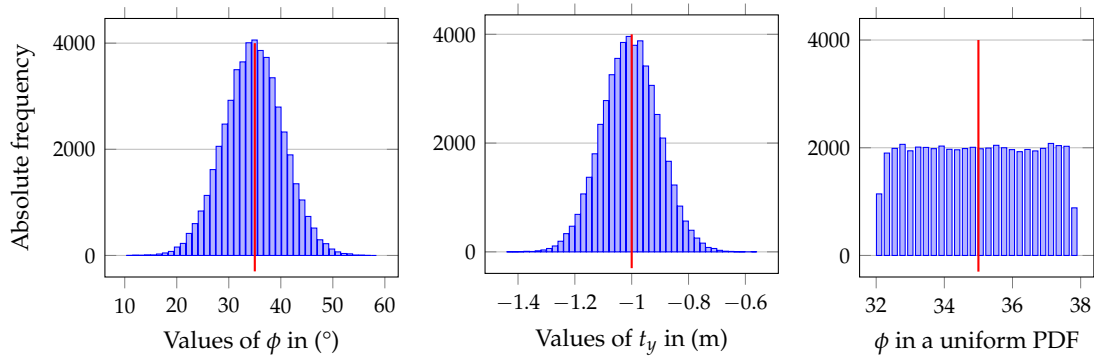


Figure 6. Histogram of variables ϕ and t_y , with mean value marked, and respectively $\sigma_\phi = 4.8^\circ$ and $\sigma_{t_y} = 0.08$ m when the original Gaussian uncertainty is 5.7° and 0.1 m. The third histogram on the right shows the final PDF of ϕ when the uncertainty is uniform, but now with $\sigma_\phi = 1.36^\circ$ (Cf. Table 6).

420 The wealth of data generated by the MCS allows the analysis of the properties of this technique
 421 that is necessary for the multi pairwise based calibration. Two main issues must be observed: how the
 422 mean value of each variable is preserved amongst the several magnitudes of uncertainty, and how
 423 uncertainty (the standard deviation around the mean) is propagated on each transformation path.
 424 Finally, and following the central purpose of this multi pairwise technique, what are the gains in terms
 425 or error propagation when combining (averaging) the results of all the transformation paths involved.

426 Table 2 shows the mean values of the six variables (orientations and translations) at the end of the
 427 five transformation paths, along with their mean values, including also the real ground truth values
 428 for easier comparison. This case was for an input uncertainty of 0.1 from a Gaussian PDF.

Table 2. Mean values of the six variables for the transformation paths using a Gaussian distribution of errors on all variables with an initial uncertainty of 5.7° for angles and 0.1 m for translations.

	Real value	${}^0\hat{\mathbf{T}}_1$	${}^0\hat{\mathbf{T}}_1^1$	${}^0\hat{\mathbf{T}}_1^2$	${}^0\hat{\mathbf{T}}_1^3$	${}^0\hat{\mathbf{T}}_1^4$	Overall mean
$\bar{\phi}$ ($^\circ$)	35	35.02	34.96	35.03	34.99	35.03	35.01
$\bar{\theta}$ ($^\circ$)	0	0.01	0.03	0.04	-0.08	-0.03	-0.01
$\bar{\psi}$ ($^\circ$)	0	-0.02	0.05	-0.03	-0.02	0.04	0.00
\bar{i}_x (m)	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
\bar{i}_y (m)	-1	-1.00	-0.96	-0.98	-0.94	-0.94	-0.96
\bar{i}_z (m)	0.25	0.25	0.25	0.24	0.25	0.25	0.25

429 It can be seen that practically all transformation paths originate mean values very close to the real
 430 value in spite of all the uncertainties of the operations. Nonetheless, this performance degrades for
 431 higher values of uncertainty as detailed ahead.

The other important issue is the propagated uncertainty on each operation, and how is it compensated by the combination of the multiple results. As stated earlier, the standard deviation is the central measure and it is used to translate the uncertainty. When averaging multiple random variables, the standard deviation of the result (σ_M) is given by (22) where for this case five variables, and their individual standard deviations, are used:

$$\sigma_M = \frac{\sqrt{\sigma_0^2 + \sigma_1^2 + \sigma_2^2 + \sigma_3^2 + \sigma_4^2}}{5}. \quad (22)$$

432 If σ_M , the final mean propagated standard deviation, is smaller than the individual uncertainty of
 433 the simple pairwise transformation, then the process reduces the uncertainty present in that single
 434 geometric transformation. For the same example given in Table 2, Table 3 shows the standard deviation

435 (propagated uncertainty) for each transformation path, along with the mean standard deviation (σ_M)
 436 that results from the average of the five transformation paths.

Table 3. Standard deviations of the several posture variables for the transformation paths assuming a Gaussian distribution of errors on all variables with an initial uncertainty of 5.7° or 0.1 meters.

	${}^0\hat{T}_1$	${}^0\hat{T}_1^1$	${}^0\hat{T}_1^2$	${}^0\hat{T}_1^3$	${}^0\hat{T}_1^4$	σ_M
σ_ϕ (°)	5.723	10.031	9.985	12.978	12.952	4.7718
σ_θ (°)	5.726	9.845	9.822	12.674	12.657	4.6777
σ_ψ (°)	5.734	10.002	10.006	12.968	12.939	4.7692
σ_{t_x} (m)	0.100	0.328	0.223	0.382	0.380	0.1353
σ_{t_y} (m)	0.100	0.178	0.177	0.232	0.236	0.0854
σ_{t_z} (m)	0.100	0.328	0.223	0.381	0.380	0.1351

As Table 3 shows, except for t_x and t_z whose nominal values are too small when compared to the large input uncertainty, thus not relevant here, the combined propagated uncertainty is notoriously smaller than the initial uncertainty, even if the individual propagated uncertainty increases in the majority of the transformation paths. To better analyze and express the results, a concept named gain of propagated uncertainty, or G_{σ} , is created and defined as in (23)

$$G_{\sigma} = \frac{\sigma_I - \sigma_M}{\sigma_I} = 1 - \frac{\sigma_M}{\sigma_I}, \quad (23)$$

437 where σ_M is the average propagated uncertainty and σ_I is the initial uncertainty on the variables (0.1 m
 438 for translations or 5.7° for rotations in the previous example).

439 Table 4 shows more details on the results of the process taking as example one of the angles (ϕ)
 440 and one of the translations (t_y). This table covers results for several input uncertainties (named σ_R for
 441 angles and σ_t for translations) and, along with the mean values for the angle and the translation ($\bar{\phi}$, \bar{t}_y),
 442 shows both the final deviation of the variables ($\Delta_r\phi$ to express the relative deviation for angle ϕ , and
 443 $\Delta_r t_y$ for the relative deviation for translation t_y) and the gain in uncertainty reduction (labeled Error
 444 reduction in the table) showing the gains for ϕ and t_y (respectively G_{σ_ϕ} and $G_{\sigma_{t_y}}$).

Table 4. Summary of uncertainty propagation when using 5 paths with Gaussian PDF for input uncertainties in the MCS analysis. Example for two variables ϕ and t_y . The last two columns in the table show the gains in error reduction by using the multi pairwise approach.

Input uncertainty		Results for ϕ and t_y						Error reduction	
σ_R (°)	σ_t (m)	$\bar{\phi}$ (°)	$\Delta_r\phi$	\bar{t}_y (m)	$\Delta_r t_y$	σ_ϕ (°)	σ_{t_y} (m)	G_{σ_ϕ}	$G_{\sigma_{t_y}}$
0.6	0.01	35.00	0.0%	-1.00	0%	0.47	0.01	17.5%	16.9%
1.1	0.02	35.00	0.0%	-1.00	0%	0.95	0.02	17.5%	16.9%
2.9	0.05	35.01	0.0%	-0.99	1%	2.37	0.04	17.4%	16.4%
5.7	0.1	34.99	0.0%	-0.96	4%	4.77	0.09	16.8%	14.8%
8.6	0.15	34.98	0.1%	-0.92	8%	7.24	0.13	15.8%	12.4%
11.5	0.2	35.00	0.0%	-0.86	16%	9.92	0.18	13.5%	9.5%
14.3	0.25	34.93	0.2%	-0.80	26%	12.95	0.23	9.6%	7.0%
17.2	0.3	34.54	1.3%	-0.72	39%	16.39	0.29	4.7%	4.8%

445 From Table 4 we can conclude that the gains in uncertainty reduction can be as high as 17%
 446 for small uncertainties but reduce gradually when the input uncertainty reaches values beyond 10°
 447 or 0.2 m. Also, the mean values of the variables almost do not degrade, although mean values of
 448 translation variables degrade faster than for angles for larger input errors.

449 Figure 7 plots the gains in uncertainty reduction for the two variables analyzed in detail, and we
 450 can even observe the case where the gains become negative for input uncertainties after the value 0.35
 451 (rad for the angles), making the approach theoretically ineffective in terms of uncertainty reduction.

452 Nonetheless, it is worth mention that the mean values of variables (at least those with nominal values
453 much larger than the input uncertainty) still hold close to the real value.

454 In summary, as long as mean values of variables do not deviate too much and the error reduction is
455 positive and meaningful (possibly around 10% or more), the technique of multi pairwise combinations
456 of geometric transformations is valid and useful.

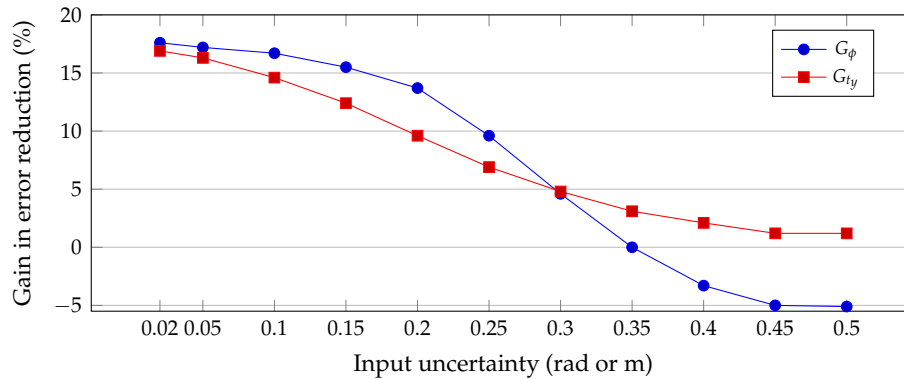


Figure 7. Gains in error reduction of the multi pairwise approach for two variables: one rotation (ϕ) and one translation (t_y). For input uncertainties beyond 0.35 (radians or meters) the gain become loss, and the process is no longer advantageous, ceasing the usefulness of the technique for this setup.

457 This MCS testing methodology has shown great reliability because different repetitions yielded
458 these same tables of results with occasional fluctuations only in some decimal places, despite the
459 random nature of the MCS. Moreover, the tests were performed with other nominal values besides
460 those presented in expression (17) and, as would be expected due to the nature of MCS, the results and
461 limits of the propagated errors are similar for the same number of geometric transformations.

462 For complementary comparison, Table 5 shows something similar to Table 4, but only with
463 transformation paths up to length 2; that is, only three transformation paths to combine.

Table 5. Summary of uncertainty propagation when using 3 paths with Gaussian PDF for input uncertainties. The last two columns in the table show the gains in error reduction by using the multi pairwise approach. The results are poorer than those shown in Table 4.

Input uncertainty		Results for ϕ and t_y						Error reduction	
σ_R (°)	σ_t (m)	$\bar{\phi}$ (°)	$\Delta_r\phi$	\bar{t}_y (m)	$\Delta_r t_y$	σ_{ϕ} (°)	σ_{t_y} (m)	$G_{\sigma_{\phi}}$	$G_{\sigma_{t_y}}$
0.6	0.01	35.00	0.0%	-1.00	0.0%	0.5065	0.0089	11.6%	11.2%
1.1	0.02	35.00	0.0%	-1.00	0.1%	1.0117	0.0177	11.7%	11.3%
2.9	0.05	35.00	0.0%	-1.00	0.5%	2.5270	0.0446	11.8%	10.8%
5.7	0.1	35.00	0.0%	-0.98	2.0%	5.0754	0.0899	11.4%	10.1%
8.6	0.15	34.97	0.1%	-0.96	4.6%	7.6610	0.1366	10.9%	8.9%
11.5	0.2	34.99	0.0%	-0.92	8.3%	10.3880	0.1859	9.3%	7.1%
14.3	0.25	35.09	0.3%	-0.88	13.4%	13.2510	0.2367	7.5%	5.3%
17.2	0.3	34.97	0.1%	-0.84	19.6%	16.4970	0.2898	4.0%	3.4%

464 It can be observed that the results on Table 5, where only three transformation paths are used, and
465 not five as earlier, are poorer, especially in the error reduction gain. From these two last tables it can
466 be concluded that the most interesting solution is to combine more transformation paths, but shorter
467 transformation paths are preferable because they propagate less uncertainty (as Table 3 confirms).

468 The previous observation can be used to corroborate one of the heuristic proposals made earlier
469 on how to pick which transformation paths to combine. So, a good proposal seems to be use more
470 short transformation paths instead of fewer longer transformation paths. For many sensors this would
471 require a further study to find the best trade-off of on many transformation paths of length 3, 4 or more
472 would be useful to add to the list of all transformation paths of length 2.

473 To conclude this analysis, a last table (Table 6) is presented where the PDF of the input uncertainties
 474 is taken as uniform and not Gaussian. Here, five transformation paths were used for the same values
 475 of initial uncertainties as before. The most notorious fact is the large gain in uncertainty reduction
 476 (more than 75 %) for all tested uncertainties. Indeed, uniform distributions keep the error limited and
 477 restrict the propagation. Mean values are well preserved for all variables and standard deviations
 478 kept low (though still larger for translations) but, in the end, showing clear benefits of using the multi
 479 pairwise technique. The only caveat is that, most likely, the sources of error can not be considered
 480 uniform, that is, not strictly limited to an interval, and that is why Gaussian approaches, being more
 481 conservative, appear thus to be safer in performing uncertainty predictions in this context.

Table 6. Summary of uncertainty propagation when using five paths with uniform PDF for input uncertainties in the MCS analysis. The last columns in the table show the gains in error reduction by using the multi pairwise approach.

Input uncertainty		Results for ϕ and t_y						Error reduction	
$\sigma_R(^{\circ})$	$\sigma_t(\text{m})$	$\bar{\phi}(^{\circ})$	$\Delta_r\phi$	$\bar{t}_y(\text{m})$	$\Delta_r t_y$	$\sigma_{\phi}(^{\circ})$	$\sigma_{t_y}(\text{m})$	$G_{\sigma_{\phi}}$	$G_{\sigma_{t_y}}$
0.6	0.01	35.00	0.00%	-1.00	0.01%	0.137	0.002	76.2%	76.1%
1.1	0.02	35.00	0.00%	-1.00	0.01%	0.273	0.005	76.2%	76.0%
2.9	0.05	35.00	0.00%	-1.00	0.08%	0.682	0.012	76.2%	76.0%
5.7	0.1	35.00	0.00%	-1.00	0.30%	1.363	0.024	76.2%	76.0%
8.6	0.15	34.99	0.02%	-0.99	0.69%	2.047	0.036	76.2%	75.9%
11.5	0.2	35.00	0.00%	-0.99	1.21%	2.739	0.048	76.1%	75.8%
14.3	0.25	35.01	0.03%	-0.98	1.92%	3.429	0.061	76.1%	75.7%
17.2	0.3	34.98	0.07%	-0.97	2.77%	4.123	0.073	76.0%	75.6%
20.1	0.35	35.00	0.01%	-0.96	3.76%	4.814	0.086	76.0%	75.5%
22.9	0.4	35.00	0.00%	-0.95	4.94%	5.528	0.099	75.9%	75.3%
25.8	0.45	34.99	0.03%	-0.94	6.31%	6.230	0.112	75.8%	75.2%
28.6	0.5	35.03	0.07%	-0.93	7.89%	6.976	0.125	75.7%	75.0%

482 In conclusion, this section demonstrated the viability and advantage of the multi pairwise
 483 technique to perform sensor calibration within some conditions and limits. For a setup with four
 484 sensors, the technique is advantageous and viable if initial angular uncertainties are less than about 15°
 485 and translation uncertainties less than about 0.15 m. If these restrictions are ensured then, statistically,
 486 and assuming Gaussian uncertainties, the multi pairwise approach improves the accuracy of the
 487 calibration among multiple sensors.

488 In summary, the operation of "averaging" geometric transformations can reduce the error present
 489 in the relative position of sensors. Nonetheless, although demanding and perhaps too conservative,
 490 the results of this theoretical approach are harder to compare with those from experiments using real
 491 data because ground truth is usually not available or not very precise. Hence, in real data experiments,
 492 either there is an estimate of the ground truth of the sensor relative placements, or some alternative
 493 metric has to be used, such as the mean square deviations of 3D points recalculated with the estimated
 494 geometric transformations. Next section, dedicated to results, describes these issues in detail.

495 5. Results

496 The previous section presented an analysis of uncertainty propagation using a case study for
 497 more clarity, but demonstrated that, in some conditions, the multi pairwise approach reduces the
 498 uncertainty in calculating the geometric transformations, by analyzing the propagated uncertainty in
 499 final orientations and translations of the several sensors relative localization, which is the essence of
 500 extrinsic calibration.

501 This section provides an approach using sets of points (also named point clouds throughout this
 502 paper) to extract the single pairwise transformations and obtain the multi pairwise transformation and
 503 compare performance in two perspectives.

504 Indeed, this procedure to obtain results hinders a subtle, but determinant, change since the
 505 geometric transformations do not exist as before in section 4, but are extracted after the point clouds,
 506 which will lead to interesting results as discussed further. Experiments were done both with real data
 507 from sensors and simulated point clouds.

508 The experiment used four different sensors. The principle is independent of the nature of the
 509 sensor, as long as it can detect the center of the ball, so four different types of cameras were used: RGB,
 510 monochromatic, infrared and Kinect-based (for depth maps).

511 An example of simple pairwise based calibration has been previously used for the first three
 512 cameras [9]. In the current work, we add a depth sensor to enrich the initial setup. Indeed, depth maps
 513 are obtained and, being 2D images, ball positions are detected with a Deep Learning trained network
 514 used for RGB, mono and IR images, which are all native, or converted to, simple grayscale intensity
 515 images.

516 The detection of the ball center on all sensors is performed using Detectron2 deep learning
 517 techniques [38] to detect a ball in successive frames. Figure 8 shows an example for the calibration
 518 procedure with three point clouds: one obtained from the RGB camera (blue), another from the Mono
 519 camera (green), and the third (red), which is the calibrated version of the green relative to the RGB
 520 camera.

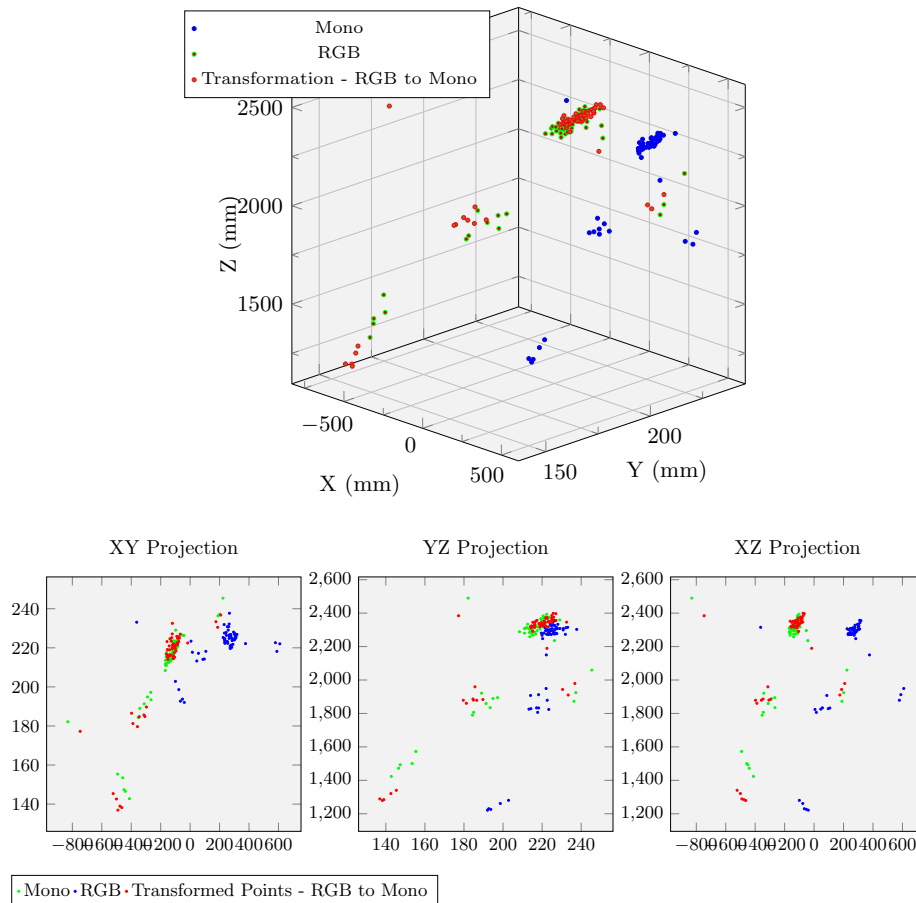


Figure 8. Illustration of the results of calibrating a RGB and a Mono cameras. Through the registration procedure, green points are transformed into red points which match closely the blue point cloud.

521 In the experiments carried out, the four cameras were placed in several positions ensuring a
 522 common field of view favorable to capture the ball during its motion. In each frame, the center of
 523 the ball is detected and saved in a point cloud of all the viewed ball centers for each of the 4 cameras.
 524 This center can be calculated in the coordinate frame of the camera, in meters, using optics and lenses

525 equations (intrinsic matrix). By using the methods presented earlier for matching point clouds, the
 526 pairwise transformations between the sensors are estimated, opening the way to the application of the
 527 multi pairwise method proposed in this paper.

528 Contrarily to simulation, no ground truth is available for the experimental data. Hence, the
 529 metrics to assess the performance of the method is based on the deviation between the point cloud
 530 from the reference sensor and the point cloud from another sensor after the calibration process. For the
 531 metric based on distance deviations, we use the relative deviation (in percentage) of the root mean
 532 square (RMS) of absolute errors which are measured in mm.

533 Experiments have shown that outliers occur frequently in the ball detection phase. That can have
 534 several reasons and depends on the type of sensor used. The deep learning detection algorithm shows
 535 most of the times very high levels of confidence (98%), but sometimes they are lower. The resolution of
 536 some cameras (namely the infrared) is much lower than the other sensors, creating uncertainties in
 537 obtaining the center point of the ball; for some sensors, it is harder to obtain precise intrinsic parameters
 538 and, again, the infrared camera is the strongest example. Additionally, illumination conditions are not
 539 always ideal and that too can affect precise detection. All these issues may occur in smaller or larger
 540 extents and can generate errors in the ball center detection and outliers.

These outliers can be detected by evaluating the distance between the reference point cloud
 and the transformed point clouds of the sensors (using the pairwise transformation). Figure 9 (left)
 illustrates the situation for the case of two sensors where several samples (after sample number 40)
 exhibit a large disparity when compared to the remainder. The error displayed in the plot is calculated
 for each point P_i of point cloud from sensor 0 (reference) to respective point Q_i of point cloud from
 sensor m , after calibration, in the following way:

$$e_i = \frac{\|P_i - {}^0\hat{T}_m Q_i\|}{\|P_i\|} \quad (24)$$

541 Clearly, some of the points illustrated are outliers and result possibly from a poor detection of the
 542 target in one, or both, sensors. Those original samples that do not match a given criteria have to be
 543 removed from both point clouds to keep the correspondence for the pairwise calculation.

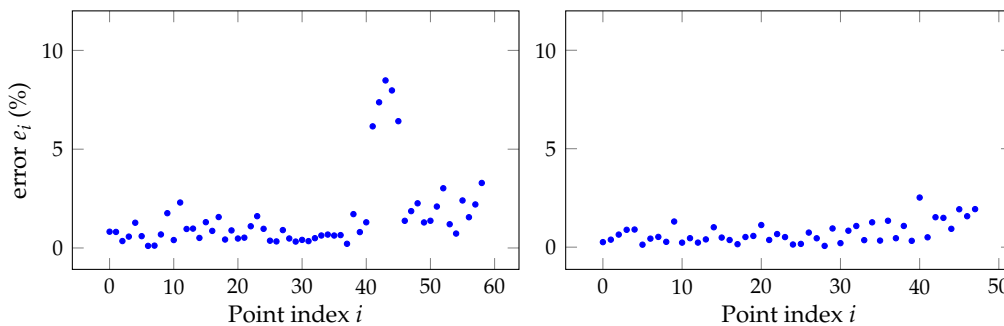


Figure 9. Percentage of error between each point of one point cloud with another point cloud before (left) and after (right) the application of Chauvenet criterion for outliers removal.

544 The process selected to eliminate outliers is the Chauvenet criterion, first devised by William
 545 Chauvenet in 1863 [39], but broadly applied and documented in many sources [40]. This technique
 546 identifies samples that fail to fit a normal distribution with a given probability based on their deviation
 547 from the mean using also the standard deviation for the calculation. The samples that fail the test are
 548 considered outliers and the pairwise calculation of transformations is recomputed with the "clean" set
 549 of samples.

550 The Chauvenet test can be applied recursively until no outliers remain. However, if the
 551 distribution is not Gaussian (or unknown) the process may exaggerate the pruning of the data set
 552 and can remove too many samples. Hence, in this work, two iterations of the criterion are applied to
 553 ensure that potential residual outliers from the first iteration are removed. With the point clouds freed

554 from the outliers (as in the right side of Figure 9), the calculation of calibration matrices, both single
 555 and multi pairwise, can be performed with more confidence.

556 To test and analyze the technique proposed, simple experimental setups with the four sensors
 were used as the one illustrated in Figure 10.

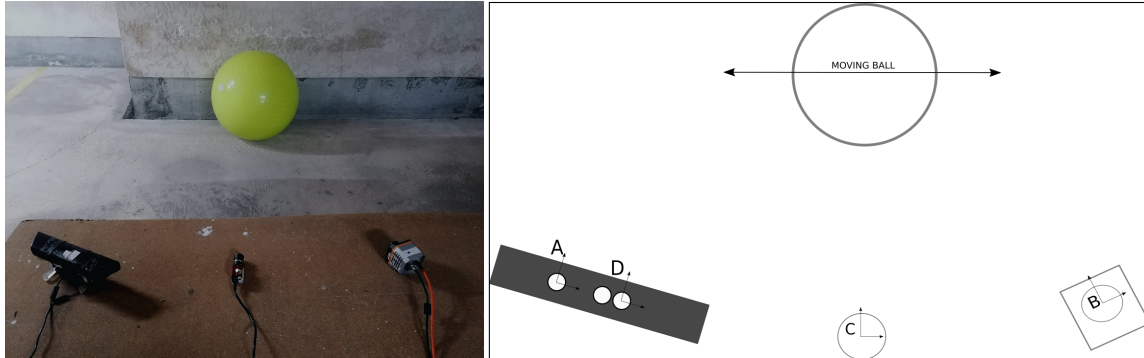


Figure 10. Example of setup to collect data with the sensors placed in variate positions and orientations. A is the depth camera, B the IR camera, C the monochromatic camera and D the RGB camera.

557

558

559

560

561

562

Similarly to the setup in Figure 10, other data collections were gathered using other setups and sensor arrangements. For example, the setup shown in Figure 11 generated the point clouds illustrated on the right, where clearly some of them are defective, compromising the proper illustration of the concepts being proposed in this paper. This may have occurred both for the reasons described earlier and for other unexpected factors that restricted the data acquisition procedures.

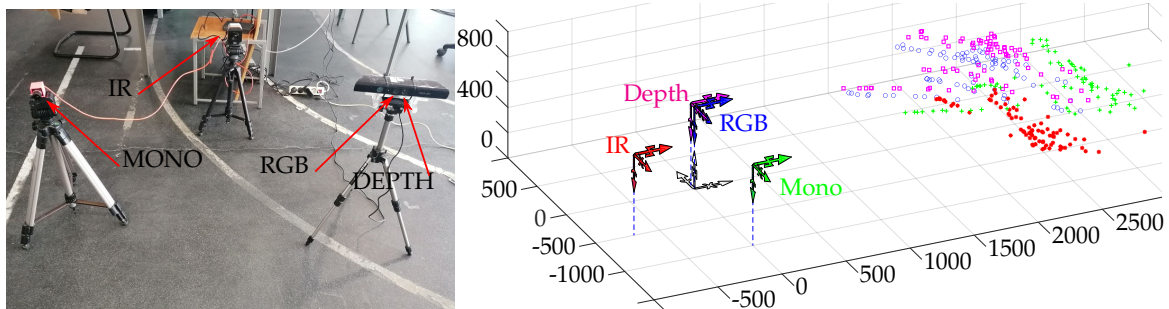


Figure 11. Example of another setup to collect data from sensors. On the right an actual collection of point cloud that shows serious defects making them mostly unusable for calibration. Units are in mm.

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

To overcome the restrictions encountered in the particular experimental data collection, as the one shown in Figure 11, it was also decided to synthesize point clouds on a similar configuration and add noise in all coordinates as an attempt to emulate real point clouds. Although these points and the results thereof are not from real sensors, they hopefully follow the pattern of a real setup, and the noise emulates some of the acquisition errors. The results were analyzed using two different metrics: one based on the relative value of the root mean square of the distance between point clouds (from the reference sensor and from the calibrated sensor), as given by expression (24), and a second one based on the deviation of the calibration matrices, in a line similar to the study performed in section 4, and deviation in translations and Euler angles are assessed. The first metric can be applied without having any ground truth (compare the results with single versus multi pairwise) and the second one requires a ground truth to compare the calibration matrices from single and multi pairwise approaches with the correct value. This second metric can be used in simulated data only because only there a ground truth is available.

Multiple experiments were done with simulated point clouds, both encompassing systematic and random errors as the one shown in Figure 12 where nominal points follow a grid-like layout for easier visual tracking, but other arrangements, even fully random, were tested.

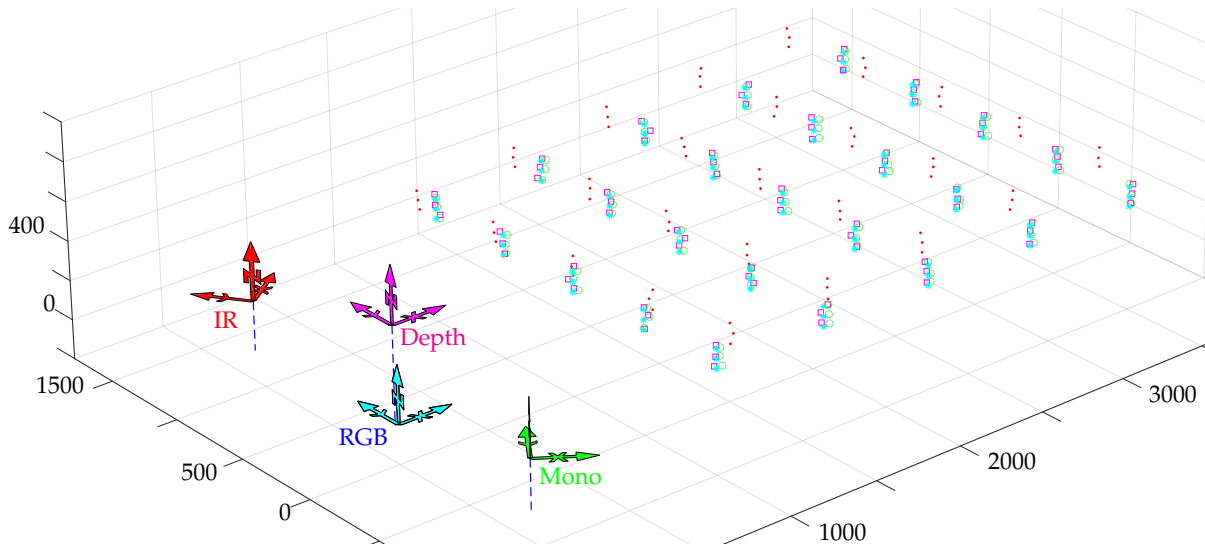


Figure 12. Example of synthetic point clouds with systematic and random errors in a sensor arrangement similar to the one in ATLASCAR2. Units are in mm.

579 However, almost all results have shown little or no improvement at all of the direct application of
 580 the multi pairwise (MPW) matrices when compared to the single pairwise (SPW). Figure 13 shows the
 581 detailed point-to-point relative error for the pairs mono to depth, mono to RGB and mono to IR for
 some other simulated point clouds, and differences between methods are barely noticeable.

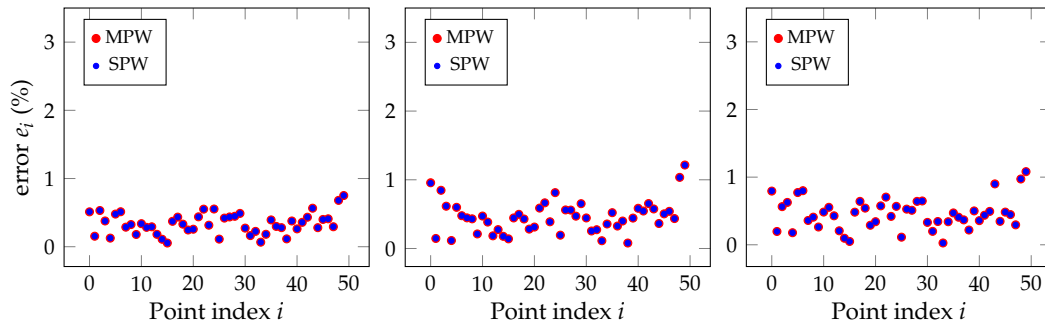


Figure 13. Relative distance errors between every pair of points for the SPW and MPW approaches after removing the outliers with the Chauvenet method for an input error of about 10 mm in the point clouds. Both techniques give practically the same results, as shown by the clear overlap of plots.

582 An explanation for what was observed in simulated data is given next. In ideal conditions, the
 583 SPW and MPW approaches give the same results for multi sensor calibration, making it unnecessary
 584 to use the later. Additionally, as verified, when using synthetic point clouds, the advantages of MPW
 585 have shown to be too little to be useful. Indeed, when comparing with the study made in section 4, now
 586 the geometric transformations are not all truly independent since they are obtained after pairs of point
 587 clouds, hence, the same point cloud is used multiple times for different geometric transformations.
 588 These effects are even stronger in simulated data, and that is why the experiments were redirected into
 589 another point of view. The validity of transformations using MPW still stands but, as it is shown next,
 590 it will be used in a indirect approach.

592 Since the simulated point clouds were not as rich as real data, the experiment presented earlier
 593 from the setup in Figure 10 was chosen and analyzed from another perspective. Both SPW and MPW
 594 approaches were tried and, as shown in Figure 14, and also in the Chauvenet polished version in
 595 Figure 15, there are points in the real point clouds for which the MPW performs better than for the
 596 SPW, and vice versa. In average, for the entire point cloud, the MPW performs similarly to the SPW,
 597 but individually in each point it performs either better or worse, which is a hint to exploit further.

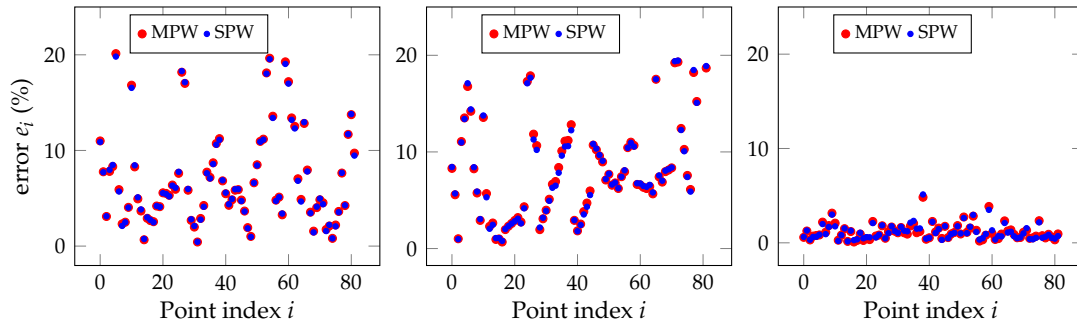


Figure 14. RMS errors for SPW and MPW for the three sensors relatively to the reference sensor (RGB camera) for the raw point clouds with 82 points obtained in the setup from Figure 10.

598 Figure 15 shows the individual point errors after applying the Chauvenet criterion to eliminate
 599 outliers on the raw point clouds with the results shown in Figure 14.

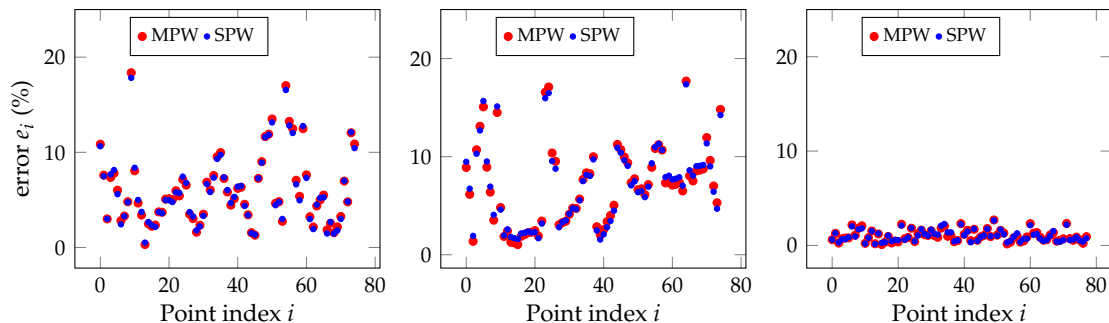


Figure 15. RMS errors for SPW and MPW for the three sensors relatively to reference sensor for the point clouds whose results are shown in Figure 14 but where outliers have been removed by Chauvenet criterion. Number of removed outliers varies with the combination of sensors.

600 After observing the results in Figure 14 or Figure 15, the hypothesis proposed is that poorer
 601 performance in the MPW in some points can represent situations of points that were captured with
 602 larger errors and compromised the point clouds and the geometric transformation generated after
 603 them. So, using the real data, a series of experiments was carried out in order to improve the quality of
 604 the point clouds using the MPW approach. The workflow is the following for four sensors:

- 605 • Obtain the three SPW transformations after the four point clouds.
- 606 • Obtain the associated MPW transformations using all SPW and all the other inter point cloud
 607 transformations.
- 608 • Apply the Chauvenet criterion to eliminate points by analyzing the recalculated error using the
 609 SPW with expression (24).
- 610 • Recalculate SPW and MPW matrices for the three sensors relatively to the reference sensor.
- 611 • Eliminate from the point clouds all points that have a larger error when using the MPW matrix.
- 612 • With the new filtered point cloud, recalculate the SPW matrix and use it as final value for the
 613 geometric transformation w.r.t. the reference sensor.

614 The procedure just described was applied and produced the plots of Figure 16 and results
 615 summarized in Table 7.

616 From Table 7, it is clear that the process of removing outliers decreases the RMS error along with
 617 its standard deviation, which is equivalent to state that the geometric transformations (calibration
 618 matrices) are more accurate than the ones calculated with the raw point clouds. The reduction of also
 619 the standard deviation reinforces the fact that more accurate point clouds are obtained in this process.

620 Chauvenet criterion gives a first step in discarding outliers but the analysis of the MPW for each
 621 point allows the elimination of additional selected points, hence giving smaller mean errors. In the

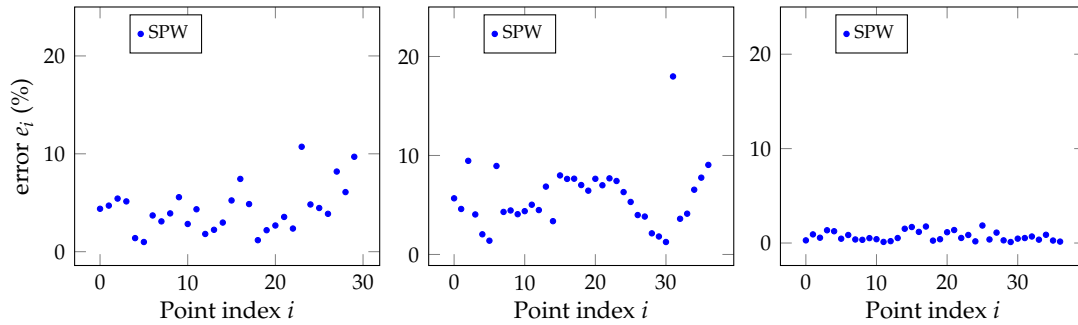


Figure 16. RMS errors for the three sensors relatively to reference sensor for the point clouds whose results are shown in Figure 15 but with additional points removed by observing the MPW versus the SPW results. The observable conclusion when comparing to previous figures is that additional outliers have been removed even after the application of the Chauvenet criterion.

end, as shown in the last column of Table 7 there is a substantial reduction of the RMS error associated to the calibration matrix for each sensor relatively to the reference sensor. In the particular case of sensor 3, the mean error was already small because sensor 3 and the reference sensor lay attached on the same physical structure, but even there occurs an improvement on the point cloud, hence a better calibration matrix is obtained.

The results just presented demonstrate the usefulness of the MPW technique to assist the improvement of the point clouds in order to obtain better transformation matrices for the multi sensor extrinsic calibration process.

Table 7. Summary of results for the RMS relative error and its associated standard deviation between each sensor and the reference sensor upon outlier removal with Chauvenet criterion and using MPW. Errors are in percent values as well as their respective standard deviations (stdev). The last column shows the reduction factor (final value/initial value) of mean errors and standard deviations from original raw point cloud to the final point cloud purged from outliers. Overall mean errors and standard deviations for SPW and MPW are similar so only one of them is shown.

Target Sensor		Raw point cloud	Chauvenet removal	MPW based removal	Reduction factors
1	Total points	82	75	29	
	Mean error	7.0	5.4	4.3	61.4%
	Stdev	4.9	3.6	2.3	46.9%
2	Total points	82	75	36	
	Mean error	8.5	6.5	5.8	68.2%
	Stdev	6.0	4.0	3.0	50.0%
3	Total points	82	78	36	
	Mean error	1.2	1.0	0.7	58.3%
	Stdev	0.8	0.6	0.5	62.5%

6. Conclusions

This paper proposes a methodology to perform semi automatic calibration of several sensors. The requirement is to detect, simultaneously in all the sensors, corresponding 3D points in space, for example, by detecting the center of a moving ball in their field of view. The solution builds upon a technique proposed in previous works [1,9], by combining multiple geometric transformations in several transformation paths to refine or complement the computation of extrinsic parameters of all sensors w.r.t. one sensor defined as the reference. We refer to this approach as a multi pairwise

637 approach for sensor extrinsic calibration, since, instead of considering only transformations between
638 pairs of sensors (as is the case of many previous works), this technique takes into account all sensors
639 to establish the geometric relation between each pair. The technique also proposes alternatives to
640 limit the number of transformation paths to use in order to circumvent the potential explosion in the
641 number of paths that can originate from a transformation graph.

642 The technique has some advantages and applications, although not all of them always occurring,
643 depending on the data and nature of sensors. Indeed, there are two different ways of using the MPW.
644 First, if estimations of all geometric transformations between pairs of sensors exist and are obtained
645 independently of each other, even if under some conditions, the MPW gives a solution with less error
646 than SPW as was demonstrated in section 4. On the other hand, if point clouds are available instead,
647 and geometric transformations are obtained using matching techniques, the MPW technique allows
648 a refinement of the point clouds by eliminating points that escaped traditional outlier filtering, as
649 demonstrated in the experimental results using the Chauvenet criterion.

650 More specifically, the technique was demonstrated theoretically to reduce the propagation of
651 uncertainty under certain conditions. For a configuration inspired on a real setup with an instrumented
652 car, initial uncertainty of values of up to 0.3 rad or 0.3 m on all angles and translations, are reduced by
653 the MPW approach, which shows its advantage, as was demonstrated with a Monte Carlo Simulation
654 methodology.

655 When point clouds are available, as shown in the results section, and that is probably the most
656 common situation in the context of this work, the geometric transformations are not all independent
657 and the simple replacement of the SPW matrix by the MPW may not result in error reduction. However,
658 the MPW results in individual points can be used to filter out points that exhibit errors that degrade the
659 calculation of the SPW (and the MPW all along). The outlier removal by Chauvenet criterion produces
660 interesting results, but the further elimination of points with the application of the MPW refined the
661 point clouds, allowing a new geometric transformation with even less error.

662 It was also verified that the uncertainty on the calculated calibration matrix is better for a
663 larger number of transformation paths, and preferably using the shortest paths. Indeed, shorter
664 transformations paths propagate less uncertainty and more transformation paths reduce the final
665 uncertainty. In the cases presented using four sensors, five transformation paths exist, one with
666 length 1 (the direct path, the one used for the SPW), two with length 2 and two with length 3. In the
667 context of the approach described in section 4, using all the five transformation paths achieves better
668 results than using only three of them. This effect would be stronger in setups with more sensors and
669 more transformation paths. All this opens a door for future developments by analyzing the several
670 transformations paths available and pick only some of them that fit some criteria to be investigated.

671 The paper focused mainly on the advantages of the multi pairwise approach in two different
672 perspectives and contexts of the available data, but the overall technique using the ball can be
673 developed further. Indeed, the technique presented does not yet solve all the challenges of multi-modal
674 extrinsic calibration. A door is left open to merge with optimization techniques discussed in the related
675 work. Other future developments will continue mainly to overcome limitations inherited from the
676 earlier approach, and not yet tackled in the present work, like the ambiguity in the detection of the ball
677 hemisphere which occurs in some LIDAR sensors. Solutions for those challenges will possibly count
678 with 3D dynamic tracking of the ball put in motion using more elaborate motion patterns.

679 **Author Contributions:** Vitor Santos: conceptualization, theoretical analysis and writing of the original draft.
680 Daniela Rato: experimental tests, data processing and presentation of results; Miguel Oliveira and Paulo Dias:
681 review of the document and contributions on the team's collective research effort on the calibration techniques.

682 **Funding:** This research was partially funded by Fundo Azul, Ref. FA_02_2017_011.

683 **Conflicts of Interest:** The authors declare no conflict of interest.

684 **References**

- 685 1. Pereira, M.; Silva, D.; Santos, V.; Dias, P. Self calibration of multiple LIDARs and cameras on autonomous
686 vehicles. *Robotics and Autonomous Systems* **2016**, *83*, 326 – 337. doi:10.1016/j.robot.2016.05.010.
- 687 2. Zhang, Q.; Pless, R. Extrinsic calibration of a camera and laser range finder (improves camera calibration).
688 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No.
689 04CH37566). IEEE, 2004, Vol. 3, pp. 2301–2306.
- 690 3. Zhang Q., X.C. Deep ChArUco: Dark ChArUco Marker Pose Estimation. *Intelligent Robotics and*
691 *Applications. ICIRA 2017. Lecture Notes in Computer Science*, 2017, Vol. 10463.
- 692 4. Czyzewski, M.A. An Extremely Efficient Chess-board Detection for Non-trivial Photos. *ArXiv* **2017**,
693 *abs/1708.03898*.
- 694 5. Garrido-Jurado, S.; Muñoz-Salinas, R.; Madrid-Cuevas, F.; Medina-Carnicer, R. Generation of fiducial
695 marker dictionaries using Mixed Integer Linear Programming. *Pattern Recognition* **2016**, *51*, 481 – 491.
696 doi:10.1016/j.patcog.2015.09.023.
- 697 6. Romero-Ramirez, F.J.; Muñoz-Salinas, R.; Medina-Carnicer, R. Speeded up detection of squared fiducial
698 markers. *Image and Vision Computing* **2018**, *76*, 38 – 47. doi:10.1016/j.imavis.2018.05.004.
- 699 7. Hu, D.; DeTone, D.; Malisiewicz, T. Deep ChArUco: Dark ChArUco Marker Pose Estimation. 2019
700 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 8428–8436.
- 701 8. Ruan, M.; Huber, D. Calibration of 3D Sensors Using a Spherical Target. 2014 2nd International Conference
702 on 3D Vision, 2014, Vol. 1, pp. 187–193. doi:10.1109/3DV.2014.100.
- 703 9. Rato, D.; Santos, V. Automatic Registration of IR and RGB Cameras using a Target detected with Deep
704 Learning. 2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC),
705 2020, pp. 287–293.
- 706 10. Kwon, Y.C.; Jang, J.W.; Choi, O. Automatic sphere detection for extrinsic calibration of multiple RGBD
707 cameras. 2018 18th Int. Conf. on Control, Automation and Systems (ICCAS), 2018, pp. 1451–1454.
- 708 11. Almeida, M.; Dias, P.; Oliveira, M.; Santos, V. 3D-2D Laser Range Finder Calibration Using a Conic Based
709 Geometry Shape. *Image Analysis and Recognition*, 2012, pp. 312–319.
- 710 12. Mueller, G.R.; Wuensche, H. Continuous stereo camera calibration in urban scenarios. 2017 IEEE 20th Int.
711 Conf. on Intelligent Transportation Systems (ITSC), 2017, pp. 1–6. doi:10.1109/ITSC.2017.8317675.
- 712 13. Wu, L.; Zhu, B. Binocular stereovision camera calibration. 2015 IEEE Int. Conf. on Mechatronics and
713 Automation (ICMA), 2015, pp. 2638–2642. doi:10.1109/ICMA.2015.7237903.
- 714 14. Rou Su.; JingLiang Zhong.; QiaoLiang Li.; SuWen Qi.; HuiSheng Zhang.; TianFu Wang. An
715 automatic calibration system for binocular stereo imaging. 2016 IEEE Advanced Information
716 Management, Communicates, Electronic and Automation Control Conf. (IMCEC), 2016, pp. 896–900.
717 doi:10.1109/IMCEC.2016.7867340.
- 718 15. Ling, Y.; Shen, S. High-precision online markerless stereo extrinsic calibration. 2016 IEEE/RSJ Int. Conf.
719 on Intelligent Robots and Systems (IROS), 2016, pp. 1771–1778. doi:10.1109/IROS.2016.7759283.
- 720 16. Dinh, V.Q.; Nguyen, T.P.; Jeon, J.W. Rectification Using Different Types of Cameras Attached to a Vehicle.
721 *IEEE Trans. on Image Processing* **2019**, *28*, 815–826. doi:10.1109/TIP.2018.2870930.
- 722 17. Khan, A.; Aragon-Camarasa, G.; Sun, L.; Siebert, J.P. On the calibration of active binocular and RGBD
723 vision systems for dual-arm robots. 2016 IEEE Int. Conf. on Robotics and Biomimetics (ROBIO), 2016, pp.
724 1960–1965. doi:10.1109/ROBIO.2016.7866616.
- 725 18. Basso, F.; Menegatti, E.; Pretto, A. Robust Intrinsic and Extrinsic Calibration of RGB-D Cameras. *IEEE*
726 *Trans. on Robotics* **2018**, *34*, 1315–1332. doi:10.1109/TRO.2018.2853742.
- 727 19. Qiao, Y.; Tang, B.; Wang, Y.; Peng, L. A new approach to self-calibration of hand-eye vision systems. 2013
728 Int. Conf. on Computational Problem-Solving (ICCP), 2013, pp. 253–256. doi:10.1109/ICCP.2013.6893596.
- 729 20. Zhang, C.; Zhang, Z. Calibration between depth and color sensors for commodity depth cameras. 2011
730 IEEE Int. Conf. on Multimedia and Expo, 2011, pp. 1–6. doi:10.1109/ICME.2011.6012191.
- 731 21. Chen, G.; Cui, G.; Jin, Z.; Wu, F.; Chen, X. Accurate Intrinsic and Extrinsic Calibration of RGB-D Cameras
732 With GP-Based Depth Correction. *IEEE Sensors Journal* **2019**, *19*, 2685–2694. doi:10.1109/JSEN.2018.2889805.
- 733 22. Vasconcelos, F.; Barreto, J.P.; Nunes, U. A Minimal Solution for the Extrinsic Calibration of a Camera and a
734 Laser-Range-finder. *IEEE Trans. on Pattern Analysis and Machine Intelligence* **2012**, *34*, 2097–2107.

- 735 23. Qilong Zhang.; Pless, R. Extrinsic calibration of a camera and laser range finder (improves camera
736 calibration). 2004 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566),
737 2004, Vol. 3, pp. 2301–2306 vol.3. doi:10.1109/IROS.2004.1389752.
- 738 24. Häselich, M.; Bing, R.; Paulus, D. Calibration of multiple cameras to a 3D laser range finder. 2012 IEEE Int.
739 Conf. on Emerging Signal Processing Applications, 2012, pp. 25–28.
- 740 25. Chen, Z.; Yang, X.; Zhang, C.; Jiang, S. Extrinsic calibration of a laser range finder and a camera based on
741 the automatic detection of line feature. 2016 9th Int. Congress on Image and Signal Processing, BioMedical
742 Engineering and Informatics (CISP-BMEI), 2016, pp. 448–453.
- 743 26. Velas, M.; Spanel, M.; Materna, Z.; Herout, A. Calibration of RGB camera with velodyne LiDAR. Proc. of
744 WSCG2014 Conference on Computer Graphics, Visualization and Computer Vision, 2014.
- 745 27. Guindel, C.; Beltrán, J.; Martín, D.; García, F. Automatic extrinsic calibration for lidar-stereo vehicle sensor
746 setups. 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC) 2017, pp. 1–6.
- 747 28. Lee, G.; Lee, J.; Park, S. Calibration of VLP-16 Lidar and multi-view cameras using a ball for 360 degree 3D
748 color map acquisition. 2017 IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems
749 (MFI), 2017, pp. 64–69. doi:10.1109/MFI.2017.8170408.
- 750 29. Levinson, J.; Thrun, S. Automatic Online Calibration of Cameras and Lasers. Robotics: Science and
751 Systems, 2013.
- 752 30. Dezhi Gao.; Duan, J.; Xining Yang.; Zheng, B. A method of spatial calibration for camera and
753 radar. 2010 8th World Congress on Intelligent Control and Automation, 2010, pp. 6211–6215.
754 doi:10.1109/WCICA.2010.5554411.
- 755 31. Santos, V.; Almeida, J.; Ávila, E.; Gameiro, D.; Oliveira, M.; Pascoal, R.; Sabino, R.; Stein, P. ATLASCAR -
756 technologies for a computer assisted driving system, on board a common automobile. 13th Int. IEEE Conf.
757 on Intelligent Transportation Systems, 2010, pp. 1421–1427. doi:10.1109/ITSC.2010.5625031.
- 758 32. Liao, Y.; Li, G.; Ju, Z.; Liu, H.; Jiang, D. Joint kinect and multiple external cameras simultaneous
759 calibration. 2017 2nd Int. Conf. on Advanced Robotics and Mechatronics (ICARM), 2017, pp. 305–310.
760 doi:10.1109/ICARM.2017.8273179.
- 761 33. Rehder, J.; Siegwart, R.; Furgale, P. A General Approach to Spatiotemporal Calibration in Multisensor
762 Systems. *IEEE Trans. on Robotics* 2016, 32, 383–398. doi:10.1109/TRO.2016.2529645.
- 763 34. Pradeep, V.; Konolige, K.; Berger, E., Calibrating a Multi-arm Multi-sensor Robot: A Bundle Adjustment
764 Approach. In *Experimental Robotics: The 12th Int. Symposium on Experimental Robotics*; Springer Berlin
765 Heidelberg: Berlin, Heidelberg, 2014; pp. 211–225. doi:10.1007/978-3-642-28572-1_15.
- 766 35. Oliveira, M.; Castro, A.; Madeira, T.; Pedrosa, E.; Dias, P.; Santos, V. A ROS framework for the extrinsic
767 calibration of intelligent vehicles: A multi-sensor, multi-modal approach. *Robotics and Autonomous Systems*
768 2020, 131, 103558. doi:10.1016/j.robot.2020.103558.
- 769 36. Arun, K.S.; Huang, T.S.; Blostein, S.D. Least-Squares Fitting of Two 3-D Point Sets. *IEEE Trans. Pattern*
770 *Anal. Mach. Intell.* 1987, 9, 698–700. doi:10.1109/TPAMI.1987.4767965.
- 771 37. Santos, V. ATLASCAR: A Sample of the Quests and Concerns for Autonomous Cars. Informatics in
772 Control, Automation and Robotics; Gusikhin, O.; Madani, K., Eds.; Springer International Publishing:
773 Cham, 2020; pp. 355–375. doi:10.1007/978-3-030-11292-9_18.
- 774 38. Wu, Y.; Kirillov, A.; Massa, F.; Lo, W.Y.; Girshick, R. Detectron2. [https://github.com/facebookresearch/
775 detectron2](https://github.com/facebookresearch/detectron2), 2019.
- 776 39. Chauvenet, W. *Method of least squares. Appendix to Manual of Spherical and Practical Astronomy, Vol. 2*;
777 Lippincott: Philadelphia, 1863.
- 778 40. Barnett, V.; Lewis, T. *Outliers in Statistical Data, 3rd ed.*; John Wiley & Sons: Chichester, UK, 1994.