robotics

MDPI

Article

Tilt Correction of Panoramic Images for a Holistic Visual Homing Method with Planar-Motion Assumption

Christoph Berganski, Annika Hoffmann and Ralf Möller *

Computer Engineering Group, Faculty of Technology, Bielefeld University, 33615 Bielefeld, Germany
* Correspondence: moeller@techfak.uni-bielefeld.de

Abstract: Holistic local visual homing based on warping of panoramic images relies on some simplifying assumptions about the images and the environment to make the problem more tractable. One of these assumptions is that images are captured on flat ground without tilt. While this might be true in some environments, it poses a problem for a wider real-world application of warping. An extension of the warping framework is proposed where tilt-corrected images are used as inputs. The method combines the tilt correction of panoramic images with a systematic search through hypothetical tilt parameters, using an image distance measure produced by warping as the optimization criterion. This method not only improves the homing performance of warping on tilted images, but also allows for a good estimation of the tilt without requiring additional sensors or external image alignment. Experiments on two newly collected tilted panoramic image databases confirm the improved homing performance and the viability of the proposed tilt-estimation scheme. Approximations of the tilt-correction image transformations and multiple direct search strategies for the tilt estimation are evaluated with respect to their runtime vs. estimation quality trade-offs to find a variant of the proposed methods which best fulfills the requirements of practical applications.

Keywords: visual navigation; visual homing; image warping; tilt; tilt correction; tilt estimation

Citation: Berganski, C.; Hoffmann, A.; Möller, R. Tilt Correction of Panoramic Images for a Holistic Visual Homing Method with Planar-Motion Assumption. *Robotics* **2023**, *12*, 20. https://doi.org/10.3390/robotics12010020

Academic Editor: Xinjun Liu

Received: 28 November 2022
Revised: 22 January 2023
Accepted: 24 January 2023
Published: 31 January 2023

1. Introduction

A fundamental building block of visual robot navigation methods is the estimation of relative camera poses from images taken at nearby locations, in the following referred to as *local visual homing*. In this publication, we focus on methods using pairs of panoramic images, e.g., obtained from an upward facing camera equipped with a fisheye lens.

Local visual homing can be coarsely divided into two main branches: feature-based (e.g., [1,2]) and holistic methods (e.g., [3,4]). Feature-based methods detect interest points in images, describe features in the vicinity of the interest points and build a pose estimation pipeline on matched feature descriptors. Holistic methods use the images as a whole at the pixel level without any previous feature extraction. One holistic local visual homing method is MinWarping, first proposed in [3] as a reformulation and extension of 2D-warping [5] which in turn derives from 1D-warping [6]. Given two input images, one taken at the current position (current view) and one at a previously visited position (snapshot), MinWarping computes a compass angle and the angle of a home vector pointing from the current position to the snapshot position. MinWarping includes a systematic search through these two angles and optimizes a pixelwise image distance measure. The computations are based on geometric relationships between hypothetical camera poses and on pixelwise matches between the images under simplifying assumptions. A direct comparison of feature-based methods and MinWarping regarding quality, speed, and robustness can be found in [7]. In this comparison, no method was clearly predominant considering all evaluation criteria. MinWarping produced competitive results while being very fast.

Multiple improvements have been proposed for MinWarping to handle illumination changes. MinWarping in combination with edge-filtered input images and a suitable

distance function reaches good homing performance under varying illumination conditions in indoor environments [8,9]. Additional preprocessing is able to improve the performance in challenging outdoor environments by suppressing misleading cloud edges [10].

Another important issue for visual homing is robustness against the tilt of the camera. MinWarping is built on the assumption of an underlying planar movement. In outdoor environments, this is violated by a slanted or uneven ground; in indoor environments, especially carpet borders or doorsteps lead to camera tilt. These violations of the planar-motion assumption lead to errors in the home-direction estimates (see Figure 1), which increase with the amount of tilt (see [11] Figure 6.4). Tilted images can be corrected by an additional sensor, such as an inertial measurement unit, but this involves higher costs and a further calibration effort and is not pursued here. For indoor environments, an approach for visual tilt estimation was proposed in [12] using vertical image edges and the vanishing point theory. In [13], a visual 3D compass was presented to compute the rotation between two hemispherical images. This compass was based on the concept of real spherical harmonics and an exhaustive search with a coarse-to-fine approach. The main problem of the 3D compass was that the results were only reliable if no or only small translations occurred between the camera positions. However, in many applications, the translational offset between camera poses cannot be neglected. A generalization of 2D-warping to handle nonplanar movement was proposed in [11] and was called 3D-warping. The approach of 3D-warping was related to the 3D compass and was also based on real spherical harmonics. Nonplanar movements were divided into a translational and a rotational part, and warped versions of the current view were computed for a set of movement hypotheses. The best match between the snapshot and all warped current views gave the estimated 3D movement. For nonplanar movement, 3D-warping increased the homing accuracy compared to 2D-warping and MinWarping, especially when computed on skyline-segmented images. However, for planar movements, 2D-warping and MinWarping outperformed 3D-warping.

In this paper, we propose to include tilt estimation in the MinWarping framework. We present a solution to camera-pose estimation between one upright and one tilted view; the more general problem of two tilted views will be explored in future work. Figure 2 visualizes this simplified and the more general case as well. The main approach is to perform a search over two tilt parameters within MinWarping. The potentially tilted view is transformed according to a set of hypothetical tilt parameters and matched to the upright view by using MinWarping as the objective function. A search for the best match is performed. Different search strategies are implemented and evaluated with respect to speed and homing performance. At the moment, the tilt angles are restricted to small values, as they are typical in indoor settings. The proposed extension of MinWarping improves the camera-pose estimation, but is also relevant in other applications, e.g., when MinWarping is used for loop-closure detection [14], where invariance to the tilt is also likely to be beneficial.

The paper is structured as follows: In Section 2, the coordinate systems and parametrization for a small out-of-plane tilt are introduced. Using these, solutions for transforming panoramic images for tilt correction are derived based on previous work [15]. These are combined with warping as an objective function via one of three proposed direct search methods to optimize for the ideal tilt estimation. To test the proposed methods, two newly collected indoor panoramic image databases are presented in Section 3, comprising images with a systematic tilt applied to the camera. In Sections 4 and 5, the homing, tilt-correction, and runtime performance of the proposed methods are evaluated and discussed in the context of practical applicability with suggestions for further improvements or open research questions. Finally, a conclusion is drawn in Section 6.

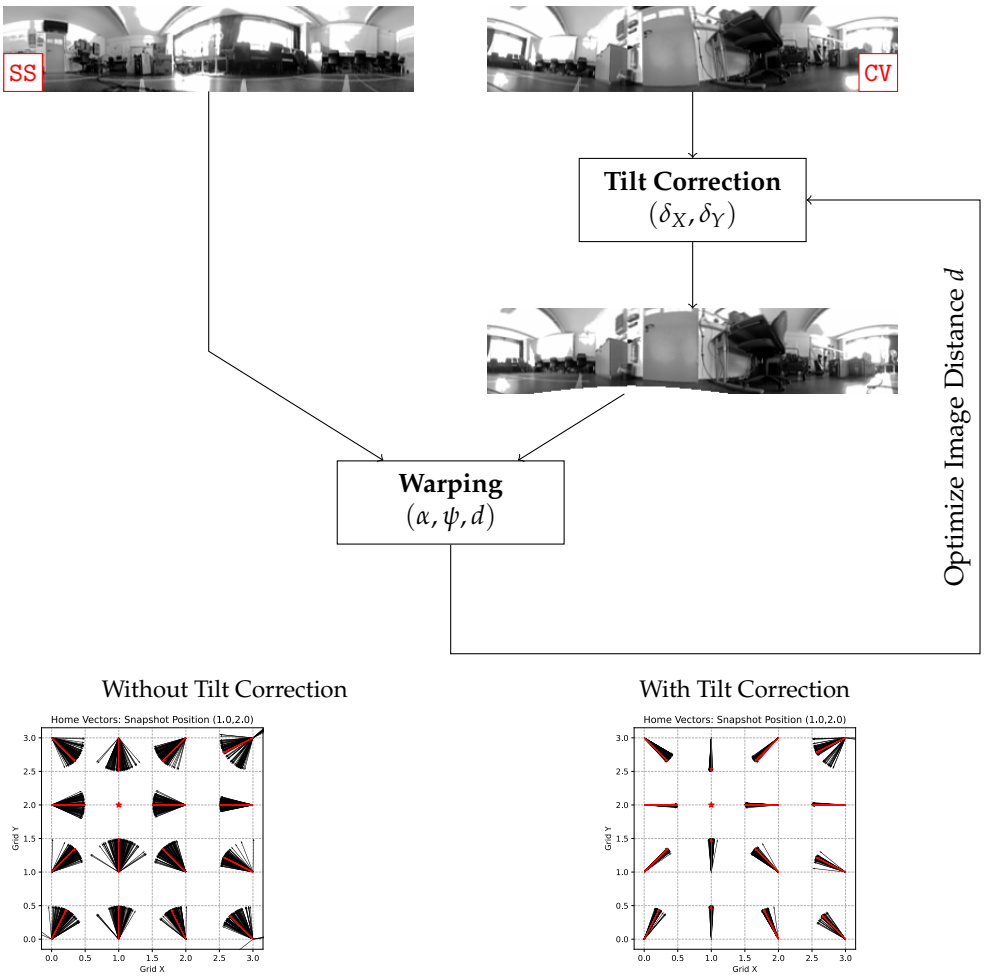
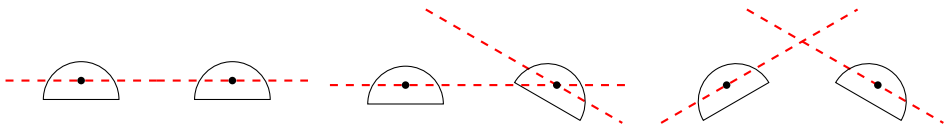


Figure 1. Overview and exemplary results of the proposed tilt-correction warping methods; see the text for a detailed explanation of the algorithm. The vector-field plots below visualize the home-direction estimates by warping at each position over the different tilts in the database, in case of the original warping algorithm without tilt correction (**left**) and of the proposed extension incorporating a search for the tilt correction of the current view image. Warping with tilt-corrected images (**right**) shows a markedly reduced spread of the direction estimates. The sample without tilt correction is produced from the `n--@lab-dd` experiment. The sample with tilt correction is produced from the `pen@lab-dd` experiment, showing the results of one of the heuristic search strategies, pattern search.



(**a**) Planar Motion (**b**) One Camera Tilted (**c**) Both Cameras Tilted

Figure 2. Progressive violations of the planar-motion assumption. The panoramic camera is visualized by the outermost lens as it appears in an ultrawide-angle fisheye lens system. (**a**) depicts the planar-motion assumption underlying the original warping method. None of the cameras is tilted and the horizontal plane (red, dashed line) of each intersects with the optical center (black dots) of the other, i.e., both horizontal planes are identical. (**b**) depicts a first violation of this planar-motion assumption by tilting one of the cameras out of this shared horizontal plane. The optical center of this tilted camera

still lies within the horizontal plane of the not-tilted camera, but not vice versa. This intermediate case is the one handled by our newly proposed method, when tilt correction is applied to the view of the camera depicted on the right side. Finally, (**c**) depicts the more general case of both cameras tilted out of a shared plane connecting the optical centers. The horizontal planes now intersect outside of both of the cameras. This more general case is left for future work and might also extend to or even require out-of-plane translations (see [11] for the case of the generalized 3D-warping).

2. Methods

Figure 1 provides an overview of the proposed extension of the warping algorithm by tilt-correction methods and gives an impression of the improvement of the home-direction estimates achieved by including a tilt correction. Given are two input panoramic images: the snapshot (ss) and the potentially tilted current view (cv). The current view undergoes a tilt-correcting transformation as derived in Section 2.3. This explicit transformation of the image according to hypothetical movement parameters follows the same general idea as the original warping method. The transformations account for a small tilt of the camera, which leads to deviations from the planar motion assumption underlying the warping algorithm. The movement parameters are the two tilt parameter angles δ_X and δ_Y which are introduced in Section 2.1 together with their corresponding coordinate systems.

From the snapshot image and the tilt-corrected current view, the warping algorithm estimates the relative movement parameters α and ψ between the two image locations and an image distance measure d. This image distance d is used in turn as an optimization criterion (as motivated in Section 2.2) to search for the best-matching tilt correction using one of the direct, gradient-free search strategies proposed in Section 2.4. This optimization scheme requires multiple evaluations of the original warping method (for each transformed current view) to estimate the tilt correction; this is indicated in Figure 1 by the looping connection from the *warping* block back to the *tilt correction* block. This connection is where the proposed search strategies are implemented on a search space spanned by the two tilt parameter angles.

While the primary intention of this extension is to improve the homing performance of the warping method in case of small out-of-plane tilts of one of the cameras (as visualized in Figure 1 in the vector-field plots), the estimation of the tilt parameter angles is a first step towards a full three-dimensional, relative pose estimation between two panoramic images.

2.1. Tilt-Angle Representations

For describing the tilt of a robot with respect to the horizontal reference plane, i.e., flat ground, two representations are proposed: axis-angle and roll-pitch representations. In both cases, the tilt is assumed to be a small rotation out of the horizontal plane without affecting the view direction of the robot. The initial (not-tilted) coordinate system (X, Y, Z) is mounted on the robot such that the X- and Y-axis span the horizontal plane, where the X-axis is pointing in the forward direction of the robot and the Y-axis is pointing to the left. The Z-axis is pointing upwards. A second coordinate system (X', Y', Z') is then obtained by tilting the first one out of the horizontal plane. To be fully specified, the tilting out of a 2D plane requires two parameters as this is a 2-degree-of-freedom (DoF) subset of the general 3-DoF problem of describing arbitrary 3D orientations, which would also include changes in the view direction of the robot. In case of the roll-pitch representation, these two parameters are given by consecutive rotations about the two coordinate axes X and Y spanning the reference plane. This can be expressed by multiplying the corresponding 3D basic rotation matrices:

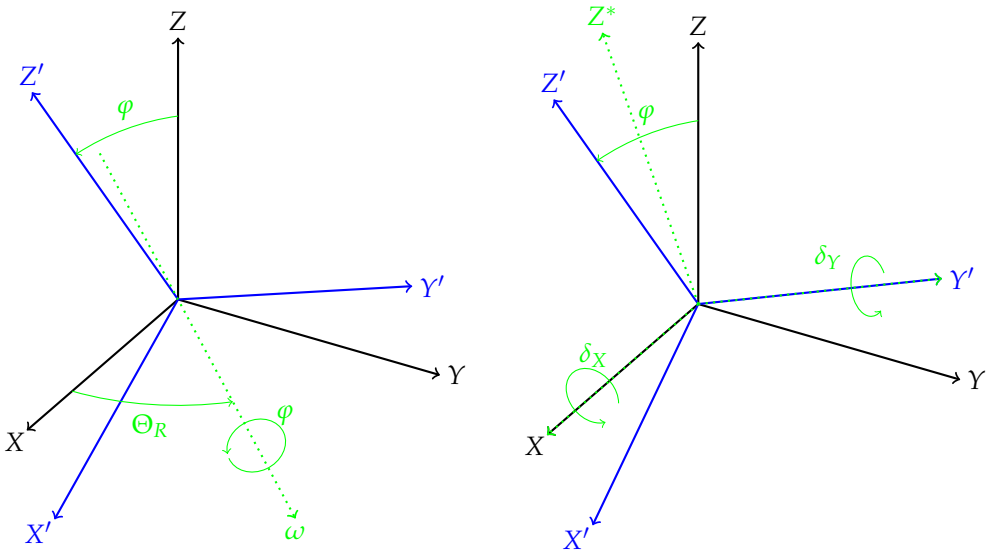
$$\mathrm{R}_X(\delta_X)\mathrm{R}_Y(\delta_Y) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_{\delta_X} & -s_{\delta_X} \\ 0 & s_{\delta_X} & c_{\delta_X} \end{bmatrix} \begin{bmatrix} c_{\delta_Y} & 0 & s_{\delta_Y} \\ 0 & 1 & 0 \\ -s_{\delta_Y} & 0 & c_{\delta_Y} \end{bmatrix} \tag{1}$$

Here and in the following, s_* is short for $\sin(*)$, and c_* is short for $\cos(*)$. The two parameters (δ_X, δ_Y) are the tilt angles about their corresponding axis, together controlling

the direction and total amount of tilt. An axis-angle representation can be constructed by specifying only a single rotation by an angle φ about a tilt axis ω which remains fixed. Here, this tilt axis can be specified by only one parameter Θ_R which places it within the reference plane relative to the X-axis:

$$\Theta_R = \angle(\omega, X) \qquad \text{where } \omega = \mathrm{R}_Z(\Theta_R)X \tag{2}$$

Thus, the parameters Θ_R and φ of the axis-angle representation independently control the direction Θ_R of the tilt axis ω and the total amount of tilt φ. The layout of the coordinate systems and the effect of the tilt according to both representations is depicted in Figure 3.



(**a**) Axis-Angle Representation (**b**) Roll-Pitch Representation

Figure 3. Depiction of a coordinate system tilted according to the two proposed representations. The axis-angle representation (**a**) describes the tilt as a single rotation by a small angle φ about the tilt axis ω, which lies within the reference plane of the nontilted coordinate system and is described relative to the X-axis by an angle Θ_R. This angle does not correspond to a real rotation of the robot as it just specifies the axis of rotation. Furthermore, the angle Θ_R typically is *not* small, i.e., it can range from 0° to 360° as the robot may tilt in any direction. The roll-pitch representation (**b**) describes the tilt as two consecutive small rotations δ_X and δ_Y about the X- and Y-axis which span the reference plane of the nontilted coordinate system. As two rotations generally do not commute, the order of these two matters: the X-axis rotation by δ_X is always carried out first, i.e., the Y-axis rotation δ_Y is about the already tilted axis Y'. The green, dotted coordinate system with the Z^* axis indicates this intermediate step where only the X-axis rotation is performed. In both representations, the total amount of tilt is given by the angle between the nontilted and tilted Z-axis, which is pointing upwards: $\varphi = \angle(Z, Z')$

As both representations are useful for different parts of the tilt-correction process described in the following, conversions between the two parameter sets are derived under the assumption of a small tilt, i.e., δ_X, δ_Y, and φ are all close to 0. This approximation is required since the equivalence of roll-pitch and axis-angle representations can only exactly be established—according to Euler's rotation theorem [16]—if *three* angles are used for both representations.

The first set of conversions determines the tilt direction Θ_R and the amount of tilt φ of the axis-angle representation from the rotations about the X- and Y-axis of the roll-pitch representation:

$$\Theta_R = \operatorname{atan2}(\delta_Y, \delta_X) \tag{3}$$

$$\varphi = \arccos(\cos(\delta_X)\cos(\delta_Y)) \tag{4}$$

Equation (4) can also be interpreted as measuring the magnitude of the tilt of the roll-pitch representation, which is useful, for example, when analyzing results with respect to the amount of tilt irrespective of the direction in which the robot is tilted.

The other set of conversions computes the opposite direction, i.e., determines the tilt angle per axis δ_X and δ_Y from the tilt direction and the total amount of tilt given by the axis-angle representation:

$$\delta_X = \varphi\cos(\Theta_R) \tag{5}$$

$$\delta_Y = \varphi\sin(\Theta_R) \tag{6}$$

This can be thought of as projecting the shared amount of tilt φ to its X- and Y-axis components. Both conversion directions are derived in the Appendix A.1.

2.2. Outline of the Tilt-Correction Warping Algorithm

To give an outline of the tilt-correction extension of warping (that means MinWarping according to [3]), we ignore the inner details and interpret warping as a function mapping two images—the snapshot *ss* (goal location) and the current view *cv*—to the warp parameters α, ψ, and the corresponding image distance d:

$$(\alpha, \psi, d) = \text{warping}(\text{ss}, \text{cv}) \tag{7}$$

The image distance d gives a measure of how well the two images match after applying the transformation by α and ψ. It is assumed that the smaller the distance d warping yields, the better the ss-cv image pair matches. This means it is easier to warp one image into the other for a smaller image distance d. Furthermore, it is assumed that a smaller d coincides with better matching estimates of the movement parameters α and ψ.

This interpretation of the image distance d allows warping itself to be used as a distance measure/criterion for other optimization algorithms: given a transformation for tilt correction of an image $\text{tilt}_\delta(\text{cv})$ with some parametrization of the tilt δ, it is now possible to use the interpretation of warping as a quality measure for image matching to optimize this δ with respect to the distance measure d. This is done by running warping multiple times while applying a systematic tilt distortion to one image and selecting the result with the minimal distance measure d:

$$\hat{\delta} = \underset{\delta}{\text{argmin}}\{d : (\alpha, \psi, d) = \text{warping}(\text{ss}, \text{tilt}_\delta(\text{cv}))\} \tag{8}$$

The best set of warp parameters $(\hat{\alpha}, \hat{\psi}, \hat{d})$ is then obtained by warping using the tilt transformation with the best set of tilt parameters $\hat{\delta}$ found by the optimization procedure:

$$\left(\hat{\alpha}, \hat{\psi}, \hat{d}\right) = \text{warping}\left(\text{ss}, \text{tilt}_{\hat{\delta}}(\text{cv})\right) \tag{9}$$

The complexity of this sketched algorithm is in $\mathcal{O}(n)$ where n is the number of warping invocations necessary to find the optimum. This number depends on the strategy used to realize the argmin operation as well as the resolution of the tilt-parameter space.

2.3. Tilt Correction in Panoramic Images

In the proposed extension of the warping method, a panoramic image obtained from a potentially tilted camera is tilted back according to hypothetical tilt-correction angles. For this, it is necessary to describe how an image is affected by tilt. Following [15], a relation between the angular pixel coordinates (Θ, δ) and the 3D coordinate system (X, Y, Z) is

established by projecting the panoramic image onto (parts of) the unit sphere. This relation is illustrated in Figure 4.

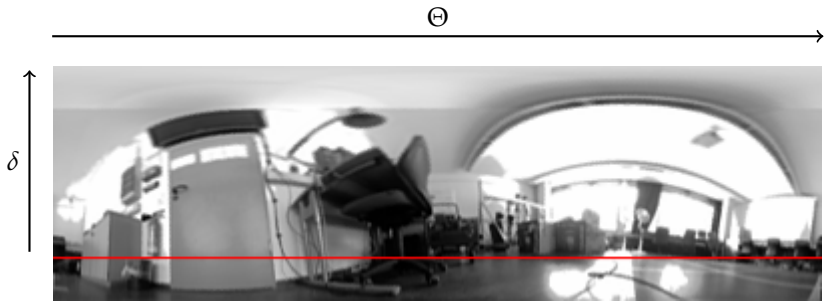


Figure 4. Angular coordinates of pixels in the panoramic image: modified spherical coordinates where the vertical angle δ is measured from the horizon (red line) which lies somewhere inside the vertical range of the image. The horizontal angle Θ runs from left to right in the reading direction.

The conversion between Cartesian and these modified spherical coordinates for the original and tilted coordinate system (X', Y', Z') is then expressed in the following way:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} c_\Theta c_\delta \\ s_\Theta c_\delta \\ s_\delta \end{pmatrix}, \quad \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} c_{\Theta'} c_{\delta'} \\ s_{\Theta'} c_{\delta'} \\ s_{\delta'} \end{pmatrix} \tag{10}$$

Tilt correction is applied to the image by copying the source pixel at (Θ', δ') to the destination pixel at (Θ, δ). A relation between angular coordinates and pixel indices (i, j) is established via the horizontal resolution hr and vertical resolution vr of the panoramic image. These parameters (in rad/pixel) are stored as image support data.

$$(i, j) = (\Theta/hr, \delta/vr) \tag{11}$$

The pixel indices i and j are generally real-valued and require rounding or interpolation to be mapped to integer-valued pixel indices.

The original and the tilted Cartesian coordinates are related according to the tilt described in the axis-angle representation by applying the rotation of the tilt angle φ about the X-axis:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\varphi & -s_\varphi \\ 0 & s_\varphi & c_\varphi \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{12}$$

This assumes the coordinate system to already be aligned with the tilt axis, which can be realized most conveniently by introducing an offset in the horizontal angle Θ corresponding to Θ_R. Before computing the source coordinates (Θ', δ'), the destination horizontal angle Θ is shifted by $-\Theta_R$ to align the subsequent computations with the X-axis. Note the backwards direction as Θ_R is measured in a mathematically positive direction but the horizontal angle Θ in reading direction, i.e., mathematically negative:

$$\Theta \leftarrow \Theta - \Theta_R \tag{13}$$

Afterwards, to not introduce an actual image rotation, the source horizontal angle Θ' is shifted back by Θ_R in the opposite direction:

$$\Theta' \leftarrow \Theta' + \Theta_R \tag{14}$$

The tilt direction Θ_R can be neglected in the following, deriving the special case for a tilt aligned with the X-axis, which can be achieved in the general case by the offsetting as just

described. By inserting Equation (12) into (10), the following *exact* solution in terms of angular pixel coordinates can be obtained [15]:

$$\Theta' = \operatorname{atan2}\left(c_\varphi c_\Theta c_\delta + s_\varphi s_\delta, c_\Theta c_\delta\right) \tag{15}$$

$$\delta' = \arcsin\left(-s_\varphi s_\Theta c_\delta + c_\varphi s_\delta\right) \tag{16}$$

To accelerate computations, an *approximate* solution is derived by eliminating multiplications and trigonometrical functions from Equations (15) and (16). Again this is done by following [15] assuming a small tilt angle φ and small vertical angle δ (note, however, that Θ is not small as it covers the whole 0° to 360° range):

$$\Theta' \approx \Theta + s_\varphi c_\Theta \delta \tag{17}$$

$$\delta' \approx \delta - s_\varphi s_\Theta \tag{18}$$

In addition to the exact and approximate solutions, the so-called *vertical* solution is derived by further approximating from Equation (17), assuming the effect of the tilt on the horizontal coordinate to be negligible. Thus, this method for tilt correction operates only within vertical image columns which are just shifted vertically:

$$\Theta' \approx \Theta \tag{19}$$

This simplification reduces the computational effort from two computations per image pixel, i.e., Θ' and δ' for each Θ–δ pair, to only one computation per image column, i.e., only δ' for each Θ.

All three tilt-correction solutions (exact, approximate, and vertical) can finally be combined with either nearest-neighbor or bilinear interpolation schemes involving the 4 nearest neighbors when mapping from real-valued angular or pixel coordinates (see Equation (11)) to integer-valued pixel indices. In total, this gives six methods for tilt correction in panoramic images. When transferring a pixel from source (Θ', δ') to destination (Θ, δ), it can happen that the source pixel (for the nearest-neighbor interpolation) or any of the source pixels (in case of a bilinear interpolation) is located outside of the original image: In case the source is only located horizontally outside of the image (Θ', to the left or right), the cyclic property of 360° panoramic images is used to access the corresponding pixel from the other side by taking horizontal pixel indices modulo the width of the image. If the source pixel is (additionally) located vertically (δ', to the top or bottom) outside of the image, there is no way to use or substitute a meaningful value from the original image and accessing such a pixel would practically result in an out of bounds access. Therefore, this is handled by a special case by inserting a predefined, unique, and reserved value to indicate invalid regions of the image, which is handled appropriately by all following image operations [10]. This value is called the *mask* value. It is typically set to the maximum of the range of possible pixel values, i.e., 255 (white) in case of 8-bit-per-pixel images. These regions with invalid pixels can appear at the upper (only in case of the approximate and vertical solution) or lower border of the image. An example of these invalid regions is shown in Figure 5.

Figure 5. Example of a tilt-corrected panoramic image showing white invalid regions introduced at the bottom to prevent out of bounds accesses. This sample image is produced using the exact tilt-correction solution with a nearest-neighbor interpolation.

2.4. Search Strategies

Using the roll-pitch representation of the tilt, a uniform search space with equal range and scaling for both parameter axes (in contrast to the axis-angle representation) can be constructed:

$$\delta = (\delta_X, \delta_Y) \in \text{Range} = [\delta_{\min}, \delta_{\max}]^2 \tag{20}$$

The two parameters, $\delta_{\min}$ and $\delta_{\max}$, were chosen to be small and symmetric (in their absolute values) about the zero tilt configuration in order to model the tilt as a small perturbation out of the plane as described before. In the most general case, the search range can be limited by forcing all candidate solutions outside to evaluate to a value bigger than any solution inside of the range. This is done by introducing a penalty or constraint term $f_c(\delta)$ added to the objective function value:

$$d \leftarrow d + f_c(\delta), \qquad \text{where } f_c(\delta) = \begin{cases} 0 & \text{if } \delta \in \text{Range} \\ \infty & \text{if } \delta \notin \text{Range} \end{cases} \tag{21}$$

To solve the optimization scheme sketched in Equation (8) using the uniform and limited search space, three *direct* search methods, i.e., methods only comparing function values of the objective without requiring it to be differentiable, are proposed in the following.

2.4.1. Exhaustive Search

The exhaustive search strategy constructs a discretized search grid G_δ from the limited but continuous search range specified in Equation (20) by sampling points with a fixed step size Δ along both axes:

$$G_\delta = [\delta_{\min}, \delta_{\max}; \Delta]^2 = \{\delta_{\min} + i\Delta : i \in \mathbb{N}_0 \wedge \delta_{\min} + i\Delta \leq \delta_{\max}\}^2 \tag{22}$$

This does not require the introduction of the limiting penalty term (21), as the search grid is already a limited search space by construction. As this gives a finite and enumerable set of candidate solutions, an approximation of the optimal tilt correction can then be found among these grid points by systematically running an exhaustive search of warping over all tilt configurations in the search grid:

$$\hat{\delta} \approx \underset{\delta \in G_\delta}{\operatorname{argmin}} \{d : (\alpha, \psi, d) = \text{warping}(\text{ss}, \text{tilt}_\delta(\text{cv}))\} \tag{23}$$

An example of such a uniform search grid is depicted in Figure 6.

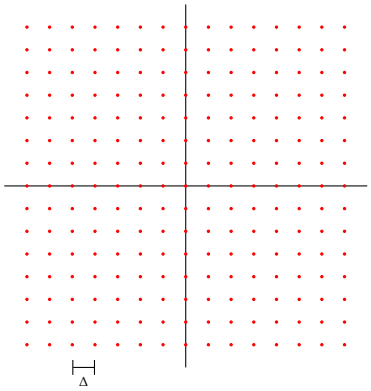


Figure 6. Exhaustive search: depiction of the uniform search grid (red dots) of spacing Δ. The dots mark the tilt configurations which are systematically probed to find the one minimizing the warping objective. The depicted grid has a total of 225 points, i.e., 15 tilt configurations per axis. The black lines mark the center axes of the search space, crossing at the $(0, 0)$ tilt configuration.

It is straightforward to improve the quality of this approximation by increasing the resolution of the search grid. However, this comes at a runtime cost: the number of candidate solutions which need to be exhaustively probed, i.e., the size of the search grid G_δ, is quadratic in the number of points along each axis and thus scales inversely proportional to the square of the step size, i.e., the resolution of the search grid:

$$n = \|G_\delta\| = \left\lfloor \frac{\delta_{\max} - \delta_{\min}}{\Delta} + 1 \right\rfloor^2 \in \mathcal{O}\left(\frac{1}{\Delta^2}\right) \tag{24}$$

Thus, halving the step size (doubling the resolution) increases the number of candidate solutions (warping runs to be done, and thereby the runtime) by a factor of four. Therefore, the exhaustive search provides a single parameter Δ for controlling the trade-off between estimation quality and runtime.

2.4.2. Pattern Search

The heuristic pattern search strategy [17,18] tries to reduce the number of function evaluations by considering only a small subset P of points of the search space. This is achieved by surrounding an initial guess of the optimum at the center $c = (c_X, c_Y)$ with a cross-shaped pattern of candidate solutions. This pattern has a total of five points—two along each axis of the search space plus the center point—and is used to iteratively construct improved guesses by finding the optimum among these candidate solutions. At each step of the iterative procedure (for a depiction of the control flow see Figure 7), the pattern is characterized by two parameters: the center point c tracking the current optimum and the width w describing the distance between the center and the outer points. These are then adapted according to one of two update rules depending on the location of the new optimum c' among the five pattern points: the pattern can either *move* to the new optimum if it is one of the outer points or it can *shrink* in place if the new optimum is the old center.

The shrinking of the pattern is controlled by a coefficient $0 < s < 1$ which controls the rate at which the width of the pattern decreases. As this parameter is smaller than 1, the pattern only ever gets smaller. Typically it is set to 0.5, meaning the pattern halves at each shrinking step. Some pattern followed by the two update rules is depicted in Figure 8.

While no assumptions about continuity or differentiability of the objective are made, for the pattern search (as well as the Nelder–Mead search introduced below) to converge fast to the correct solution, it needs to yield at least a favorable landscape with a unique global minimum surrounded by a valley of decreasing function values. Furthermore, the landscape should not be too "bumpy", i.e., contain no or only few local minima in which

the search pattern may get trapped. To prevent the pattern from diverging, i.e., running away out of a reasonable search range, here it is indeed necessary to introduce the limiting penalty term (21). This forces a shrink step (or a move to some suboptimal but within-range solution) once the new optimum would be outside of the range of valid tilt configurations.

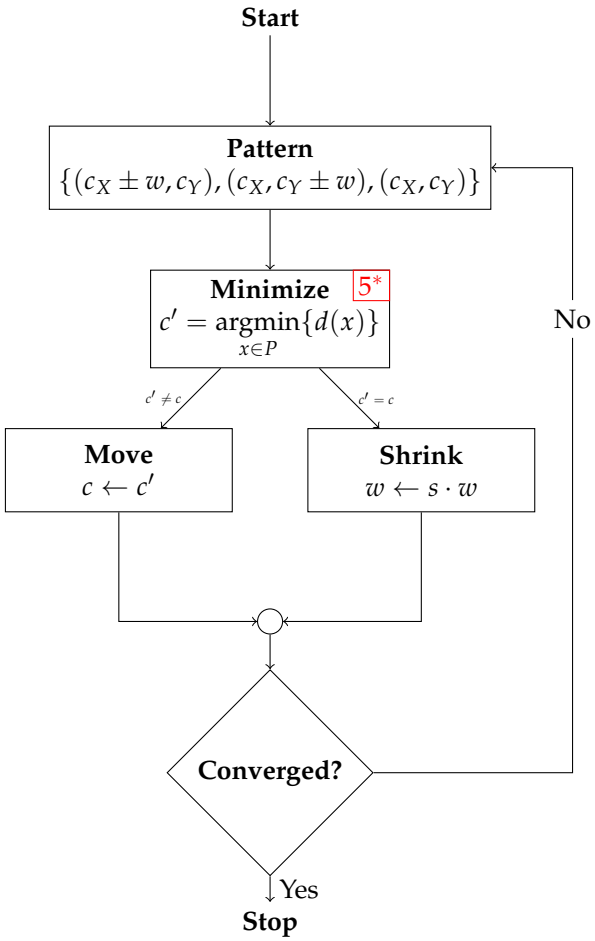


Figure 7. Flowchart of the pattern search algorithm. A red number at the corner of a box indicates the number of evaluations of the warping objective necessary at that step. However, by remembering already evaluated points (not all change at each update step), this number can in practice be reduced to 3 or 4 evaluations per iteration (to indicate this, the number is marked by a red star). Arrows annotated with inequalities indicate mutually exclusive branches and should be read like decision boxes, which are not drawn to keep the figure more compact.

Finally, as already hinted at in Figure 7, a convergence criterion must be given to stop the iterative procedure once a sufficiently good solution is found; this can be stated most naturally by comparing the width of the pattern w to some predefined threshold value τ beyond which the pattern is considered small enough to stop:

$$w < \tau \implies P \text{ converged} \tag{25}$$

To make the performance of the pattern search relatable to the exhaustive search, this threshold was chosen to be the same as the step size of the grid, meaning the pattern search stopped at solutions of similar resolution: $\tau = \Delta$. The initial center and initial width of the pattern were already given by the underlying search space as shown in Figure 8, the convergence threshold τ was chosen just as described to match the desired resolution of the search space Δ, and the shrink coefficient s was set without further exploration to the reasonable default of 0.5.

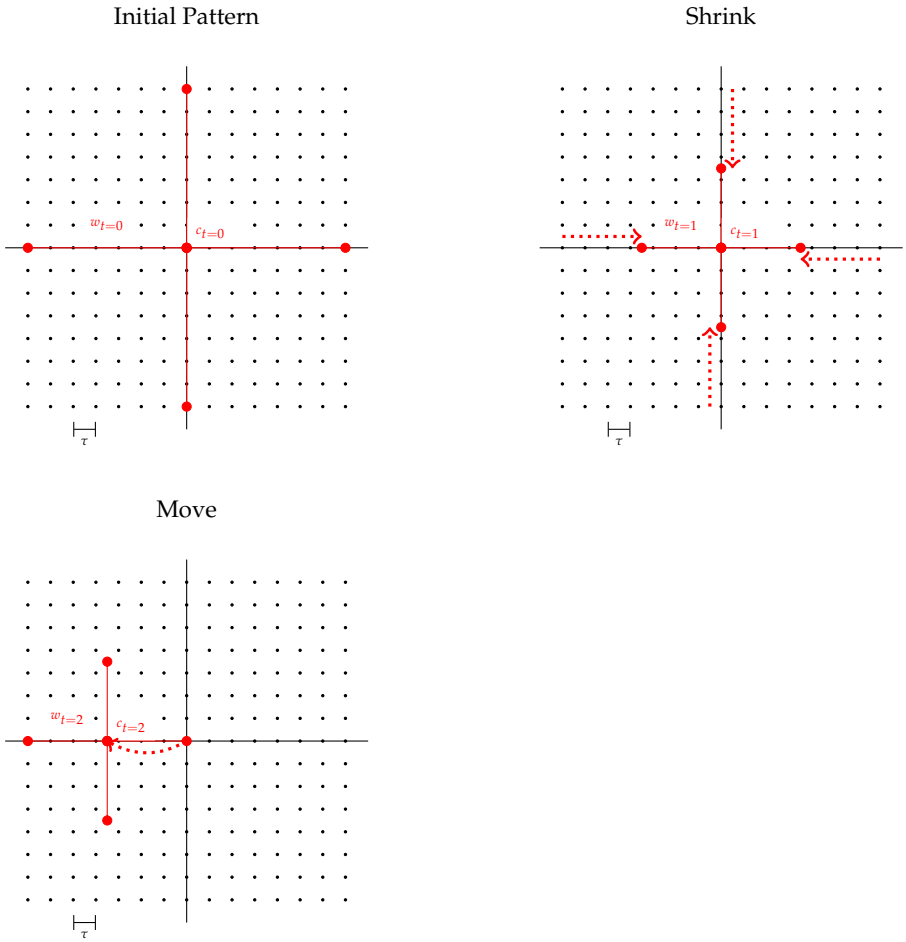


Figure 8. Pattern search: depiction of a sequence of the shrink and move operation starting from an initial pattern covering the whole search space (black dots). The spacing between the dots indicates the threshold resolution of the search pattern τ to test for convergence. The black lines mark the center axes of the search space, crossing at the $(0, 0)$ tilt configuration.

2.4.3. Nelder–Mead Search

Similar to the pattern search, the Nelder–Mead search strategy according to [19] is a heuristic and iterative procedure trying to reduce the number of function evaluations to minimize the objective. This method is also known as the simplex method, as it uses a simplex S—a polytope of $n + 1$ points in n dimensions, i.e., a triangle in the two dimensions used here—as the search pattern. At each step of the iterative procedure (for a depiction of the control flow see Figure 9), a replacement of the worst candidate among the vertices of the simplex is sought by constructing new candidates relative to these three points. The simplex can either *reflect*, *expand*, or *contract* along the line connecting the worst candidate to the so-called centroid between the two best candidates or *shrink* towards the best candidate if this yields no improvement. These four operations are each controlled by their corresponding coefficients: the reflection α, expansion γ, contraction ρ, and shrink coefficient σ. These parameters were all selected without further exploration to the best choice of parameters according to [19]: $\alpha = 1$, $\gamma = 2$, $\rho = 0.5$, and $\sigma = 0.5$. An example simplex and the four update rules are shown in Figure 10. As for the pattern search, it is necessary to prevent the procedure from diverging by introducing the limiting penalty term (21). Here, this effectively constrains the applicability of the reflection or expansion step, i.e., forces a contraction, when these would result in a vertex outside of the range of valid tilt configurations.

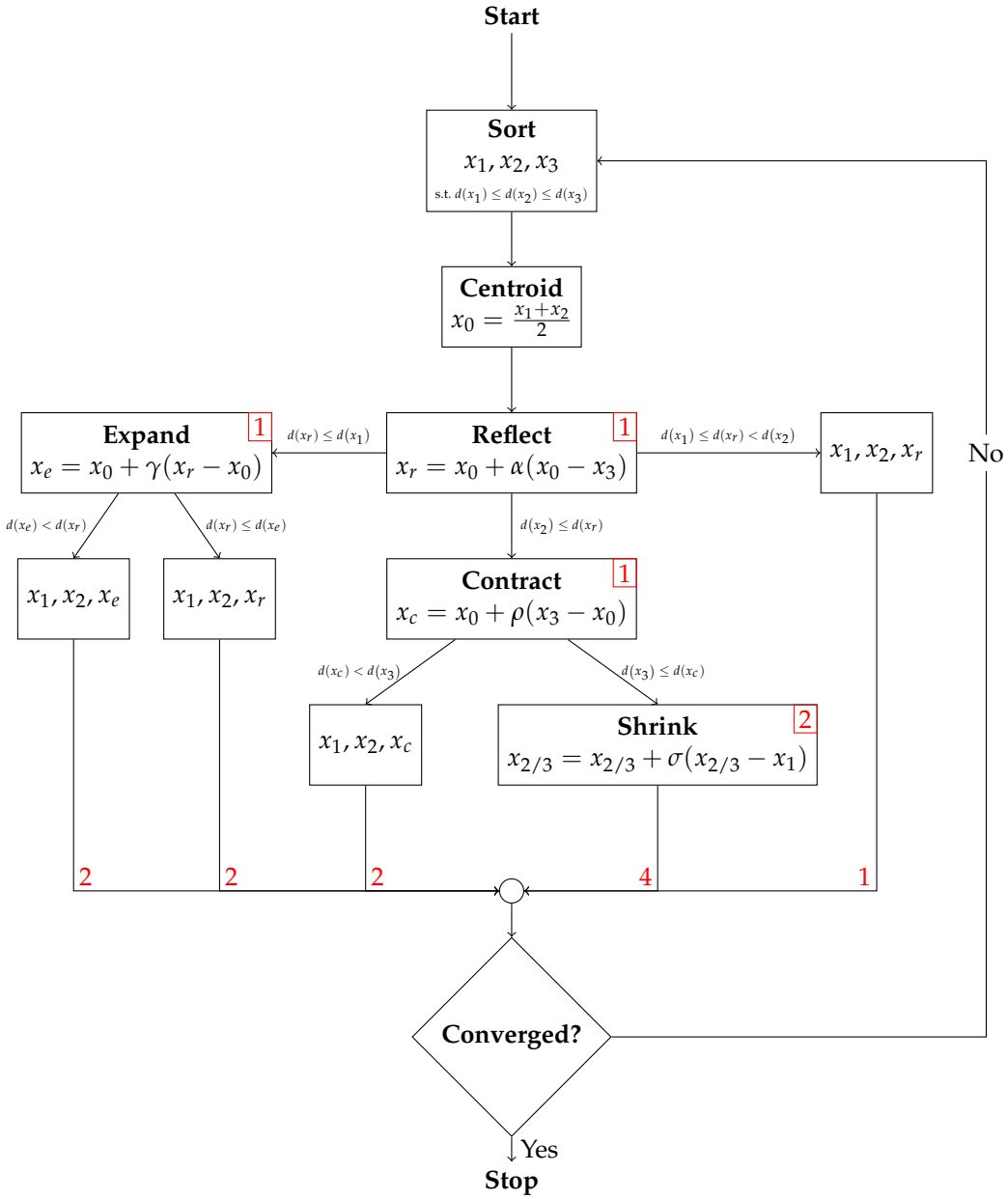


Figure 9. Flowchart of the Nelder–Mead search algorithm. A red number at the corner of a box indicates the number of evaluations of the warping objective necessary at that step. Right before merging all paths, red numbers indicate the total number of evaluations along each path. Arrows annotated with inequalities indicate mutually exclusive branches and should be read like decision boxes, which are not drawn to keep the figure more compact.

The initial simplex is constructed as shown in Figure 11 by placing one vertex at a corner of the search space and the other two at the center of the opposing edges. However, in contrast to the pattern search, it is not possible to construct a simplex of three points fully covering a square search space. While this naive initialization is sufficient for a first comparison of the search strategies, it is probably worthy of future investigations.

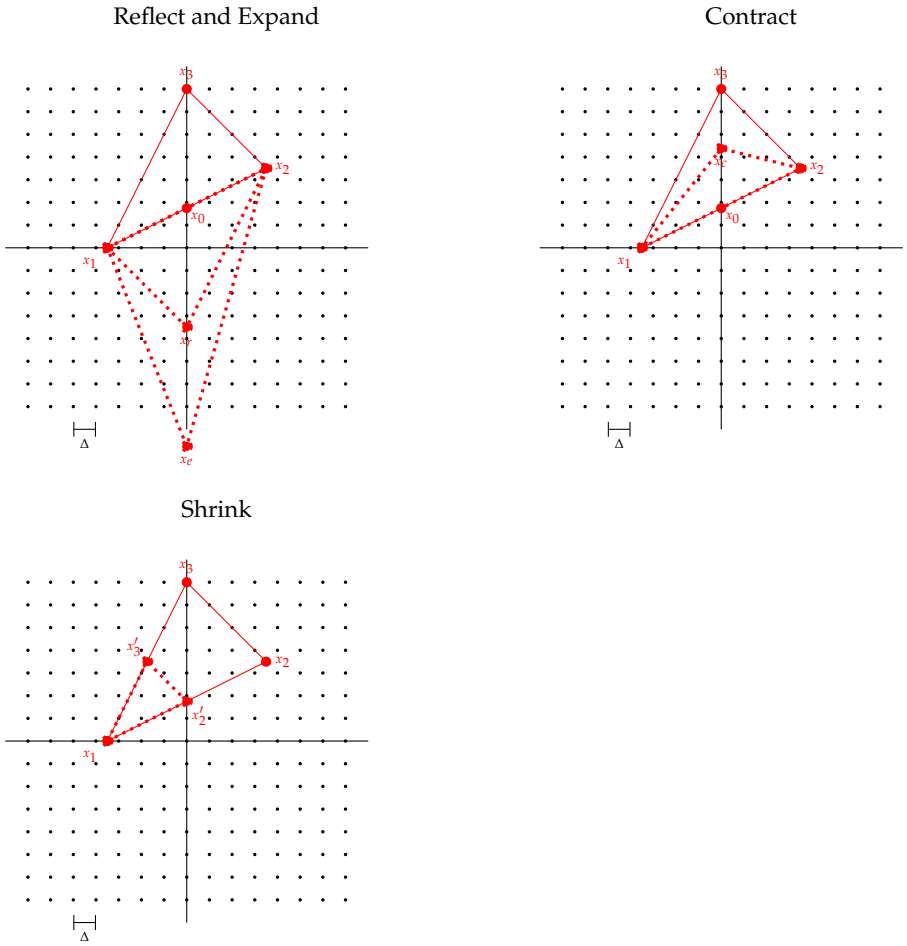


Figure 10. Nelder–Mead (simplex) search: depiction of the four update operations on the search simplex superimposed on the uniform search grid (black dots). Starting from the ordered set of vertices $[x_1, x_2, x_3]$, an improvement is sought by reflecting x_r, expanding x_e, or contracting x_c along the line connecting the worst candidate x_3 to the centroid x_0 between the best candidates. If no such improvement is found, the simplex shrinks towards the best candidate x_1. The black lines mark the center axes of the search space, crossing at the $(0, 0)$ tilt configuration.

Finally, to stop the iterative procedure once a sufficiently good solution is found, a convergence criterion must be given. Originally, ref. [19] proposed to use some predefined threshold τ compared to the sample standard deviation of the objective function values $d(x_i)$ at the vertices, which in the case of the three-vertex simplex looks as follows:

$$\mathrm{std}(d) = \sqrt{\frac{\sum_{i=1}^{3} \left(d(x_i) - \bar{d}\right)^2}{3}} < \tau \implies S \text{ converged} \tag{26}$$

However, as there is no prior knowledge on the range and distribution of the function values d produced by warping, besides the assumption that they are decreasing for better matching image pairs,this criterion yields no meaningful interpretation and is hard to adjust; the absolute value and variation of the distance estimate depends on aspects of the image pair and the configuration of the warping algorithm and does not allow to control the resolution in tilt-parameter space. Thus, a simpler and more interpretable criterion based on the size w of the simplex in search space is proposed, resembling that of the pattern search:

$$w < \tau \implies S \text{ converged} \tag{27}$$

This size of the simplex is estimated by the length of the longer side of an axis-aligned rectangular bounding box, which can be quickly computed by (elementwise) minimum and maximum operations on the set of vertices:

$$w = \left\| \max_{\text{elementwise}} (x_1, x_2, x_3) - \min_{\text{elementwise}} (x_1, x_2, x_3) \right\|_{\max} \tag{28}$$

The elementwise minimum operation (and the maximum analogously) over the three two-dimensional vertex coordinates is defined in the following way:

$$\min_{\text{elementwise}} \left(\begin{pmatrix} x_{11} \\ x_{12} \end{pmatrix}, \begin{pmatrix} x_{21} \\ x_{22} \end{pmatrix}, \begin{pmatrix} x_{31} \\ x_{32} \end{pmatrix} \right) = \begin{pmatrix} \min(x_{11}, x_{21}, x_{31}) \\ \min(x_{12}, x_{22}, x_{32}) \end{pmatrix}$$

The procedure of constructing this bounding box is depicted in Figure 11.

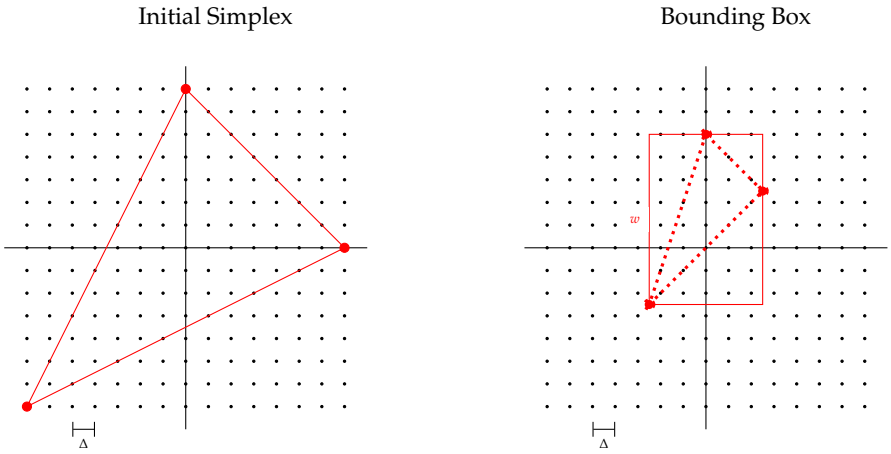


Figure 11. Nelder–Mead (simplex) search. The left figure shows the initialization of the simplex covering most parts of the search space by placing the vertices at two edges and a corner of the search space. The right figure shows the axis-aligned bounding box used to estimate the size of the simplex by the length w of the longer side of the box.

While other notions of the size of the simplex might be conceivable, e.g., computing the area or volume of the search space covered by the simplex, this bounding-box approach is the easiest to interpret (it also yields an upper bound on the area), as the width of the bounding box can be directly related to the width of the search pattern or the step size of the search grid. This allows us to adjust the threshold parameter to a comparable resolution, $\tau = 2\Delta$, as the pattern measures a center-to-edge distance while the bounding box measures an edge-to-edge distance (twice the center-to-edge distance).

3. Image Databases

The proposed methods were evaluated by conducting experiments on two newly collected real-world tilted panoramic image databases. These databases were collected by placing the tiltable panoramic camera (iDS UI-3241LE-M-GL with a 220° Lensagon BF16M220DC fisheye lens by Lensation) on all intersection points of a regular Cartesian grid on the horizontal ground plane. The **LAB** database was collected in a laboratory room of the Computer Engineering Group at Bielefeld University. The layout of this room and the location of the 3.0 m × 3.0 m grid with a total of 16 positions at a spacing of 1.0 m is shown in Figure 12. The laboratory environment comprised a tidy workspace of desks and office chairs directly adjacent to the database grid and a more cluttered area filled with various items of lab equipment on the opposite side of the room. The **LIV** database was collected in a living room with a high ceiling on a more elongated 1.2 m × 3.0 m grid

of 18 positions at a spacing of 0.6 m. This environment comprised mostly large pieces of furniture such as tables, cupboards, and sofas as shown in Figure 13.

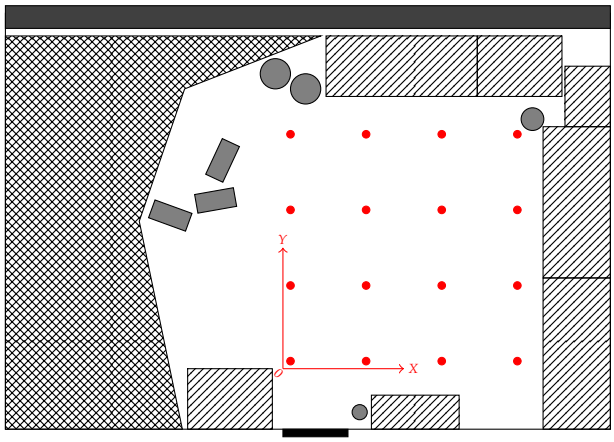


Figure 12. Layout of the laboratory room where the LAB database was collected. The database covered an area of 3.0 m × 3.0 m in steps of 1.0 m. The grid of image locations is marked in red. The whole room had a size of 8.0 m × 5.5 m. The areas comprising furniture such as cupboards, desks, or chairs are marked by hatched regions, and gray boxes and circles mark the position and rough shape of small items such as card boxes and fans. The door of the room is marked by the black bar at the bottom wall. The cross-hatched area on the left half of the room comprised a variety of laboratory equipment, including robots, cables, and tools. The dark gray area along the top wall of the room marks the window facade, which was partially covered by curtains.

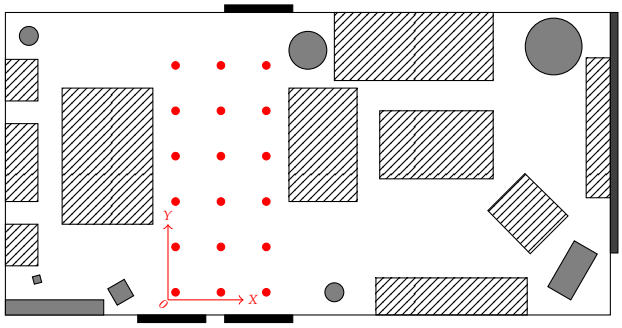
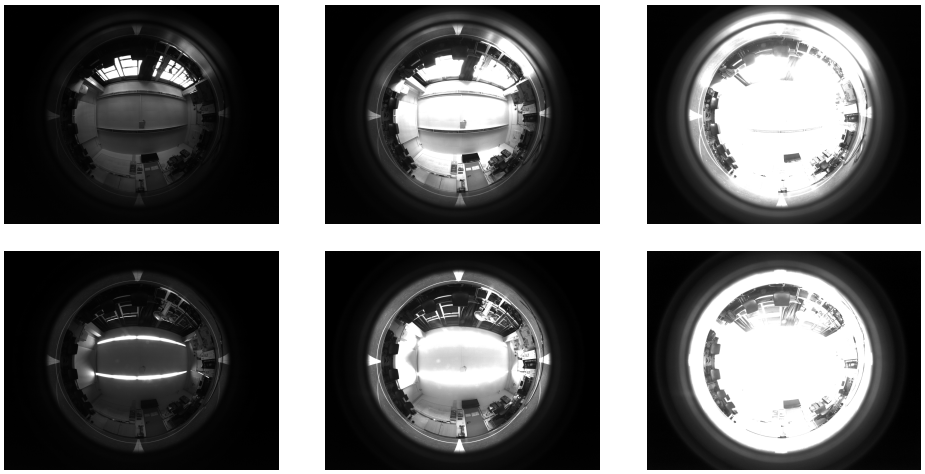


Figure 13. Layout of the living room where the LIV database was collected. The database covers an area of 1.2 m × 3.0 m in steps of 0.6 m. The grid of image locations is marked in red. The whole room had a size of 8.0 m × 4.0 m and each of the furniture items was at least partially visible at one of the locations. The areas comprising furniture such as cupboards, tables, chairs, or sofas are marked by hatched regions, and gray boxes and circles mark the position and rough shape of decor items. The three doors of the room are marked by the black bars. The dark gray area along the right wall of the room marks the window facade, which was partially covered by shutters and curtains.

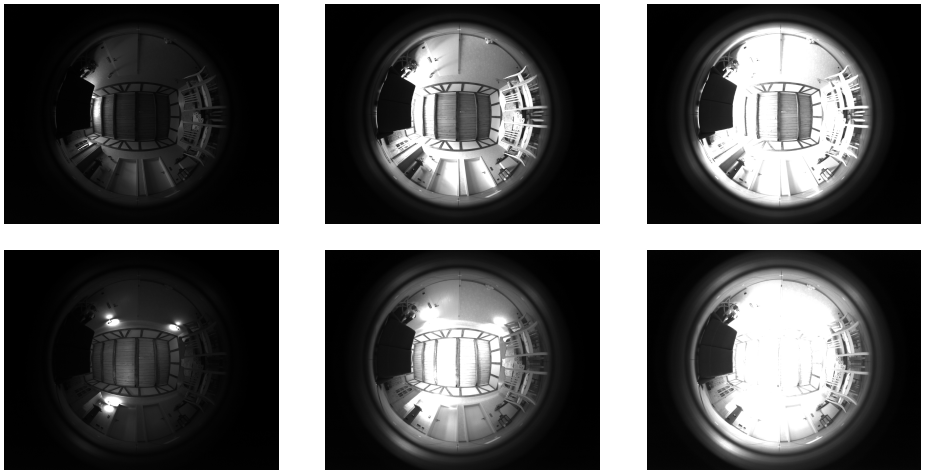
Variants under varying conditions of illumination—one during the day and one during the night—were collected for both databases. During these two data collection sessions, the illumination was kept as constant as possible, and an exposure control mechanism was used to keep the average brightness of the images constant as well. Additionally, to make the databases reusable for future high-dynamic-range (HDR) experiments, under- and overexposed variants of each image were collected using an exposure time which was derived from the controlled average brightness by a constant factor: $\times 0.2$ for the underexposed and $\times 2.0$ for the overexposed image. Thus, there were in total three differently illuminated images at each position in the night variant. In the daylight variant, three differently illuminated images were saved for each combination of position and tilt. A set

of sample images of these variants from the LAB and LIV database is shown in Figure 14 and Figure 15, respectively.



Underexposed Average Brightness 128 Overexposed

Figure 14. Sample images from the LAB database. All images were collected at position $(1, 2)$ of the database grid in the $(0, 0)$, i.e., no-tilt, configuration. The images of the top row were collected during the day with bright light shining through the windows and artificial lighting turned off. The images of the bottom row were collected during the night at the same pose. At night there was no light shining through the windows and the artificial lighting was turned on. There were no changes to the environment between the two variants besides the illumination conditions.



Underexposed Average Brightness 128 Overexposed

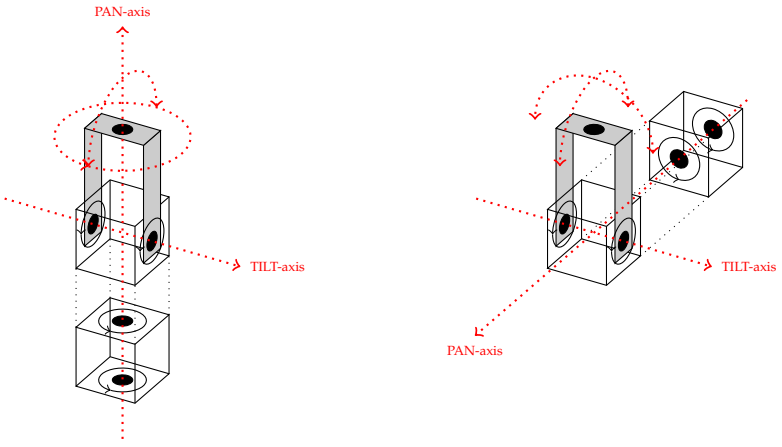
Figure 15. Sample images from the LIV database. All images were collected at position $(1, 3)$ of the database grid in the $(0, 0)$, i.e., no-tilt, configuration. The images of the top row were collected during the day with bright light shining through the windows and artificial lighting turned off. The images of the bottom row were collected during the night at the same pose. At night there was no light shining through the windows and the artificial lighting was turned on. There were no changes to the environment between the two variants besides the illumination conditions.

3.1. Pan–Tilt Unit

The day variant of both databases comprised images captured with a systematic tilt applied to the camera at each grid location. This was done by mounting the camera on top of a pan–tilt unit (PTU) (PTU-C46 by Directed Perception, Inc., Burlingame, California/ FLIR Systems) allowing for rotations about two orthogonal axes, i.e., the PAN- and the TILT-axis. In its standard configuration, the PTU was setup in a way corresponding to the axis-angle representation of the tilt: rotated about the PAN-axis to select the direction

Θ_R of the tilt axis ω and then rotated by φ around the TILT-axis. Using the conversion derived earlier in Equations (3) and (4), it was also possible to control the PTU with angles given in the roll-pitch representation. However, in contrast to the roll-pitch and axis-angle representations, which were formulated to avoid any relation between the tilt direction and the view direction of the robot, the PTU actually changed the view direction when applying tilt. This was due to the PAN-axis not only selecting the direction of the TILT-axis but also rotating the camera about its optical axis. This caused a coupling of the view direction of the robot to the tilt which was always present when using the PTU in its standard configuration.

To avoid this problem, the PTU was mounted in a flipped configuration, flipping the PAN-axis horizontally by rotating 90° about the TILT-axis. In this configuration, the PAN- and TILT-axis spanned the horizontal plane and could be directly associated with the roll-pitch representation of the tilt. As the camera was still mounted upright with its optical axis aligned to a third axis orthogonal to the rotating axes, this configuration had no relation between the tilt and the view direction. Thus, the view direction was fully determined by the static orientation of the camera mount. This view direction needed to be taken into account during the image preprocessing as well as during the evaluation of the results, as it corresponded to one of the homing parameters estimated by warping. A schematic of both PTU configurations is shown in Figure 16 and a photo of the modified PTU is shown in Figure 17. The operating range of the PTU was $\pm 159°$ on the PAN- and $-47°$ to $+31°$ on the TILT-axis, which was more than sufficient for the small amounts of tilt considered in this work. In both configurations, tilting the camera led to a small change in the camera position, i.e., a small offset in all three spatial dimensions. However, considering the small amounts of tilt, this change in position was small enough with respect to the database grid resolution to be neglected in the experiments—in the worst case, this yielded a displacement of only about 3 cm compared to the 100 cm or 60 cm resolution of the grid. The change in position depended on the tilt and the offset of the camera mount above the plane spanned by the moving PAN- and TILT-axis: in the standard configuration, the camera was mounted about 8.15 cm and in the flipped configuration about 17 cm above the intersection of the PAN- and TILT-axis. Including the immovable parts of the setup, these offsets corresponded to images collected about 18.15 cm or 36 cm, respectively, above the ground.



Standard/Axis-Angle Flipped/Roll-Pitch

Figure 16. Two configurations of the pan–tilt unit (PTU). The standard configuration (**left**) is the one giving the PTU its name with the two servo modules mounted on top of each other. This configuration matched the axis-angle representation of tilt where the PAN-axis rotation selected the tilt axis and the TILT-axis rotation the amount of tilt. However, with the camera mounted rigidly on top of the setup,

this caused a change of view direction due to the panning as well and thus coupled the view direction to the tilt. To get a configuration which decoupled the view direction from the tilt, the PTU was flipped on its side (**right**) by mounting it at a 90° angle about the TILT-axis while keeping the camera mount upright. This way, the PAN- and TILT-axis spanned the horizontal plane and could be associated with the X- and Y-axis of the roll-pitch representation. See Figure 3 for a depiction of the two tilt representations.

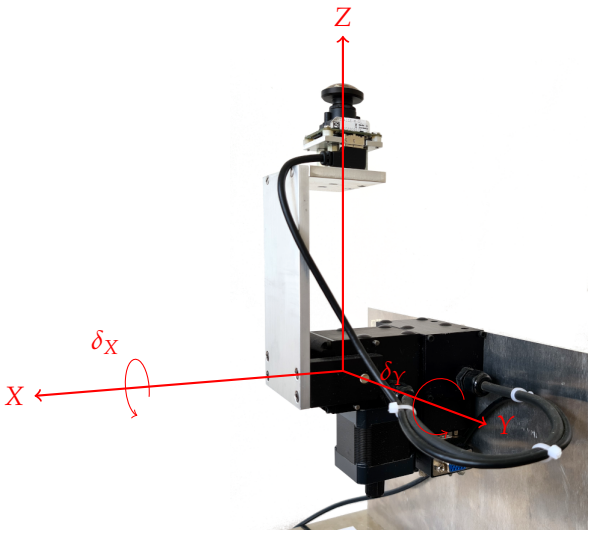


Figure 17. PTU in flipped/roll-pitch configuration. Photo of the modified PTU and camera mount for decoupling the view direction from the tilt. See Figure 16 for a schematic view of this configuration. The coordinate system associating the X- and Y-axis of the roll-pitch representation with the PAN- and TILT-axis, respectively, is drawn in red. The origin is placed at the intersection of the PAN- and TILT-axis. The camera mount is centered above (along the Z-axis) the origin.

LAB Tilt Range For collecting the LAB database, the PTU was used in its standard configuration with the tilt parameters converted to the axis-angle representation from a uniform grid in the roll-pitch representation. This grid of tilt parameters covered a range of ±5° of tilt per X- and Y-axis in steps of 1°:

$$\delta_X, \delta_Y \in [-5°, +5°; \Delta = 1°] \tag{29}$$

In total, there were 121 different tilt configurations per position in the day variant of the LAB database. In terms of magnitude (see Equation (4)), this comprised up to $\varphi \approx 7°$ of tilt. From using the PTU in its standard configuration and mounting the camera facing backwards relative to the forward facing X-axis (i.e., at 180°) and requiring a 180° PAN range to select all tilt directions, the view direction δ_R systematically varied between 180° and 360°, depending on the PAN-axis directions, which themselves depended on the two tilt parameters δ_X and δ_Y.

LIV Tilt Range For collecting the LIV database, the PTU was used in the flipped configuration, with the tilt parameters in the roll-pitch representation taken from the uniform grid without conversion. The LIV database comprised a slightly larger range of tilt of ±8° per X- and Y-axis in steps of about 1.15°:

$$\delta_X, \delta_Y \in [-8°, +8°; \Delta = 1.15°] \tag{30}$$

In total, there were 225 different tilt configurations per position in the day variant of the LIV database. In terms of magnitude, this comprised up to $\varphi \approx 11.3°$ of tilt. With the PTU used

in the flipped configuration, the view direction δ_R was constant and fully determined by the camera mount, which was oriented at $\delta_R = -90°$ relative to the forward facing X-axis.

3.2. Image Processing

The image processing generally followed the scheme described by [3,5]. The path of a sample image from the LAB database through the preprocessing stages is shown in Figure 18. Starting from the original image (step 1) with a size of 1280×1024, histogram equalization was applied to increase the contrast (step 2). The equalization was implemented such that only the ring-shaped region of valid pixels indicated by the mask shown in Figure 19 was considered. The equalized image was then low-pass-filtered to avoid aliasing artifacts due to the subsampling of the images while applying transformations to panoramic images (step 3). For all experiments, a third-order Butterworth low-pass filter with relative cutoff frequency of 0.2 was used.

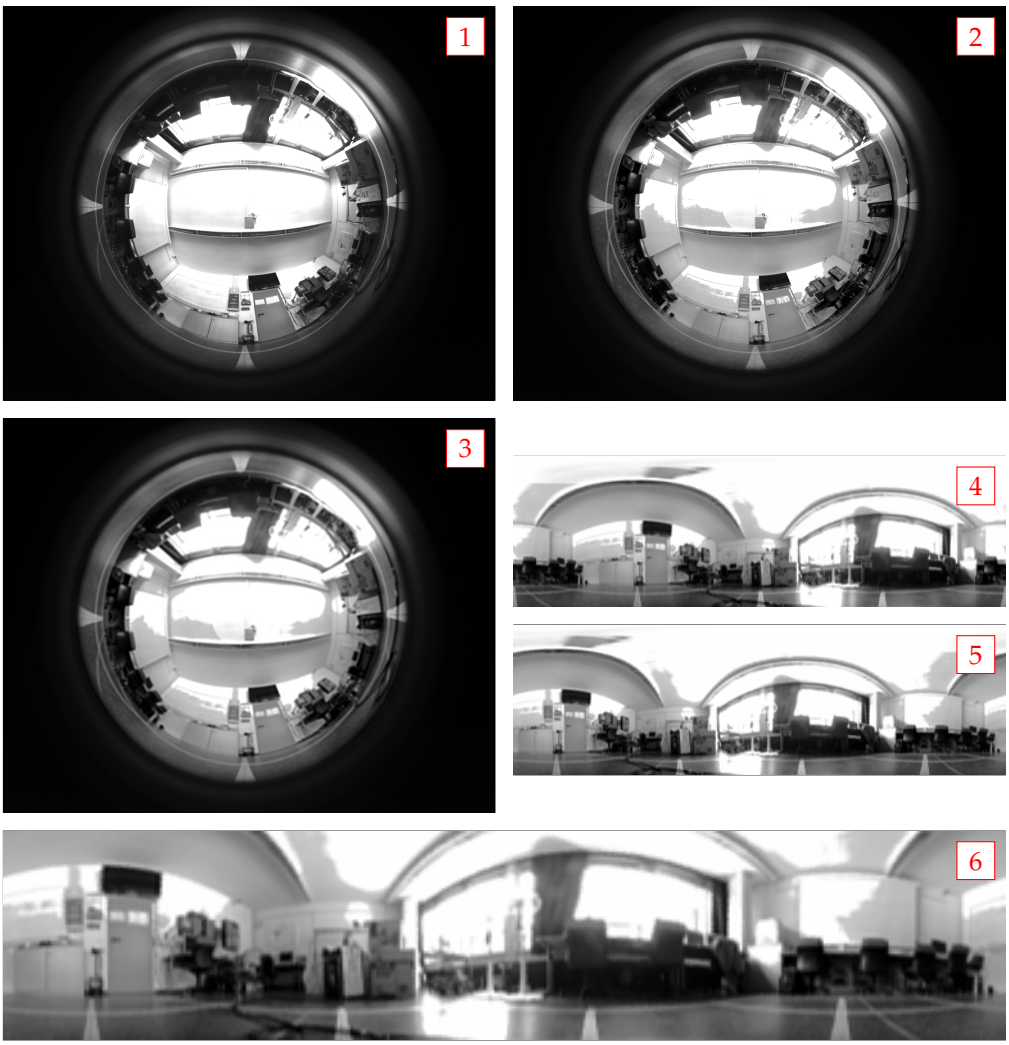


Figure 18. Path of a sample image from the day variant of the LAB database at position $(1, 2)$ without tilt through the image processing pipeline: (**1**) original image from the database; (**2**) improved contrast after histogram equalization; (**3**) low-pass-filtered using a 3rd-order Butterworth filter with relative cutoff frequency of 0.2; (**4**) unrolled panoramic representation; (**5**) simulated rotation of 60° by horizontal left shift with wrap-around; (**6**) cropping of upper 35° to remove the distorted ceiling and potentially invalid regions at the top.

From this preprocessed image, the panoramic representation (step 4) was unrolled such that the horizontal dimension corresponded to the azimuthal angle of a pixel and

the vertical dimension to its elevation angle. This unrolling was done by a mapping projecting the image onto the unit sphere, which was sampled with an equidistant spacing. To construct the projection onto the unit sphere, a camera model was required, which was obtained from a set of images showing a checkerboard calibration pattern as shown in Figure 19. These calibration images were processed into a set of camera parameters using the OCamCalib Toolbox [20]. During the unrolling process, all images were brought into the same orientation—facing forward aligned with the X-axis of the database grid—by accounting for the view direction δ_R due to the camera mount and PTU configuration mentioned earlier. The panoramic representation comprised all of the available field of view of the camera, meaning it ranges from $-20°$ below the horizon to $+90°$ above the horizon. The resulting image had a size of 384×117 pixels at a vertical resolution within an image column of approximately 0.016 rad/pixel and the horizon level at 95.72 pixels counting from the top. The sampling of the unit sphere used a bilinear interpolation.

Calibration Mask

Figure 19. Camera calibration and mask image. The calibration image is one example of several images collected showing a checkerboard pattern [21]. These calibration images were used to obtain the camera parameters necessary for "unrolling" the image into a panoramic representation [20,22]. The mask image was used for exposure control and preprocessing to ignore those regions of the image which were not exposed at all or contain no meaningful information, such as the typically overly bright ceiling. The ring-shaped region extended from about 20° on the inner side to about 110° on the outer side, or in terms of elevation and depression, up to 70° above and 20° below the horizon.

All preprocessing steps described until now (steps 1–4 in Figure 18) were applied only once for each database (for each image) and thus were not included in the time measurements of the experiments. The final two preprocessing steps (described below) were applied once per image pair (step 5, rotation) or even multiple times per image during the tilt correction search, every time the image was warped (step 6, cropping).

As all panoramic images had the same orientation after the preprocessing, warping would always find an estimate of the view direction at $\hat{\psi} = 0$, which was exactly contained in the discrete search space over α and ψ as described in more detail by [5]. To avoid this and get a more realistic homing performance, both the snapshot and current view were rotated by random azimuth angles δ_R^{ss} and δ_R^{cv} similar to [3]. This rotation corresponded to a horizontal shift with a wrap-around in the panoramic representation; while the images were unrolled in the reading direction, a positive rotation corresponded to a left shift against the reading direction. This randomization of the view directions was done for each snapshot–current view pair and kept the same for all methods evaluated on the image pair. As the tilt corresponding to each image was described in roll-pitch representation relative to the fixed coordinate system aligned with the database grid, the rotation of the image required the tilt parameters to be adapted into this rotated coordinate system to remain valid for the tilt correction or as ground truth for matching the tilt estimates: under the assumption of a small tilt, i.e., still keeping the tilted plane close to being parallel to the ground plane, the

introduction of an additional rotation of δ_R about the Z-axis approximately corresponded to a rotation of the two roll-pitch parameters within the X–Y plane:

$$\begin{pmatrix} \delta_X^* \\ \delta_Y^* \end{pmatrix} = \begin{bmatrix} \cos\delta_R & \sin\delta_R \\ -\sin\delta_R & \cos\delta_R \end{bmatrix} \begin{pmatrix} \delta_X \\ \delta_Y \end{pmatrix} \tag{31}$$

A detailed derivation of the exact adaptation of the tilt parameters under rotation and these approximations is given in the Appendix A.2.

Some image transformations, in particular the tilt correction, yielded invalid regions at the top and bottom border of panoramic images. As the scale-plane stack computations of the given warping implementation only handled these regions at the bottom border, the upper part needed to be removed. To have comparable images throughout all experiments, a constant cropping of 35° from the top was applied to all images, i.e., tilt-corrected and not tilt-corrected, just before being passed to warping. This chosen amount of cropping was more than sufficient to remove the biggest invalid regions to be expected at the most extreme tilt correction of $\varphi \approx 11.3°$. Furthermore, this cropping may be beneficial to the homing performance of warping as most images showed only distorted ceiling without much detail in the cropped region. At the vertical resolution of the panoramic representation of approximately 0.016 rad/pixel, this cropping corresponded to approximately 37.25 pixels, resulting in a final size of the panoramic image of 384×80 pixels and a horizon level at 58.47 pixels counting from the top. By using the fact that the images were stored in row-major order, the image cropping was implemented efficiently by making a view into the memory of the original image at the cropping offset. Thus, even though the cropping needed to be applied multiple times throughout the search for an optimal tilt correction, this added no significant runtime overhead.

4. Results

From each of the two databases `LAB` and `LIV`, three datasets of snapshot/current view image pairs were generated. These datasets represented situations of varying difficulty for the tested methods. The `DxD` datasets took both snapshot and current view from the day variant of the database, meaning the illumination conditions did not vary by much. As the proposed tilt-correction scheme handled only one image being tilted, the snapshot images were constrained to those (one per position) with no tilt applied to the camera. The `NxD` datasets yielded so-called cross-database tests with snapshots taken from the night variant and current views from the day variant, resulting in substantial illumination differences between snapshot and current view images. These datasets allowed us to test the methods for tolerance against changes in illumination which are known to affect the performance of visual navigation methods in general [8,9]. Finally, the `NxN` dataset was generated from only the night variant of the databases representing the most simple case with no illumination changes and no tilt applied to any of the images. This last dataset served more as a reference for testing tilt-correction methods when no tilt was actually present. Experiments were conducted over all image pairs from these datasets testing all method combinations of components proposed in Section 2, i.e., all combinations of search strategy, tilt correction, and interpolation methods.

4.1. Nomenclature and Parameters

From the three tilt-correction solutions and the two interpolation schemes proposed in Section 2.3, six tilt correction methods were implemented. The identifiers used to distinguish the tested methods were composed from the initial letters of the methods and are listed in Table 1.

These tilt-correction methods were then combined with warping and one of the three search strategies proposed in Section 2.4. The resulting methods are referred to by prepending the identifier of the search strategy to that of the tilt correction. In addition to the three search strategies, two baseline strategies were added to the method pool. The **true** tilt strategy used the known ground truth tilt parameters as recorded in the database.

This corresponded to cases where an additional sensor, such as an inertial measurement unit, was combined with warping to provide the tilt parameter estimates. Furthermore, this allowed us to investigate how the different tilt-correction methods affected the homing performance without interference of a particular search strategy. Finally, the **none** tilt strategy used the warping algorithm as it was, without any form of tilt correction. All 25 method combinations are listed in Table 2. When referring to the results on a particular dataset later on, an identifier of the dataset—comprising the database and the variants used—is appended to the method identifier: e.g., `ten@lab-nd` refers to the true tilt strategy using an exact tilt correction with a nearest-neighbor interpolation evaluated on the `NxD` cross-database test of the `LAB` database. All methods combined with warping [23,24] used the same configuration and parameters as listed in Tables 3–5. As these parameters never changed, they are not indicated by the method identifiers. As the first phase of warping, the distance image or the scale-plane stack computation, a modified version of the NSAD distance measure [9,23] was used, adapted to properly handle the invalid regions introduced by the tilt-correction methods [10]. Finally, Table 6 states the range and resolution parameters used to constrain the tilt search strategies.

Table 1. Identifiers of the six tilt-correction methods. The identifiers are composed of the initial letters of the methods, naming the tilt correction solution first, followed by the interpolation scheme.

	Nearest Neighbor	**Bilinear**
Exact	`en`	`el`
Approximate	`an`	`al`
Vertical	`vn`	`vl`

Table 2. Identifiers of the tilt-correction warping schemes. The identifier of the tilt (search) strategy is prepended to that of the tilt-correction method (see Table 1). The Nelder–Mead search uses `s` for the simplex method. The **true** tilt strategy applied a single ground-truth tilt correction as recorded in the databases, before running warping once. The **none** tilt strategy was a baseline test running warping without any tilt correction. This baseline was the same for both interpolation schemes as there was nothing to interpolate. In total, there were 25 combinations.

Nearest-Neighbor Interpolation				
	Exact	**Approximate**	**Vertical**	**None**
True	`ten`	`tan`	`tvn`	`-`
Exhaustive	`een`	`ean`	`evn`	`-`
Pattern	`pen`	`pan`	`pvn`	`-`
Nelder–Mead	`sen`	`san`	`svn`	`-`
None	`-`	`-`	`-`	`n--`
Bilinear Interpolation				
	Exact	**Approximate**	**Vertical**	**None**
True	`tel`	`tal`	`tvl`	`-`
Exhaustive	`eel`	`eal`	`evl`	`-`
Pattern	`pel`	`pal`	`pvl`	`-`
Nelder–Mead	`sel`	`sal`	`svl`	`-`
None	`-`	`-`	`-`	`n--`

Table 3. Configuration of the image distance and scale-plane stack computation. A scale-plane stack of 9 distance images was computed with a scale factor up to 2.0 using a modified NSAD column distance measure, adapted to handle the invalid regions due to the tilt-correcting image transformations.

Distance Measure	Num. Scale Planes	Max. Scale Factor
NSAD with invalid region handling	9	2.0

Table 4. Number of steps, search range, and resolution of the warp parameter search space used by MinWarping.

Search Steps	Search Range	Resolution
$n_\alpha = 128$	$0° \ldots 357.1875°$	$\Delta_\alpha = 2.1825°$
$n_\psi = 128$	$0° \ldots 357.1875°$	$\Delta_\psi = 2.1825°$

Table 5. Configuration of the MinWarping searcher. MinWarping was configured to perform a double search, i.e., a second run with exchanged snapshot/current view, with compass acceleration.

Searcher	Compass Acceleration	Partial-	Double-	Fine-Search
MinWarping	Yes, Sum	No	Yes	No

Table 6. Range and resolution parameters used by the tilt search strategies. All three strategies were set up to cover the same symmetric search range of up to 0.14 rad of tilt per axis in the roll-pitch representation. This range comprised the maximum tilt of both databases (i.e., the range of the LIV database) and was used for experiments on both. The grid and pattern search resolution also matched the step size between tilt configurations of the LIV database, and the simplex threshold was derived from these as $\tau = 2\Delta$, as described earlier.

Search Range	Grid Δ	Pattern τ	Simplex τ
$\delta_{\min} = -0.14\,\text{rad}, \delta_{\max} = +0.14\,\text{rad}$	0.02 rad	0.02 rad	0.04 rad

4.2. Image Differences

For comparing the effect of the tilt-correction solutions on panoramic images, a pairwise difference measure was used to compare images transformed by two different solutions on the same input image. This was done by computing a normalized sum of absolute pixelwise differences:

$$\text{diff}(I_1, I_2) = \frac{1}{WH} \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} |I_1[i,j] - I_2[i,j]|, \tag{32}$$

where I_1 and I_2 are transformed from the same input image by different tilt-correction solutions or using different interpolation methods. W and H are the width and height of the panoramic image. Using this difference measure, a distance matrix of the pairwise comparisons of all tilt-correction solutions combined with the two interpolation methods (nearest neighbor and bilinear) was plotted. These comparison matrices were produced for each database by averaging over all images and are shown in Figure 20. Note that this image difference experiment omitted the last two preprocessing steps of rotating and cropping the image, as these were only required for the homing experiments. These average image differences clearly set the strongest, vertical approximation apart from the other two, which were more similar to each other than to the vertical. This was not surprising as the vertical approximation removed one degree of freedom, the horizontal direction.

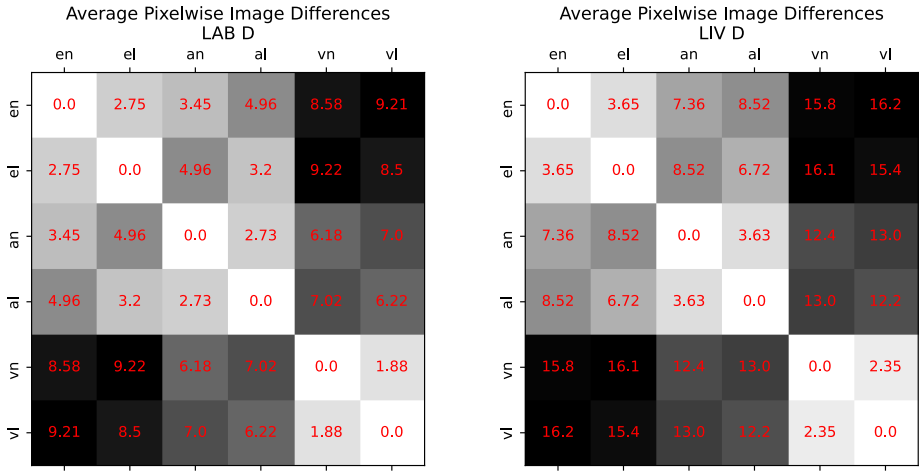


Figure 20. Plot of the average pixelwise image differences between all tilt-correction solutions. The LAB and LIV database are reported separately as the ranges of tilt and thus the expected deviations were different (LIV comprised more tilt). The results were averaged by taking the mean per method over all images contained in the database. For the tilt correction, the ground-truth tilt parameters as recorded in the database were used. The image difference measure is a symmetric distance measure, therefore the plotted matrices are symmetric about the main diagonal as well. With a stronger approximation, the effect of the different interpolation methods becomes smaller. The exact `e*` and approximate solutions `a*` are more similar to each other than to the vertical approximation `v*`.

The choice of interpolation method was only subordinate to the choice of approximation and the solutions were more similar to their interpolation variant than to the next level of approximation. This was also expected as the interpolation methods did not change anything about how the tilt correction operated but only improved on the image quality by smoothing out the lines and edges in the image. Figure 21 shows an example of an image tilt-corrected from one of the most extreme tilt configurations of the `LAB` database; the exact and approximate solutions were capable of straightening out most of the distortions due to tilt. With approximations, the deviations from straight lines became more noticeable farther away from the horizon, in particular at the top, where the small opening angle assumption was clearly violated. Without the horizontal component, the vertical approximation was not capable of straightening out curved vertical lines, and thus, noticeable distortions remained even close to the horizon, making this strongest approximation clearly inferior to the other two. This hinted at the fact that a single transforming degree of freedom, i.e., vertical shift, was not sufficient to do proper tilt corrections.

Original Tilted Image

Figure 21. *Cont.*

Figure 21. Sample image from the LAB database at position $(0, 0)$ in the most extreme $(5°, -5°)$ tilt configuration. The original tilted image is shown at the top. Below is the image after tilt correction based on the ground truth using the three approximation levels with a nearest-neighbor interpolation. Invalid regions appear at the bottom of the exact solution and at the top and bottom of the approximate solutions. The jagged edges are an artifact of the nearest-neighbor interpolation. While the exact and approximate solutions clearly straighten out most of the edges, the vertical approximation still seems to be more similar to the original, tilted image than to the other tilt-corrected images, in particular in regions farther away from the horizon. The effects become most obvious when looking at the door, which should be perfectly straight and upright after tilt correction. Note that this image difference experiment omits the last two preprocessing steps of rotating and cropping the image, and thus invalid regions can be seen at the top of the approximate and vertical solution.

4.3. Homing Performance

The homing performance was evaluated by comparing the angular error ϵ_β between the true home vector $\mathbf{h}$ and the home vector $\hat{\mathbf{h}}$ estimated by warping:

$$\epsilon_\beta = \arccos\left(\hat{\mathbf{h}}^T \mathbf{h}\right) \in [0, \pi] \tag{33}$$

These home vectors were unit-length vectors with direction $\hat{\beta}$ and β, respectively, relative to the world coordinate system, i.e., the database grid. The estimated home direction was computed from the parameters $\hat{\alpha}$ and $\hat{\psi}$ estimated by warping [3]:

$$\hat{\beta} = \hat{\psi} - \delta_R^{cv} - \hat{\alpha} + \pi \tag{34}$$

The view direction angle δ_R^{cv} was subtracted to convert from robot coordinates back to world coordinates to make the estimate comparable to the true home direction. This accounted for the randomized view direction which was introduced during preprocessing. The true home direction was derived as the angle between the positive X-axis and the vector pointing from the current view (X_{cv}, Y_{cv}) to the snapshot (X_{ss}, Y_{ss}) location on the database grid:

$$\beta = \text{atan2}(Y_{ss} - Y_{cv}, X_{ss} - X_{cv}) \tag{35}$$

Plotting all home directions per snapshot as arrows placed at the corresponding current view positions yielded a qualitative visualization of the homing performance. Each home vector was computed as the two-dimensional unit vector with polar angles $\hat{\beta}$ or β, respectively:

$$\mathbf{h} = \begin{pmatrix} \cos\beta \\ \sin\beta \end{pmatrix} \tag{36}$$

These vector field plots allowed us to observe the spread of the home direction estimates around the true home direction.

Figure 22 visualizes the effect of the different tilt-correction solutions on the home-direction estimates compared for the true tilt strategy. Furthermore, plotting the results of the no-tilt strategy clearly demonstrated how severely the homing performance of warping was affected by the tilt. While the differences between no-tilt correction and any tilt correction were vast, there was not much difference to observe when comparing the three tilt-correction solutions. Only in case of the vertical approximation could a slightly increased spread of the home vectors about the true home direction be observed. However, these differences were almost negligible compared to the no-tilt-correction baseline.

Figures 23 and 24 visualize the effect of the different search strategies on the home direction estimates when paired with the exact tilt correction and nearest-neighbor interpolation; almost all estimated home vectors aligned with the ground truth, and the exhaustive search strategy was even capable of fixing the "bad spot" in the top right corner $(3, 3)$ of the `LAB` database. As this was not the case for the true-tilt strategy, the exhaustive search must have found some image distortion not related to the ground-truth tilt but somehow improving the homing performance of warping. Another explanation for this might be wrong ground-truth data recorded for this position, although this was rather unlikely as the heuristic search strategies behaved similarly to the true-tilt strategy. Using the heuristic search strategies resulted in a markedly larger spread of the home direction estimates and, in particular for the `LIV` database, in more complete failures, i.e., home direction estimates not even roughly pointing in the right direction. The increased spread, which was still better than for the no-tilt correction, could probably be attributed to slight deviations of the tilt-correction estimate from the true tilt configuration. The complete failures were probably due to the heuristic strategies converging to a totally wrong minimum which potentially caused even worse distortions to the image than those due to the original tilt.

Finally, Table 7 summarizes the quantitative results of all methods on all datasets in terms of the angular error of the home direction ϵ_β. The results were averaged using the median as this represents the better measure for the skewed error distribution which is exemplarily plotted for one of the experiments in Figure 25.

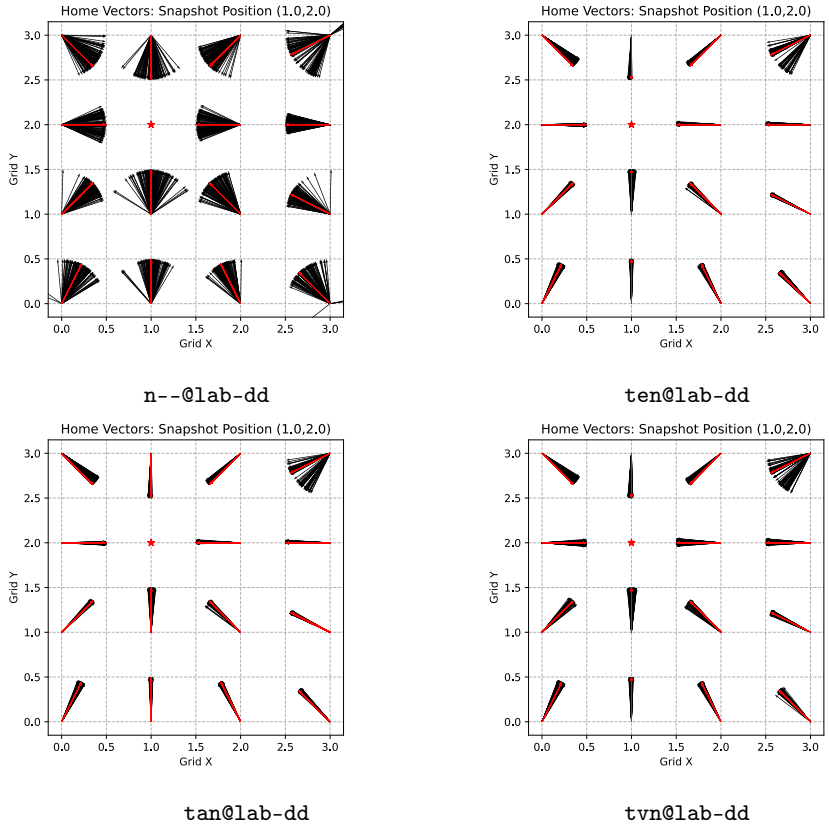


Figure 22. Home vector field plots comparing the tilt-correction solutions with a nearest-neighbor interpolation on the true tilt strategy. One sample snapshot location from the `LAB DxD` dataset is shown. The red arrows show the true home direction per position, the black arrows show the estimates obtained by the applied method—as there were 121 images per position in the dataset, there are also 121 home-direction estimates per position even though they might overlap in these plots. For this test, the snapshot position $(1, 2)$ near the middle of the grid was chosen and hence the arrows should ideally point there. The location $(3, 3)$ in the top right corner is a known "bad spot" of the `LAB` environment, near to multiple furniture items.

As already seen in the home-vector-field plots, there was not much difference between the approximations of the tilt correction when comparing within the groups of search strategies. The effect of the choice of the interpolation method was even almost negligible. Thus, the biggest influence stemmed from the search strategies. However, in the `LAB` database (with smaller amounts of tilt) the differences between the search strategies were still rather small. There were two noteworthy observations when comparing the search strategies to the true tilt strategy. First of all, on the same-database experiments `DxD` and `NxN`, the exhaustive search strategy was somehow capable of outperforming the true tilt baseline. This might be explained by the previous observation on the home-vector fields: the exhaustive search strategy might be capable of improving the homing performance of warping on "bad spots" in general, probably at the cost of tilt-correction performance. However, this effect was not observed on any of the cross-database experiments. Secondly, the heuristic search strategies showed worse performance on the no-tilt experiments `NxN` compared to the true tilt and exhaustive search strategy. This indicated that they tried to tilt-correct images which were not tilted at all and thus introduced misleading distortions.

In general, as expected, the tilt search was worse than the ground-truth tilt correction, and the heuristic search strategies yielded the worst performance. However, the differences between the pattern and the Nelder–Mead search strategies were rather small and inconclusive, only slightly favoring the pattern search (in particular on the `LIV` database comprising the bigger range of tilt). While all search strategies yielded similar performance over the same-database experiments (all similar within 2°), there was a distinctively bigger

difference over the cross-database experiments: in the worst case (`LIV NxD`), the pattern and Nelder–Mead search strategies were worse than the true tilt strategy almost by a factor of two, with the exhaustive search roughly in between, depending on the tilt-correction solution. However, the worst case of 16.67° was still better than the baseline without tilt correction at 34.60° by slightly more than a factor of two.

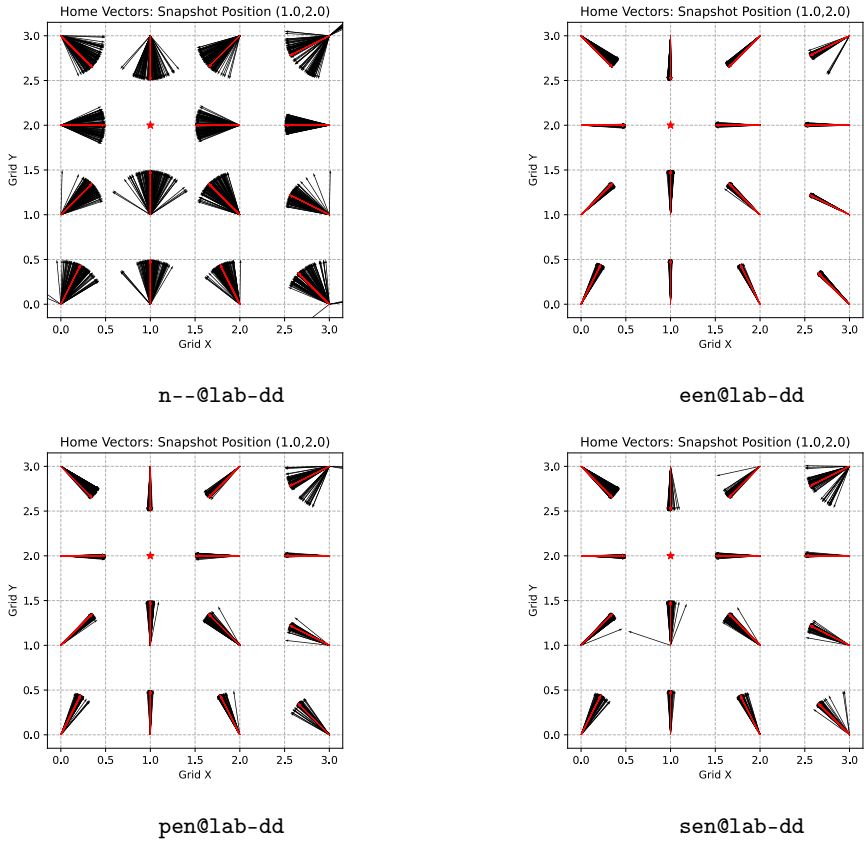


Figure 23. Home vector field plots comparing the tilt search strategies with a nearest-neighbor interpolation on the exact tilt-correction solution; the same sample snapshot location from the `LAB DxD` dataset as in Figure 22 is shown.

Table 7. Median of the home-direction error ϵ_β in degrees. The methods are grouped by tilt (search) strategy first and then arranged by tilt correction and interpolation method. In each group, the best-performing methods per dataset, i.e., those with the smallest home-direction error, are highlighted in red.

	LAB DxD	**LAB NxD**	**LAB NxN**	**LIV DxD**	**LIV NxD**	**LIV NxN**
`n--`	9.20	21.03	1.95	18.09	34.60	2.61
`ten`	2.21	**4.34**	1.95	2.98	6.65	**2.61**
`tel`	2.18	4.76	**1.93**	**2.93**	**6.27**	2.80
`tan`	2.23	4.36	1.95	3.11	6.80	**2.61**
`tal`	**2.20**	4.70	**1.93**	3.04	6.39	2.69
`tvn`	2.66	4.88	1.95	4.25	8.29	**2.61**
`tvl`	2.62	5.00	**1.93**	4.17	8.05	2.69

Table 7. *Cont.*

	LAB DxD	LAB NxD	LAB NxN	LIV DxD	LIV NxD	LIV NxN
een	**2.13**	4.54	1.93	2.96	8.83	2.48
eel	2.14	5.39	1.95	**2.87**	**7.37**	2.57
ean	2.14	**4.53**	1.95	3.14	9.06	2.36
eal	2.16	5.32	1.99	3.02	7.60	2.41
evn	2.62	5.16	**1.90**	4.35	11.38	2.25
evl	2.61	5.67	1.92	4.22	10.53	**2.21**
pen	**2.50**	**7.02**	**2.04**	3.70	13.19	2.78
pel	2.62	10.53	2.29	**3.52**	**11.14**	3.01
pan	2.51	7.12	2.18	3.83	13.17	2.72
pal	2.62	10.44	2.35	3.69	11.45	2.87
pvn	2.98	7.98	**2.04**	5.09	15.31	**2.57**
pvl	3.07	9.57	2.30	5.00	14.25	2.82
sen	**2.50**	**7.02**	2.30	4.41	14.39	2.59
sel	2.52	9.25	2.30	**4.34**	**12.08**	2.73
san	2.51	7.09	2.29	4.63	14.63	**2.53**
sal	2.57	9.01	2.33	4.55	12.40	2.84
svn	2.97	7.58	2.23	5.90	16.67	2.94
svl	2.98	8.55	**2.21**	5.83	15.22	2.82

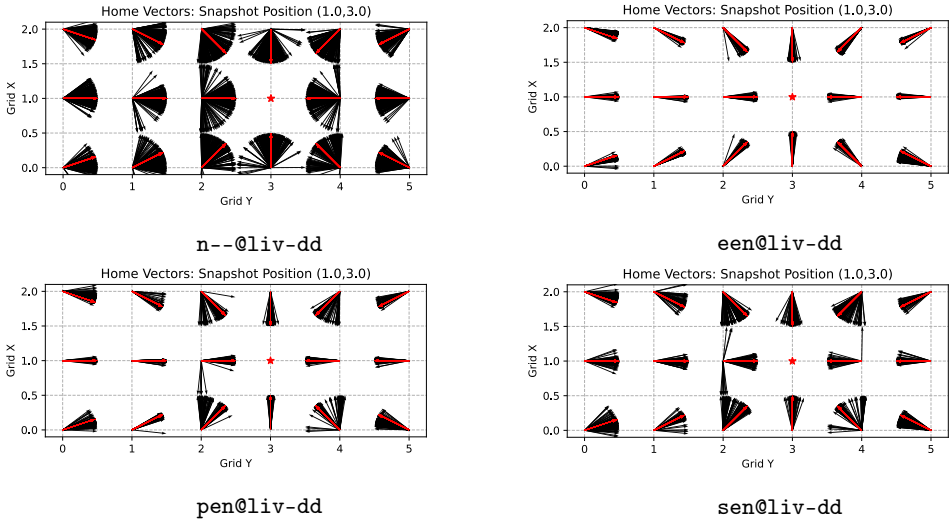


Figure 24. Home vector field plots comparing the tilt search strategies with a nearest-neighbor interpolation on the exact tilt-correction solution; one sample snapshot location from the `LIV DxD` dataset is shown. As the LIV database had 225 tilt configurations (images) per position, there are 225 potentially overlapping black arrows per position showing the estimated home directions. For this test, the snapshot position $(1, 3)$ near the middle of the grid was chosen and hence the arrows should ideally point there.

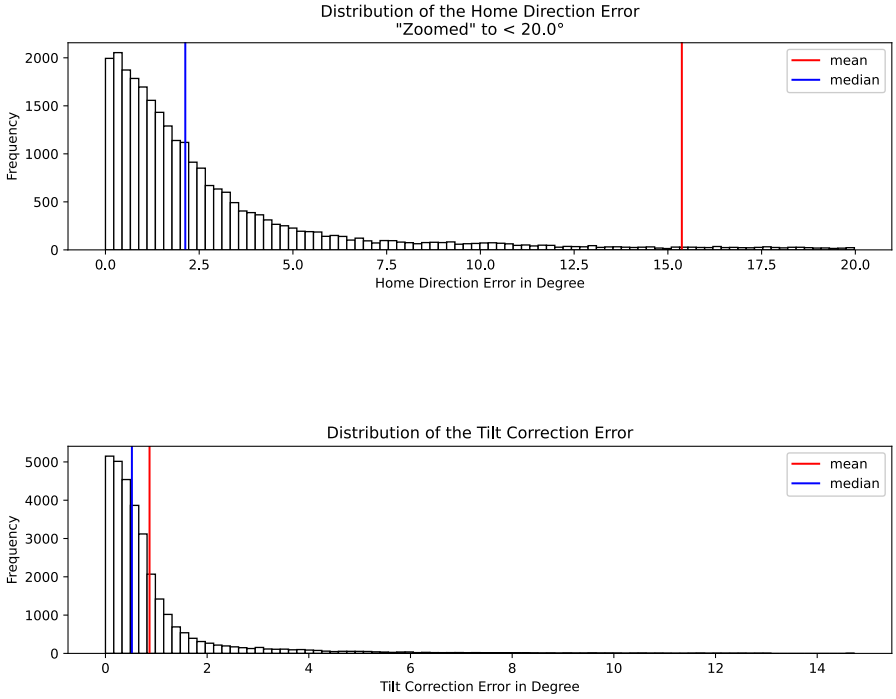


Figure 25. Distribution of the home direction error ϵ_β and tilt correction error ϵ_δ, exemplarily shown for the exhaustive tilt search using an exact tilt correction with a nearest-neighbor interpolation on the `LAB DxD` dataset (`een@lab-dd`). The median yields the better measure to summarize the homing performance, compared to the mean, as the error of the home direction tends to show a skewed distribution with a long tail of rarely occurring large errors. Though not as pronounced, the tilt correction error behaves similarly.

4.4. Tilt-Correction Performance

Similar to the homing performance, the tilt-correction performance was evaluated by comparing the angular error ϵ_δ between the true tilt vector $\mathbf{z}$ and the tilt vector $\hat{\mathbf{z}}$ estimated by the tilt correction search:

$$\epsilon_\delta = \arccos\left(\hat{\mathbf{z}}^T \mathbf{z}\right) \in [0, \pi] \tag{37}$$

These tilt vectors were computed as three-dimensional unit vectors from spherical coordinates given by the tilt parameters in the roll-pitch representation (δ_X, δ_Y) or $(\hat{\delta_X}, \hat{\delta_Y})$, respectively:

$$\mathbf{z} = \begin{pmatrix} \cos\delta_X \cos\delta_Y, & \sin\delta_X \cos\delta_Y, & \sin\delta_Y \end{pmatrix}^T \tag{38}$$

The ground-truth tilt parameters of a snapshot/current view pair were those of the current view as the snapshots were always selected with no tilt. Table 8 summarizes the quantitative results of all methods on all datasets in terms of the angular error of the tilt correction ϵ_δ. The results were averaged using the median. See again Figure 25 for a similar argument of the skewed error distribution as that for the home direction error discussed previously. In the case of no tilt correction `n--`, there was not really any angular error of the tilt correction according to Equation (37). However, when inserting a zero tilt estimate into this equation, it reduced to the magnitude of tilt described in Equation (4), which was consequently reported for all `n--` experiments. The median of this magnitude of tilt could be interpreted as the average amount of tilt which needed to be corrected in the dataset: half of the experiment samples showed images tilted less, the other half images tilted more than this value. The maximum amount of tilt was about 7° for the `LAB` and about 11° for the `LIV` database, thus the reported median was slightly above the middle of this range.

Table 8. Median of the tilt-correction error ϵ_δ in degrees. The methods are grouped by tilt (search) strategy first and then arranged by tilt correction and interpolation method. In each group, the best-performing methods per dataset, i.e., those with the smallest tilt-correction error, are highlighted in red.

	LAB DxD	LAB NxD	LAB NxN	LIV DxD	LIV NxD	LIV NxN
`n--`	4.24	4.24	0.00	6.88	6.88	0.00
`ten`	0.00	0.00	0.00	0.00	0.00	0.00
`tel`	0.00	0.00	0.00	0.00	0.00	0.00
`tan`	0.00	0.00	0.00	0.00	0.00	0.00
`tal`	0.00	0.00	0.00	0.00	0.00	0.00
`tvn`	0.00	0.00	0.00	0.00	0.00	0.00
`tvl`	0.00	0.00	0.00	0.00	0.00	0.00
`een`	**0.86**	**0.94**	**0.00**	1.15	1.79	**1.15**
`eel`	**0.86**	0.97	1.15	**1.12**	**1.60**	**1.15**
`ean`	**0.86**	**0.94**	**0.00**	1.15	1.81	**1.15**
`eal`	**0.86**	0.97	1.15	**1.12**	1.64	**1.15**
`evn`	**0.86**	0.95	**0.00**	1.22	2.03	**1.15**
`evl`	**0.86**	0.95	1.15	1.20	1.97	**1.15**
`pen`	**1.17**	**1.36**	0.00	1.57	2.50	**0.00**
`pel`	1.22	1.63	0.00	**1.49**	**2.29**	2.01
`pan`	**1.17**	1.39	0.00	1.56	2.45	**0.00**
`pal`	1.22	1.61	0.00	1.51	2.34	2.01
`pvn`	1.18	1.41	0.00	1.63	2.68	**0.00**
`pvl`	1.21	1.50	0.00	1.62	2.68	**0.00**
`sen`	**1.14**	**1.35**	0.74	1.98	2.91	1.04
`sel`	1.17	1.55	**0.71**	**1.96**	**2.72**	1.36
`san`	1.15	1.37	0.77	1.99	2.92	1.07
`sal`	1.17	1.54	**0.71**	1.99	2.77	1.35
`svn`	1.17	1.42	**0.71**	2.16	3.13	1.25
`svl`	1.16	1.44	**0.71**	2.15	3.06	**1.02**

Overall, all proposed methods were capable of actually estimating the tilt of the images. Compared to the baseline test (average amount of tilt in the dataset), an improvement of a factor between two and five was achieved. The median tilt-correction error could then be interpreted as the amount of tilt remaining in the dataset after correction. The choice of the tilt-correction solution had an almost negligible impact on the tilt-correction performance, especially, the exact and approximate solution were nearly identical. Although the vertical solution was noticeably worse, it was still within 0.5° of the other solutions.

One noteworthy observation was the variants with linear interpolation: in many cases these yielded slightly worse (or at least equivalent) tilt correction than their nearest-neighbor interpolation counterparts (although not by much). This was somewhat unexpected, as the linear interpolation was expected to produce images with an overall better quality, being beneficial to the underlying warping method and in turn making the tilt correction easier. This particular effect seemed to be more pronounced on the `LAB` database than on the `LIV` database, indicating it might be related to the content or overall structure of the images (the environment) or to the different amounts of tilt comprised in the databases, i.e., indicating that linear interpolation became beneficial only for larger amounts of tilt.

Though all search strategies were capable of finding a good tilt correction, there were more differences between the search strategies compared to those between the tilt-correction solutions. Still, they were all quite similar within 1°. Noteworthy is the pattern search strategy, which was rather consistent in correctly estimating the no-tilt condition of the `NxN` datasets. This was probably due to the initial center test point placed exactly in the middle of the search space at the no-tilt configuration, making convergence to the right tilt correction easier. The differences between the search strategies were rather independent of the chosen tilt correction, and in many cases, the interpolation was even the second

most important influence, making the choice of approximation almost insignificant to the tilt-correction performance.

Figures 26–28 visualize the search process of the different search strategies and the underlying objective function by plotting a heatmap of the function values.

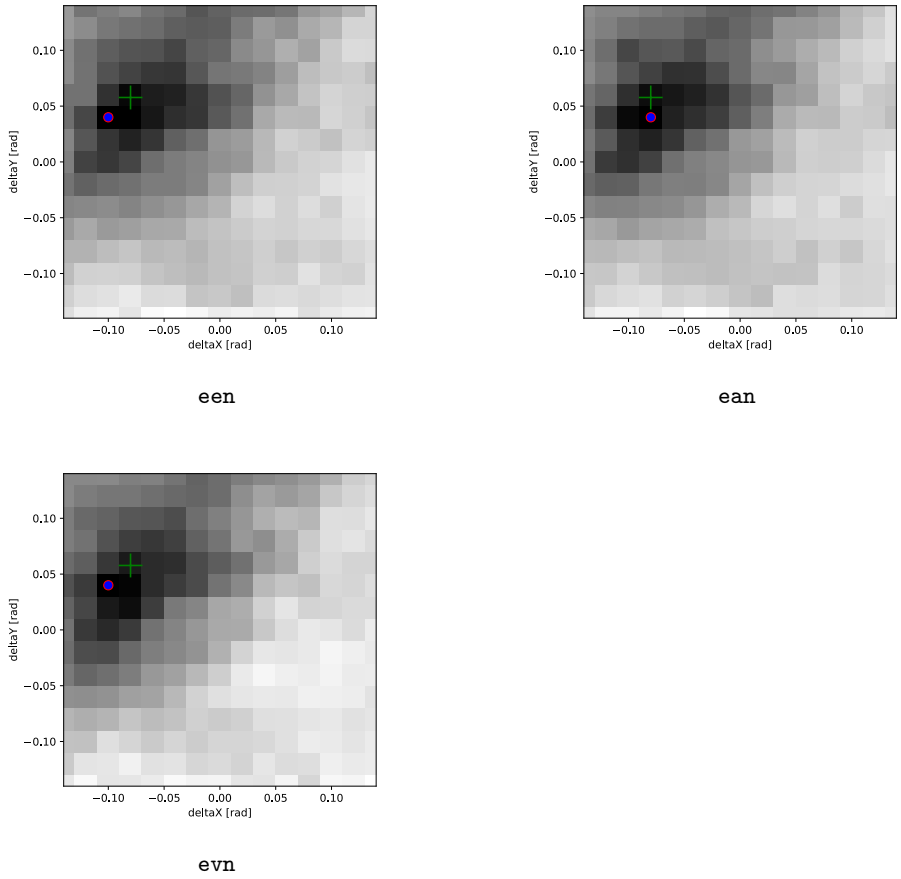


Figure 26. Visualization of the exhaustive search with the different approximations of the tilt correction and with a nearest-neighbor interpolation. For each of the tilt configurations from the exhaustive search parameter grid, the d value of the corresponding warping run, i.e., the objective function value, is plotted. The darker the pixel, the lower the d, and hence the better the match of the tilt correction. The green cross marks the location of the ground-truth tilt parameters, i.e., the expected minimum, and the red-blue circle marks the minimum found by the search procedure. For all tilt-correction solutions, good matches are clearly concentrated around the true minimum with the objective function values decreasing towards it. A pronounced valley is visible and the overall terrain is not "too bumpy", as necessary for the heuristic search methods to be applicable. The two approximations show a slightly more elongated valley of potential or near matches. In these cases, the estimated minimum of all solutions is between 1° and 2° off of the true tilt configuration, i.e., at one of the adjacent grid points. This particular sample was produced from the `LAB DxD` dataset taking the snapshot image at position $(1, 2)$ with no tilt and with an added rotation of 0.52 rad and the current view image at position $(0, 0)$ in the $(-0.07\,\text{rad}, -0.07\,\text{rad})$ tilt configuration with an added rotation of 1.41 rad. The rotation added to the current view image caused the tilt configuration to be rotated as well to the $(-0.08\,\text{rad}, 0.06\,\text{rad})$ location marked by the green cross (see Equation (31) for more details).

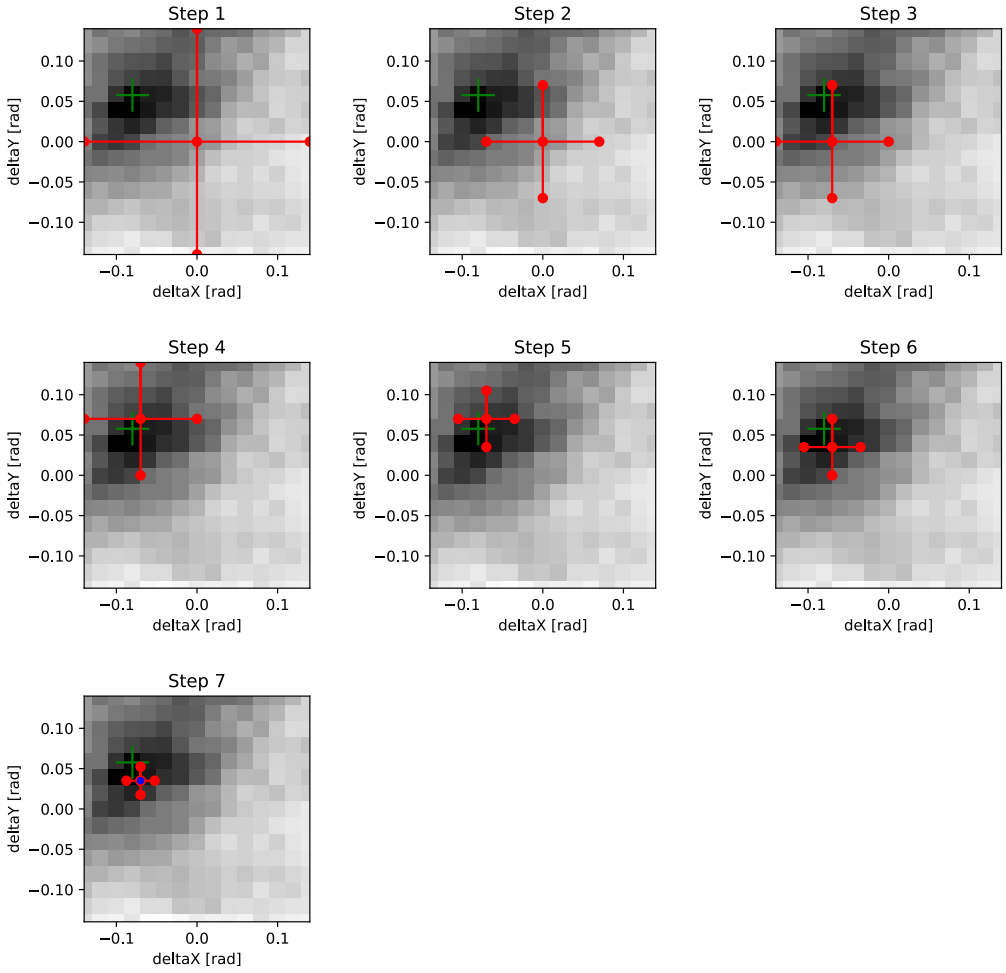


Figure 27. Visualization of the pattern search strategy `pen`. As a reference, the exhaustive search heatmap of the `een` strategy (see Figure 26 for information on the selected image pair) is plotted as the background. The green cross marks the location of the ground truth tilt parameters, i.e., the expected minimum. At each step, the state of the search pattern is visualized as a red cross connecting the test points. At the final step, the red-blue filled center circle marks the minimum found by the search procedure. After only four update steps the pattern is already centered at the top left quadrant containing the minimum. In total, only seven update steps corresponding to 22 warping evaluations are necessary to find a good approximation of the tilt correction—this is in contrast with the 225 evaluations of the exhaustive search grid. According to Table 11, this actually represents a worse-than-average performance with a mean and median number of evaluations of about 18 if tilt is present.

Just as assumed, the heatmap showed systematic variations of the warping objective d depending on the tilt parameters. As expected, the distance values d tended towards minima in the vicinity of the true tilt correction. While all strategies were capable of finding a good approximation of the true tilt up to the chosen resolution and discretization, the two heuristic search strategies did so with substantially fewer steps, i.e., evaluations of the warping objective. This, however, is investigated in more detail in the runtime analysis presented in the following section. As already inferred from the previous performance summary, the differences between the tilt-correction approximations on the objective function were minuscule and only caused a slight broadening of the valley surrounding the minimum. While this might lead to small deviations of the tilt-parameter estimate,

it probably did not result in an increase of severe errors such as shifting the minimum to another quadrant. This indicated the warping objective to be rather robust against approximations of the tilt correction. While there were some local minima, they were not as pronounced as the global minimum. With a systematic—though not necessarily continuous—decrease towards the minimum, the warping objective was well suited for the application of heuristic search strategies such as the proposed pattern and Nelder–Mead search strategies.

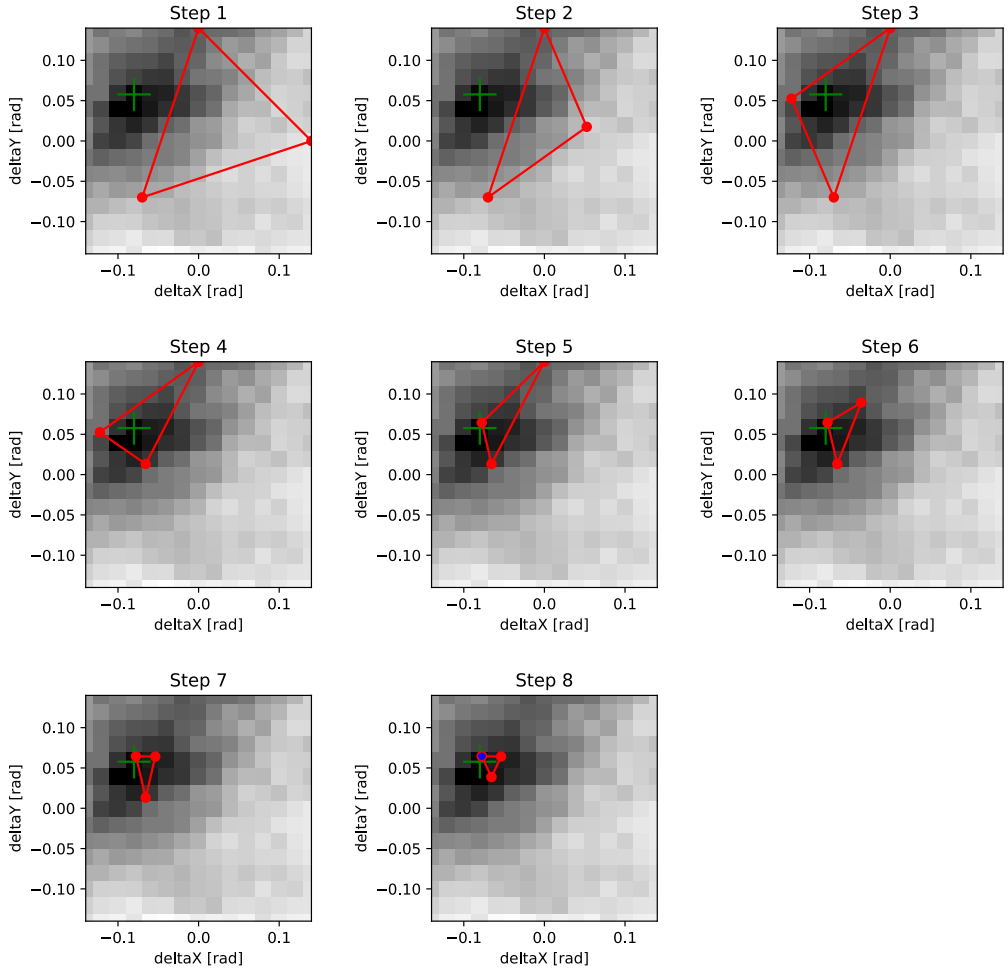


Figure 28. Visualization of the Nelder–Mead search strategy `sen`. As a reference, the exhaustive search heatmap of the `een` strategy (see Figure 26 for information on the selected image pair) is plotted as the background. The green cross marks the location of the ground truth tilt parameters, i.e., the expected minimum. At each step, the state of the search simplex is visualized as a red triangle connecting the test points. At the final step, the red-blue filled circle marks the minimum found by the search procedure. After only three update steps, the minimum already lies inside the simplex. In total, only eight update steps corresponding to 13 warping evaluations are necessary to find a good approximation of the tilt correction—this is in contrast with the 225 evaluations of the exhaustive search grid. Note that most of the update steps were contractions, requiring two evaluations of the warping objective. However, in steps 2, 4, and 6 the reflected point, which was computed before the contraction, would be outside of the search range causing the warping evaluation to be skipped, reducing the number of evaluations to one. According to Table 11, this actually represents a slightly better-than-average performance with respect to the median number of evaluations of 15 (and a mean number of 30), if tilt is present.

4.5. Runtime Performance

To assess the practical usefulness of the proposed methods, a runtime analysis of the implementation was conducted. This was done by measuring the execution time per snapshot/current view image pair at different points of the process, i.e., including/excluding the search process. For the three tilt-correction search strategies, the number of warping invocations necessary to find the optimum was counted, as this was expected to be the dominating factor influencing the total runtime. All algorithms were implemented using C++ within the framework of `WarpingSIMD` [23,24]. The experiments were compiled with optimizations enabled and executed on an `Intel(R) Core(TM) i9-10980XE CPU @ 3.00GHz` using `AVX-512` for warping and most image operations. Multithreading was only used to parallelize experiment instances, not the algorithms itself.

Before investigating the runtime of a tilt-correction search in combination with warping, Table 9 shows the standalone runtimes of the tilt-correction and interpolation methods, when applying a ground-truth tilt correction.

Table 9. Average runtime of the tilt-correction methods in milliseconds. Both the mean and the median are reported, but they only show small differences, indicating only slight variation or outliers. The average runtimes were pooled over all datasets as there was no dependence on the *content* of the images to be expected; the determining factors for the runtime were the choice of method and the *size* of the image, i.e., the number of pixels to transform, which was the same for all datasets. Note, however, that these were runtimes for **single** tilt correction runs and there might be significant speed-ups when doing **repeated** tilt corrections within the same run of the test program; see Table 10 for a comparison of this effect in the case of the warping algorithm.

	Mean	**Median**
`en`	3.41	3.36
`el`	4.24	4.15
`an`	1.03	0.99
`al`	1.81	1.76
`vn`	0.32	0.31
`vl`	0.72	0.71

The nearest-neighbor interpolation was much faster: in the case of the vertical approximation, up to two times as fast as the linear interpolation. This was expected, as the linear interpolation required four source image pixels surrounding the target pixel and some arithmetic (including multiplications) to compute a weighted sum of these values, while the nearest-neighbor interpolation required only a rounding of the pixel coordinates for copying one pixel of the source image. As expected, the stronger the approximation, the faster the image transformations were computed. This was achieved by reducing the number of trigonometric operations and multiplications, which contributed the most significant speedup: the approximate solution was about three times faster compared to the exact solution with a nearest-neighbor interpolation. Furthermore, the vertical solution allowed the computations to be reduced to one per image column instead of one per pixel. This contributed another speedup factor of roughly two to three, going from approximate to vertical (with the interpolation kept constant). Going from the most accurate solution with the linear interpolation `el` to the most simplified with the nearest-neighbor interpolation `vn`, these approximations, in total, yielded a speedup factor of about 13. However, with runtimes ranging between 4 ms and 0.3 ms, all tilt corrections operated an order of magnitude faster than the warping algorithm with runtimes of about 23 ms to 28 ms, as shown in Table 10. Thus, the runtimes of the tilt correction search methods were definitely dominated by the runtime of warping: saving a single warping evaluation was worth between 5 and 76 tilt corrections, depending on the chosen approximation.

Table 10. Comparison of the average raw warping runtime per snapshot/current view pair in milliseconds when doing only a single vs. repeated executions of the algorithm within the same run of the test program. These are average runtimes pooled over all experiments, i.e., **Single** summarizes the `n--` and `t**` methods while **Repeated** summarizes all search strategies `e**`, `p**`, and `s**`. There is a noticeable speed-up of about 4 ms when doing repeated warping runs, which is probably due to cache or branch-prediction influences only effective after multiple passes through the same portion of code or image data. This effect needs to be considered when comparing the runtimes of the search strategies which rely on repeated warping runs to those doing only a single warping run, i.e., the `t**` methods.

Single		**Repeated**	
Mean	**Median**	**Mean**	**Median**
27.2	27.7	23.7	23.0

Furthermore, thorough measurements of these warping runtimes revealed noticeable variations of the warping runtime of about 4 ms (which was on the order of magnitude of the tilt corrections), when doing repeated evaluations of the algorithm within the same run of the test program. This effect, probably due to caching and branch-prediction speed-ups, needed to be considered when comparing tilt-correction schemes with different numbers of warping evaluations in the following.

Tables 11 and 12 list the average runtimes off all method combinations and the overall speed-ups which could be achieved when using one of the heuristic search strategies instead of the exhaustive search, respectively. The results were pooled over datasets containing tilt (`DxD` and `NxD`) and those containing no tilt (`NxN`) from both databases, as there should be noticeable differences between these situations for the heuristic search strategies: the heuristic search strategies should be able to converge more quickly towards a no-tilt configuration, which lay in the center of the initial search pattern. This could indeed be observed, for both the pattern and Nelder–Mead search strategies. For both, the runtime and the warping count, the mean as well as the median were reported and generally expected to be rather similar, i.e., reflecting uniformly or even normally distributed runtimes. This was due to the runtime of most of the components being determined by the size of the images and number of parameters to be probed and not by some particularities of individual image pairs.

However, in the case of the heuristic search strategies, the number of warping evaluations actually became variable, depending on how well the search converged towards the minimum. This number then might actually depend on particularities such as the location of the minimum or the smoothness of the objective function values produced by warping. In these cases, the differences between mean and median might actually yield some interesting insights regarding the behavior of the search strategies: for the pattern search, there were almost no differences between the mean and median runtime as well as warping counts. Thus, the pattern search probably behaved consistently across all samples from the test dataset and seemed to be rather robust against outliers and difficult/extreme situations such as strong tilt or noisy warping objectives. The Nelder–Mead search on the other hand showed huge differences between the mean and median runtime, which were also reflected in the warping counts. The mean runtime of the Nelder–Mead search was almost twice as big as the median and there were more then twice as many warping evaluations necessary in the mean than median. This indicated a skewed distribution of runtimes or warping counts with at least half of the samples performing rather well—actually running faster than the pattern search—and probably few samples with very long runtimes. These long runtimes might even correspond to search processes failing to converge at all, running into the hard implementation limit of 50 steps stopping the test program from running endlessly. This might have been caused by the rather difficult initialization of the three-point search simplex covering a square search space in contrast to the almost trivial and effective initialization of the five-point, symmetric search pattern.

Table 11. Average runtime in milliseconds and average count of warping runs per snapshot–current view pair. The results are pooled by datasets containing tilt and containing no tilt. Both the mean and the median are reported and are rather similar for most of the methods, i.e., not more than one warping runtime apart. Only the Nelder–Mead search strategy shows a huge discrepancy between the mean and median runtime, which is also reflected in the warping counts. The methods are grouped by search strategy first and then arranged by tilt correction and interpolation method. In each group, the best-performing methods per column, i.e., those with the smallest runtime, are highlighted in red. Except for the Nelder–Mead strategy, the methods in each group behave consistently with `*vn`, i.e., the most optimized, being the fastest in general. The `n--`, `t**`, and `e**` methods have the count of warping runs fixed by design, i.e., 225 is the size of the exhaustive search grid.

	Time				**Count**			
	Tilt		**No Tilt**		**Tilt**		**No Tilt**	
	Mean	**Median**	**Mean**	**Median**	**Mean**	**Median**	**Mean**	**Median**
`n--`	27.5	28.1	27.3	27.9		1.0		
`ten`	30.4	31.0	30.5	30.9				
`tel`	31.3	31.9	31.6	31.9				
`tan`	28.1	28.6	28.6	28.8		1.0		
`tal`	28.9	29.5	29.2	29.7				
`tvn`	**27.8**	**28.3**	**27.5**	**28.3**				
`tvl`	28.5	29.1	29.0	29.2				
`een`	5632.0	5573.5	5528.7	5484.4				
`eel`	5773.6	5711.6	5672.9	5647.2				
`ean`	5269.6	5217.5	5178.6	5142.6		225.0		
`eal`	5401.9	5347.6	5291.4	5269.3				
`evn`	**5197.9**	**5147.0**	**5105.5**	**5072.7**				
`evl`	5311.1	5258.3	5215.1	5173.1				
`pen`	486.3	477.4	376.9	364.9	18.1	**17.0**	**14.1**	**13.0**
`pel`	509.3	497.0	416.5	395.7	18.5	18.0	15.2	**13.0**
`pan`	453.3	444.7	356.0	**345.1**	**18.0**	**17.0**	14.2	**13.0**
`pal`	474.7	463.2	390.3	372.9	18.5	18.0	15.1	**13.0**
`pvn`	**447.5**	**437.6**	**354.7**	346.9	**18.0**	**17.0**	14.3	**13.0**
`pvl`	466.9	455.7	386.8	367.0	18.5	18.0	15.1	**13.0**
`sen`	794.3	409.2	**531.5**	367.0	**30.3**	**15.0**	**19.9**	**13.0**
`sel`	957.2	430.1	771.8	380.1	35.8	**15.0**	29.2	14.0
`san`	**777.8**	**384.5**	591.6	342.4	31.8	**15.0**	24.3	14.0
`sal`	910.2	403.7	739.5	361.6	36.4	**15.0**	29.9	14.0
`svn`	873.5	387.9	618.5	**338.1**	36.6	**15.0**	26.1	14.0
`svl`	923.3	401.5	675.3	343.5	37.6	**15.0**	27.6	**13.0**

Relating the average number of warping evaluations to the total runtime and the known runtime of a single warping run (see Table 10) confirmed the expectation of the number of warping evaluations being the determining factor of the total runtime with approximation and interpolation being only secondary factors almost not affecting the warping count at all. To summarize, a speed-up of 6 to 17 could be expected by switching from an exhaustive search to one of the heuristic strategies, placing the runtime at a few hundred milliseconds instead of the more than five seconds of the exhaustive strategies. While these were clearly impressive numbers, a thorough analysis of the trade-offs between runtime and homing performance as well as tilt-correction performance was necessary to decide which methods were indeed viable improvements and which just lost too much navigation performance for too little runtime gains. This trade-off analysis is presented in Figures 29 and 30, separately for the two databases, as the viability of a method might actually depend on the difficulty of the environment or the amount of tilt under consideration.

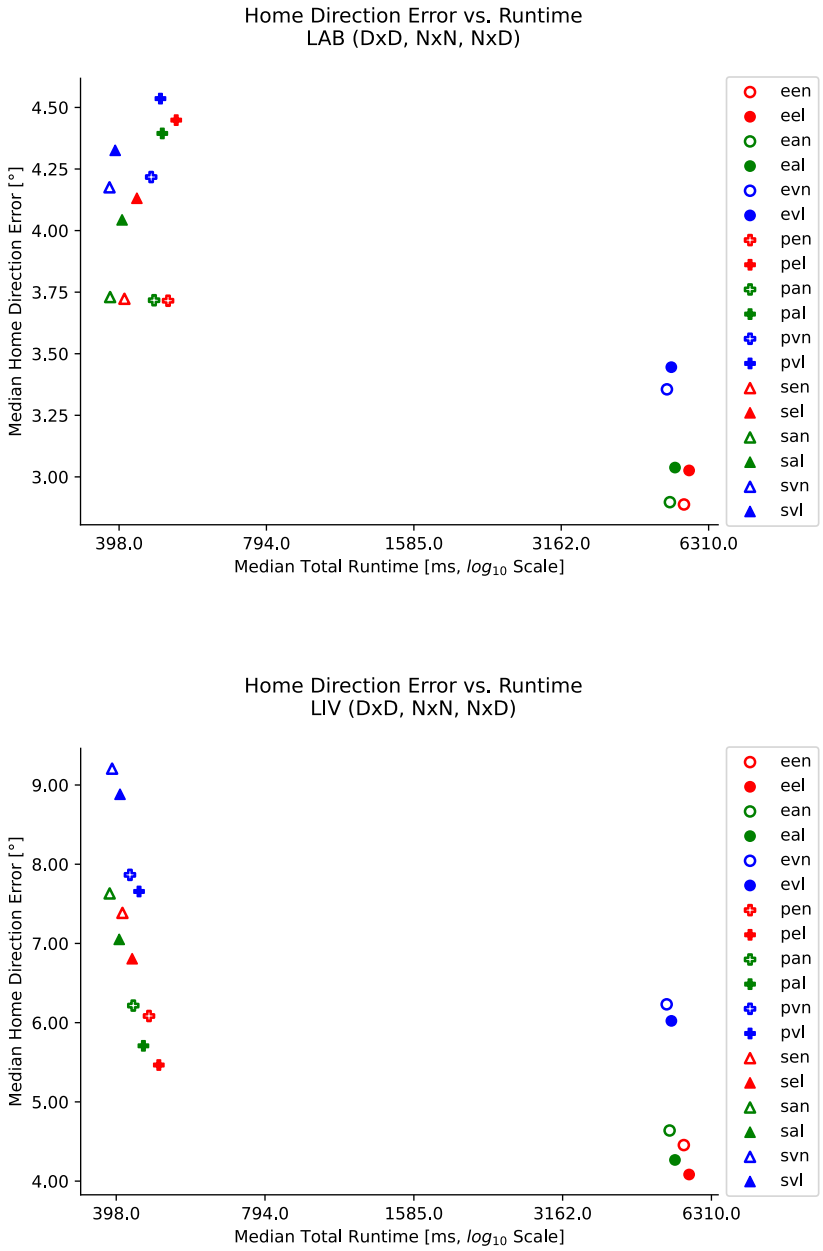


Figure 29. Runtime vs. home-direction error of all tilt-correction search method variants pooled over all datasets per `LAB`/`LIV` database. Both metrics are summarized using the median, which in case of the home direction is definitely the better measure as discussed in Figure 25. For the runtime, the simplex strategies show a difference between median and mean. According to the latter, the simplex strategies would be placed slightly behind the pattern search in terms of runtime. However, the median was chosen for the runtime as well, to treat both metrics consistently. In terms of runtime, there is not much difference between the two databases, and the difference in home-direction errors probably stems from the increased range of tilt and the more difficult environment of the LIV database. The methods form two broad clusters: the fast but more error-prone heuristic search strategies clustering around 400 ms and the slow but precise exhaustive search between 5 s to 6 s. The effects of the choice of the tilt-correction method are subordinate and the interpolation is behaving rather inconsistently between the two databases.

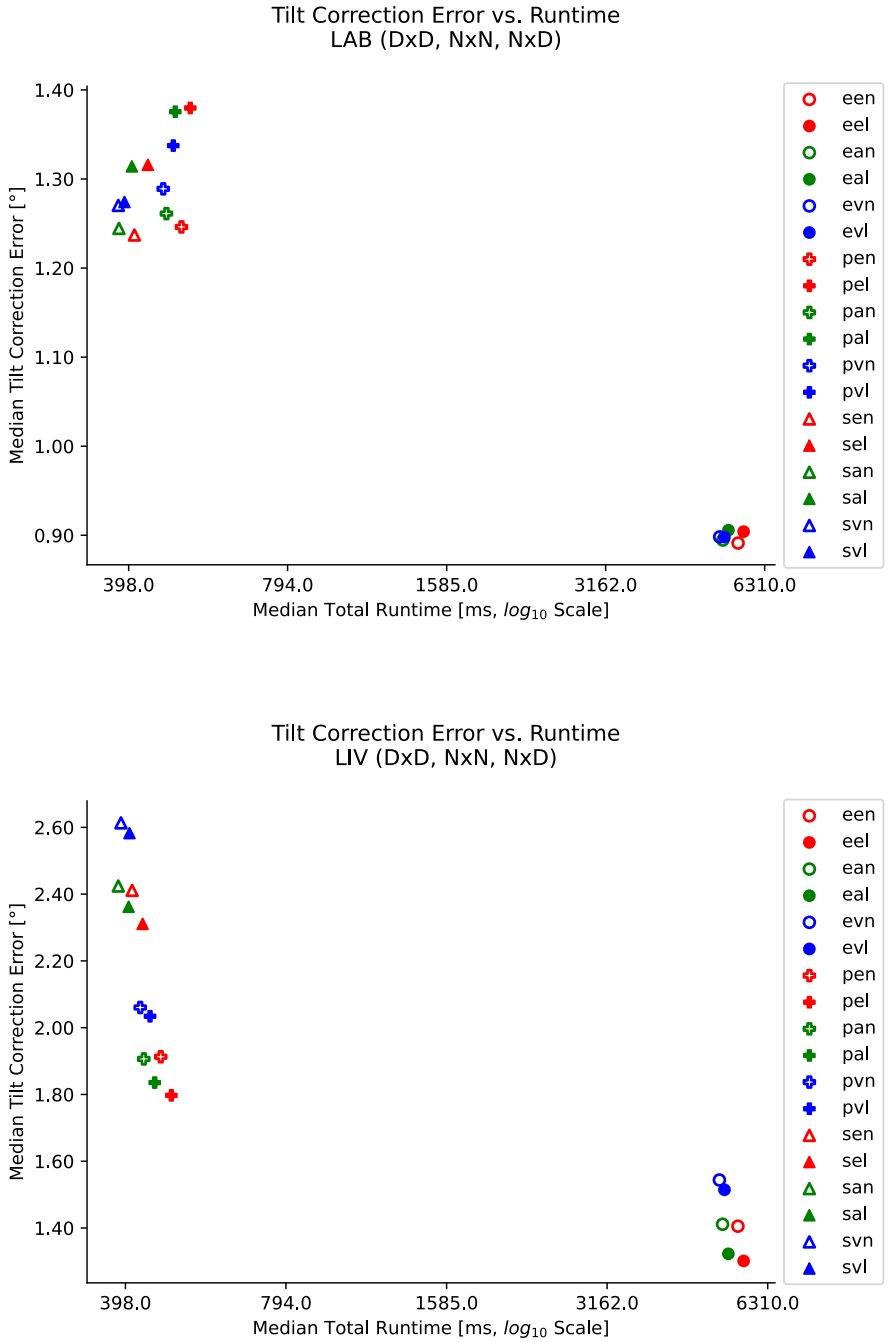


Figure 30. Runtime vs. tilt-correction error of all tilt-correction search method variants pooled per database. Similar to Figure 29, both metrics are summarized using the median instead of the mean. Again, the different error ranges between the databases probably stem from the increased range of tilt and the more difficult environment of the LIV database. Similar to the homing performance, the methods form the two clusters of heuristic and exhaustive search strategies with the tilt-correction methods being the secondary effect and the effect of the interpolation switching between the two databases.

Table 12. Relative speed-ups of the search strategies compared to the slowest strategy, i.e., the exhaustive search. For each strategy, the runtime of the fastest method from its group per column of Table 11 is related to the runtime of the fastest exhaustive search `e**` in that column. The `n--` and `t**` strategies are not reported as they do not search for the tilt correction and thus are necessarily much faster (about 180 times). As already observed in the raw runtimes, the mean speed-up of the Nelder–Mead search is surprisingly low compared to its median and to the pattern search according to both metrics.

	Time				**Count**			
	Tilt		**No Tilt**		**Tilt**		**No Tilt**	
	Mean	**Median**	**Mean**	**Median**	**Mean**	**Median**	**Mean**	**Median**
`e**`	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
`p**`	11.6	11.8	14.4	14.7	12.5	13.2	16.0	17.3
`s**`	6.7	13.4	9.6	15.0	7.4	15.0	11.3	17.3

5. Discussion

The results showed that all proposed methods—using different solutions for transforming panoramic images according to tilt as well as different search strategies—were indeed capable of vastly improving the homing performance of warping and simultaneously estimating the tilt of the camera. As expected, the homing performance of the original warping method, which was derived under the assumption of a robot moving on flat ground, was already severely impaired by small deviations; 11°, the maximum tilt angle contained in the test databases, is actually a small value in outdoor environments. The proposed tilt-correction scheme followed the same general idea as the warping method itself: formulate the transformation, i.e., warping, of the panoramic image according to hypothetical movement parameters and then do a systematic search through the set of potential parameters by optimizing a criterion based on an image distance measure. As warping already computes such a distance measure, it seemed natural to simply reuse it by wrapping the warping algorithm in additional search loops covering the two new tilt parameters.

5.1. Runtime Cost and Search Strategies

While an exhaustive search through the original warping parameters and the tilt parameters proved to be successful, the naive approach of doing an exhaustive search over the tilt parameters, wrapping the already exhaustive search of the warping algorithm, comes at a huge runtime cost. Doing an exhaustive search, the total computation time for a single home direction estimate, i.e., a single snapshot–current view image pair, was about 5 s on the hardware used for the experiments, which is rather impractical for any real application (In particular when considering the processor used for the experiments, which was a workstation processor and thus rather uncommon for the intended embedded or robotics setting). This runtime is the accumulated warping runtime which, in case of the exhaustive search, scales quadratically with the desired estimation quality. However, as the image distance measure provided by warping showed to be *well-behaved* enough (see below) to consider the application of direct and heuristic search strategies, it is possible to bring the runtime down to a more practical few hundred milliseconds at the cost of estimation quality by using the implemented pattern search and Nelder–Mead strategies. These generally find a good estimate of the tilt parameters, as well as the warp parameters, while only requiring a fraction of the warping evaluations compared to the exhaustive search. Here, *well-behaved* means the image distance varies systematically and almost smoothly depending on the tilt with a global minimum near the true tilt configuration surrounded by a wide and pronounced valley without too many distracting local minima. This good behavior of the optimization criterion motivates us to look into other direct and heuristic search strategies to further reduce the number of warping evaluations. In particular, strategies based on randomized test points surrounding the candidate solution, for example the Luus–Jaakola heuristic search [25], seem to be promising since they are simple and require only a single

test point per iteration (though the number of iterations required needs to be explored). Furthermore, it might even be possible to apply gradient-based optimization schemes if warping can either be formulated to be differentiable with respect to the tilt parameters, or it is at least possible to numerically estimate the gradient of the image distance from only a few warping runs, i.e., test points.

5.2. Mitigating the Runtime Costs

As the runtime was clearly dominated by the search process and in particular by the runtime of the warping-based optimization criterion, future research should focus on these aspects first, before looking at the more nuanced details of the tilt-correcting image transformations or thinking about extensions to even more movement parameters. The approaches for tackling the runtime problem of the proposed tilt-correction scheme fall into three broad categories: search and initialization strategies, reduction of the warping runtime, or alternatives to the warping-based optimization criterion.

5.2.1. Search and Initialization Strategies

First, as already mentioned, other heuristic search strategies might reduce the number of warping evaluations necessary to find the optimum or at least reduce the complexity of the overall search process. In this context, the initialization of the heuristic search patterns, which was more or less disregarded here by using reasonable defaults derived from the exhaustive search grid, is definitely worth more investigations. Though not further examined here, the noticeable difference between the mean and median runtime of the Nelder–Mead strategy can probably be attributed to the more difficult initialization of the three-point search simplex compared to the five-point pattern of the pattern search. These initialization difficulties need to be sorted out to make the Nelder–Mead strategy viable in general. Alternatively, future research should focus either on heuristic strategies with symmetric search patterns, as in the five-point pattern search, or those without any special initialization, such as the already mentioned, randomized Luus–Jaakola heuristic search.

5.2.2. Reduction of the Warping Runtime

The second approach to the runtime problem involves reducing the runtime of the warping-based optimization criterion. This can be achieved by reducing the size of the images or the resolution of the warp parameter space, which is searched exhaustively by warping. While this will definitely impair the homing performance, it could still allow for a sufficiently good estimation of the tilt at reduced runtime costs. Employing a coarse-to-fine strategy, it might then be possible to refine this initial estimate of the tilt and warp parameters by only a few additional (costly) evaluations using the full size and resolution on a small and constrained tilt parameter range. When using such a coarse-to-fine scheme, it might also be possible to combine different search strategies throughout different levels of coarseness; for example, a coarse exhaustive search followed by a fine heuristic (e.g., pattern) search or vice versa.

Another approach to reduce the overall time spent evaluating the warping objective is to simply run multiple warping instances in parallel, e.g., evaluate all five points of the pattern search at once using multithreading. In this way, the total runtime could be sped up roughly by the number of processor cores available. This however, requires a target platform which actually provides multiple processor cores or at least some form of simultaneous multithreading, which in the context of resource-restricted embedded or robotics platforms is not necessarily the case.

5.2.3. Alternative Optimization Criteria

Finally, the third approach is to get rid of the warping-based objective altogether, i.e., just remove the costliest operation. While this obviously does not allow to estimate the home direction simultaneously and thus would require a single, additional warping evaluation afterwards, finding a faster way to compute an image distance measure or

at least some rough estimate thereof has the potential to significantly speed up the tilt estimation. However, such a distance measure probably requires the images to be aligned first using some external compass, which in the case of warping is already part of the method itself.

One idea avoiding an additional compass is to still utilize the first phase of warping, i.e., the columnwise distance image which can be computed efficiently in just a fraction of the total warping runtime. These distance images should show an increased number of matching image columns if they are brought into alignment by tilt correction first. If it is somehow possible to uncover the number of these matches using only simple operations, e.g., some statistics over the distance values, this could serve as an estimator for the image matching without requiring the full search through the warp parameters. Using the tilt correction implemented within the warping framework and the two newly collected image databases, it should now be possible to systematically produce enough sample distance images under varying tilt to experimentally search for such an estimator.

5.3. Approximations of Tilt-Correction Solutions

Regarding the proposed approximations (and the chosen interpolation method as well) of the tilt-correction solutions of panoramic images, the results were rather inconclusive. While the trade-off between the runtime and the correction quality was clearly evident and behaved as expected, the runtime was at least an order of magnitude below that of the dominating warping runtime. Thus, even the accumulated runtime of multiple tilt corrections was still almost negligible relative to the whole search process. That means that before figuring out a faster optimization scheme as discussed before, there is no point in considering the runtime gains by approximating the tilt correction. Only in the case of known tilt-correction parameters, for example measured by an inertial measurement unit, when only a single warping evaluation is done, might a more inaccurate solution yield a considerable runtime improvement.

If the revision of the optimization scheme based on statistics over the distance images of the first warping phase proposed earlier succeeds, the runtime of this estimator should be on the same order of magnitude as the tilt correction. At that point the runtime–quality trade-off will indeed become an important factor to consider. In that case, the approximate solutions `an` and `al` are probably the most reasonable compromise as the homing and tilt-correction performance was very similar to the more costly exact solution and thus the runtime improvement by a factor of about three (counting only the runtime per image transformation) was almost free in terms of cost of estimation quality. The vertical solution, i.e., the strongest approximation, performed noticeably worse compared to the other solutions and thus cannot be recommended as a runtime-saving alternative at the moment. This is somewhat unfortunate, as the simplicity of handling the tilt only within the same image column opens up a lot of possibilities for optimizing the implementation in terms of computation as well as memory efficiency; for example, instead of computing two new indices per pixel, only one offset per image column is required. This is not just much faster to compute (as proven by the results) but also small enough to consider precomputing an offset table for all possible tilt configurations. However, as long as the homing and tilt-correction performance of the vertical solution cannot be improved—and this might actually be an inherent limitation of neglecting the horizontal component of tilt—these practical optimizations are not worth any effort.

5.4. Extension to More Difficult Scenarios

The proposed tilt-correction methods were derived under some simplifying assumptions: the tilt was assumed to be a small out-of-plane rotation of only one of the images, i.e., the snapshot always served as an absolute reference orientation and was assumed to be not tilted. Thus, the methods should further be tested in situations where these assumptions are violated. Of particular interest is an extension of the methods to handle both images tilted relative to a third reference orientation. While it should be straightforward to simply feed

tilted snapshot images into the existing implementation as well, deriving a tilt estimation for both images is probably not as simple as it might seem: there is an infinite number of solutions bringing two tilted planes back into correspondence. That means a reasonable constraint is necessary to select one of these solutions. Furthermore, tilting a second image requires the introduction of two additional tilt parameters, resulting in the search space of the already impractically slow exhaustive search now growing to the fourth power of the search resolution. Thus, the extension to a two-image tilt can practically only be solved by heuristic search strategies. Finally, it is not really clear how to estimate absolute tilt parameters per image from such a relative tilt correction or whether this is even necessary for a local navigation method.

An extension to large amounts of tilt, however, should already be doable with the existing implementation: a large tilt merely invalidates the assumptions leading to the approximate and vertical solution and thus should be handled by the exact solution. The final limit on the range of tilt is then the vertical opening angle of the panoramic image, which is related to the field of view of the camera/lens and the image cropping during preprocessing. Exceeding this limit will result in tilting whole image columns out of sight with only invalid pixels remaining at which point the column matching of the warping algorithm probably starts to break down. However, testing these claims requires the collection of additional databases, as the two introduced here comprise only small amounts of tilt up to 11°. Furthermore, increasing the amount of tilt also requires increasing the tilt parameter search range. If the estimation quality, i.e., the resolution of the search space in case of the exhaustive search or the convergence threshold in the case of the pattern and Nelder–Mead search strategies, is kept constant, this inevitably results in an increased runtime as well.

6. Conclusions

Adding a tilt-correction phase to only one of the input images of warping leads to vast improvements of the homing performance under moderate amounts of camera tilt, as is typical for indoor environments. If the tilt parameters are known, for example by measurements of an additional sensor, these improvements are almost free in terms of runtime overhead, as the added runtime for a single image transformation is almost negligible compared to the warping runtime. If no tilt parameters are known beforehand, warping can be interpreted as an image distance measure and coupled to the tilt correction via a search over hypothetical tilt parameters. This yields improvements of the homing performance while at the same time providing good estimates of the relative tilt between the snapshot and current view.

Thus, the warping framework can be extended to not only estimate the planar movement parameters of the robot but also two additional parameters for a small out-of-plane tilt, without the need for additional sensors or external image alignment. This is a first step towards extending warping to a full pose estimation between two camera locations, which additionally would include tilting the camera at both locations, larger amounts of tilt, and possibly vertical translation out of the plane. As this extension comes at a huge runtime cost, only the proposed heuristic search strategies for tilt correction, i.e., the pattern and the Nelder–Mead search strategies, can be recommended for practical application in visual robot navigation. These are about 10 times faster compared to the exhaustive search at only moderate trade-offs in terms of homing performance and tilt estimation. The prohibitively slow exhaustive strategy, however, will remain relevant in more theoretical considerations, for example when further analyzing the systematic variations of the warping objective with respect to the tilt parameters.

To the best of our knowledge, this is the second approach to handle tilt in a warping-based holistic homing method. Compared with the previous approach described in [11], we directly used the input images instead of performing computations in the frequency domain by wrapping the already established warping method *MinWarping*. However, in [11], a more general solution was proposed for arbitrary 3D movements. Though it is conceivable

to extend our method to this more general case as well, this is currently limited by the almost prohibitively huge runtime cost of the parameter search, as discussed previously.

Author Contributions: All authors conceived and designed the experiments. C.B. added the vertical approximation, developed and implemented the method, suggested and implemented the application of heuristic search strategies, added extensions to MinWarping, collected and prepared the image databases, performed the experiments, analyzed the results, and wrote the paper. A.H. added extensions to MinWarping, contributed the handling of invalid image regions and the unrolling of images into a panoramic representation, and helped calibrate the camera. R.M. supervised the project, proposed the initial idea, derived the image transformations, and provided the MinWarping implementation. All authors have read and agreed to the published version of the manuscript.

Funding: We acknowledge support for the publication costs by the Open Access Publication Fund of Bielefeld University and the Deutsche Forschungsgemeinschaft (DFG).

Data Availability Statement: The image databases and code for reproducing the experimental results are available from the website of the Computer Engineering Group of Bielefeld University under the following URL : http://www.ti.uni-bielefeld.de/html/people/ahoffmann/tiltdb.html (accessed on 23 January 2023). The implementation of MinWarping the tilt correction image transformations are available from the website of the Computer Engineering Group of Bielefeld University under the following URL: http://www.ti.uni-bielefeld.de/html/people/moeller/tsimd_warpingsimd.html (accessed on 23 January 2023).

Acknowledgments: The authors thank Klaus Kulitza, Bielefeld University, for designing the flipped/roll-pitch configuration of the pan–tilt unit.

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

Appendix A. Derivations

Appendix A.1. Conversion between Axis-Angle and Roll-Pitch Representations

As described by Equation (1), the roll-pitch representation can be expressed by the composition of the two basic rotation matrices about the X- and Y-axis. To construct a conversion to the axis-angle representation, it is necessary to express this in terms of rotation matrices as well. As described by Equation (2), the axis-angle representation expresses the tilt as a rotation of φ about an axis which is described relative to the X-axis by rotating Θ_R about the Z-axis. A third parameter, Θ'_R, is introduced to guarantee the existence of exact solutions—this can be interpreted as reverting the effect of the first Z-axis rotation, as the tilt is assumed to introduce no change in the view direction of the robot:

$$\mathrm{R}_X(\delta_X)\mathrm{R}_Y(\delta_Y) \stackrel{!}{=} \mathrm{R}_Z(\Theta_R)\mathrm{R}_X(\varphi)\mathrm{R}_Z\left(\Theta'_R\right)$$

This matrix equation can be solved by matching suitable sets of entries with only one of the axis-angle parameters appearing on the right-hand side:

$$\begin{bmatrix} c_{\delta_Y} & 0 & s_{\delta_Y} \\ s_{\delta_X} s_{\delta_Y} & c_{\delta_X} & -s_{\delta_X} c_{\delta_Y} \\ -c_{\delta_X} s_{\delta_Y} & s_{\delta_X} & c_{\delta_X} c_{\delta_Y} \end{bmatrix} = \begin{bmatrix} c_{\Theta_R} c_{\Theta'_R} - s_{\Theta_R} c_\varphi s_{\Theta'_R} & -c_{\Theta_R} s_{\Theta'_R} - s_{\Theta_R} c_\varphi c_{\Theta'_R} & s_{\Theta_R} s_\varphi \\ s_{\Theta_R} c_{\Theta'_R} + c_{\Theta_R} c_\varphi s_{\Theta'_R} & -s_{\Theta_R} s_{\Theta'_R} + c_{\Theta_R} c_\varphi c_{\Theta'_R} & -c_{\Theta_R} s_\varphi \\ s_\varphi s_{\Theta'_R} & s_\varphi c_{\Theta'_R} & c_\varphi \end{bmatrix}$$

Assuming the tilt angle φ to be positive and small (allowing us to eliminate s_φ), the solutions for Θ_R, φ, and Θ'_R can be obtained from the entries at $[1,3]/[2,3]$, $[3,3]$, and $[3,1]/[3,2]$:

$$\Theta_R = \operatorname{atan2}(\sin(\delta_Y), \sin(\delta_X)\cos(\delta_Y)) \tag{A1}$$

$$\varphi = \arccos(\cos(\delta_X)\cos(\delta_Y)) \tag{A2}$$

$$\Theta'_R = \operatorname{atan2}(-\cos(\delta_X)\sin(\delta_Y), \sin(\delta_X)) \tag{A3}$$

By assuming only small tilt angles δ_X and δ_Y, Equations (A1) and (A3) can be simplified, and it is possible to express Θ'_R in terms of Θ_R getting rid of the additional parameter:

$$\Theta_R \approx \text{atan2}(\delta_Y, \delta_X) \tag{A4}$$

$$\Theta'_R \approx \text{atan2}(-\delta_Y, \delta_X) \approx -\Theta_R \tag{A5}$$

Thus, the axis-angle representation can be seen as an X-axis rotation undergoing a similarity transform to align the coordinate system with the tilt axis:

$$\text{Axis-Angle} \triangleq \mathrm{R}_Z(\Theta_R)\mathrm{R}_X(\varphi)\mathrm{R}_Z(\Theta_R)^{-1} \approx \mathrm{R}_X(\delta_X)\mathrm{R}_Y(\delta_Y) \tag{A6}$$

Conversions from axis-angle to roll-pitch representation can be obtained from the entries at $[2,3]/[3,3]$ and $[1,3]$, again assuming the previous small angle approximations:

$$\delta_X = \text{atan2}(\cos(\Theta_R)\sin(\varphi), \cos(\varphi)) \approx \varphi\cos(\Theta_R) \tag{A7}$$

$$\delta_Y = \arcsin(\sin(\Theta_R)\sin(\varphi)) \approx \varphi\sin(\Theta_R) \tag{A8}$$

Appendix A.2. Adaptation of Tilt Parameters under Rotation

Starting from a representation $X-Y-Z$ where a tilt (δ_X, δ_Y) is applied first before rotating the image by δ_R about the tilted Z-axis, a parametrization $(\delta'_R, \delta^*_X, \delta^*_Y)$ describing the same tilt configuration where the robot tilts in the already-rotated coordinate frame is sought. Note that the order of the tilt axes is kept the same, treating the tilt as the same atomic operation according to the introduced roll-pitch representation:

$$\mathrm{R}_X(\delta_X)\mathrm{R}_Y(\delta_Y)\mathrm{R}_Z(\delta_R) \overset{!}{=} \mathrm{R}_Z(\delta'_R)\mathrm{R}_X(\delta^*_X)\mathrm{R}_Y(\delta^*_Y)$$

This matrix equation can be solved by matching suitable sets of entries with only one of the adapted parameters appearing on the right-hand side:

$$\begin{bmatrix} c_{\delta_Y}c_{\delta_R} & -c_{\delta_Y}s_{\delta_R} & s_{\delta_Y} \\ s_{\delta_X}s_{\delta_Y}c_{\delta_R} + c_{\delta_X}s_{\delta_R} & -s_{\delta_X}s_{\delta_Y}s_{\delta_R} + c_{\delta_X}c_{\delta_R} & -s_{\delta_X}c_{\delta_Y} \\ -c_{\delta_X}s_{\delta_Y}c_{\delta_R} + s_{\delta_X}s_{\delta_R} & c_{\delta_X}s_{\delta_Y}s_{\delta_R} + s_{\delta_X}c_{\delta_R} & c_{\delta_X}c_{\delta_Y} \end{bmatrix}$$
$$=$$
$$\begin{bmatrix} c_{\delta'_R}c_{\delta^*_Y} - s_{\delta'_R}s_{\delta^*_X}s_{\delta^*_Y} & -s_{\delta'_R}c_{\delta^*_X} & c_{\delta'_R}s_{\delta^*_Y} + s_{\delta'_R}s_{\delta^*_X}c_{\delta^*_Y} \\ s_{\delta'_R}c_{\delta^*_Y} + c_{\delta'_R}s_{\delta^*_X}s_{\delta^*_Y} & c_{\delta'_R}c_{\delta^*_X} & s_{\delta'_R}s_{\delta^*_Y} - c_{\delta'_R}s_{\delta^*_X}c_{\delta^*_Y} \\ -c_{\delta^*_X}s_{\delta^*_Y} & s_{\delta^*_X} & c_{\delta^*_X}c_{\delta^*_Y} \end{bmatrix}$$

Assuming the magnitude of all tilt angles to be smaller than $\pi/2$ (to allow the elimination of the corresponding cosine within the atan2), the solutions for the tilt parameters δ^*_X, δ^*_Y, and the rotation δ'_R can be obtained from the entries at $[3,2]/[3,3]$, $[3,1]/[3,3]$, and $[1,2]/[2,2]$:

$$\delta^*_X = \text{atan2}(\cos(\delta_X)\sin(\delta_Y)\sin(\delta_R) + \sin(\delta_X)\cos(\delta_R), \cos(\delta_X)\cos(\delta_Y)/\cos(\delta^*_Y)) \tag{A9}$$

$$\delta^*_Y = \text{atan2}(\cos(\delta_X)\sin(\delta_Y)\cos(\delta_R) - \sin(\delta_X)\sin(\delta_R), \cos(\delta_X)\cos(\delta_Y)) \tag{A10}$$

$$\delta'_R = \text{atan2}(\cos(\delta_Y)\sin(\delta_R), -\sin(\delta_X)\sin(\delta_Y)\sin(\delta_R) + \cos(\delta_X)\cos(\delta_R)) \tag{A11}$$

By assuming only small tilt angles δ_X and δ_Y, Equations (A9)–(A11) can be simplified, and it is possible to express δ'_R in terms of δ_R by getting rid of the additional parameter:

$$\delta^*_X \approx \delta_X \cos(\delta_R) + \delta_Y \sin(\delta_R) \tag{A12}$$

$$\delta^*_Y \approx -\delta_X \sin(\delta_R) + \delta_Y \cos(\delta_R) \tag{A13}$$

$$\delta'_R \approx -\delta_R \tag{A14}$$

These two approximations for the adapted tilt parameters correspond to a two-dimensional rotation of $-\delta_R$ within the X-Y plane, and the rotation approximately acts as a similarity transformation on the original tilt matrix in the roll-pitch representation:

$$\mathrm{R}_X(\delta^*_X)\mathrm{R}_Y(\delta^*_Y) \approx \mathrm{R}_Z(-\delta_R)\mathrm{R}_X(\delta_X)\mathrm{R}_Y(\delta_Y)\mathrm{R}_Z(\delta_R) \tag{A15}$$

$$\begin{pmatrix} \delta^*_X \\ \delta^*_Y \end{pmatrix} \approx \begin{bmatrix} \cos\delta_R & \sin\delta_R \\ -\sin\delta_R & \cos\delta_R \end{bmatrix} \begin{pmatrix} \delta_X \\ \delta_Y \end{pmatrix} \tag{A16}$$

References

1. Stewenius, H.; Engels, C.; Nistér, D. Recent developments on direct relative orientation. *ISPRS J. Photogramm. Remote Sens.* **2006**, *60*, 284–294. [CrossRef]
2. Booij, O.; Zivkovic, Z. *The Planar Two Point Algorithm*; IAS Technical Report IAS-UVA-09-05; Informatics Institute, Faculty of Science, University of Amsterdam: Amsterdam, The Netherlands, 2009.
3. Möller, R.; Krzykawski, M.; Gerstmayr, L. Three 2D–warping schemes for visual robot navigation. *Auton. Robot.* **2010**, *29*, 253–291. [CrossRef]
4. Vardy, A.; Möller, R. Biologically plausible visual homing methods based on optical flow techniques. *Connect. Sci.* **2005**, *17*, 47–89. [CrossRef]
5. Möller, R. Local visual homing by warping of two–dimensional images. *Robot. Auton. Syst.* **2009**, *57*, 87–101. [CrossRef]
6. Franz, M.O.; Schölkopf, B.; Mallot, H.A.; Bülthoff, H.H. Where did I take that snapshot? Scene-based homing by image matching. *Biol. Cybern.* **1998**, *79*, 191–202. [CrossRef]
7. Fleer, D.; Möller, R. Comparing holistic and feature-based visual methods for estimating the relative pose of mobile robots. *Robot. Auton. Syst.* **2017**, *89*, 51–74. [CrossRef]
8. Möller, R.; Horst, M.; Fleer, D. Illumination tolerance for visual navigation with the holistic Min-Warping method. *Robotics* **2014**, *3*, 22–67. [CrossRef]
9. Möller, R. *Column Distance Measures and Their Effect on Illumination Tolerance in MinWarping*; Technical Report; Computer Engineering Group, Faculty of Technology, Bielefeld University: Bielefeld, Germany, 2016.
10. Hoffmann, A.; Möller, R. Cloud-edge suppression for visual outdoor navigation. *Robotics* **2017**, *6*, 38. [CrossRef]
11. Differt, D. *Holistic Methods for Visual Navigation of Mobile Robots in Outdoor Environments*; Bielefeld University: Bielefeld, Germany, 2017.
12. Fleer, D. Visual tilt estimation for planar-motion methods in indoor mobile robots. *Robotics* **2017**, *6*, 32. [CrossRef]
13. Differt, D. Real-time rotational image registration. In Proceedings of the International Conference on Advanced Robotics (ICAR), Hong Kong, China, 10–12 July 2017; pp. 1–6.
14. Horst, M.; Möller, R. Visual place recognition for autonomous mobile robots. *Robotics* **2017**, *6*, 9. [CrossRef]
15. Möller, R. *Tilt Correction in Panoramic Images*; Technical Report; Computer Engineering Group, Faculty of Technology, Bielefeld University: Bielefeld, Germany, 2009.
16. Euler, L. Formulae generales pro translatione quacunque corporum rigidorum. *Novi Comment. Acad. Sci. Petropolitanae* **1776**, *20*, 189–207.
17. Fermi, E.; Metropolis, N. *Numerical Solution of a Minimum Problem*; Technical Report; Los Alamos Scientific Lab.: Los Alamos, NM, USA, 1952.
18. Hooke, R.; Jeeves, T.A. "Direct Search" solution of numerical and statistical problems. *J. ACM* **1961**, *8*, 212–229. [CrossRef]
19. Nelder, J.A.; Mead, R. A simplex method for function minimization. *Comput. J.* **1965**, *7*, 308–313. [CrossRef]
20. Scaramuzza, D.; Martinelli, A.; Siegwart, R. A toolbox for easily calibrating omnidirectional cameras. In Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, 9–13 October 2006; IEEE: Piscataway, NJ, USA, 2006; pp. 5695–5701.

21. Rufli, M.; Scaramuzza, D.; Siegwart, R. Automatic detection of checkerboards on blurred and distorted images. In Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 22–26 September 2008; IEEE: Piscataway, NJ, USA, 2008; pp. 3121–3126.
22. Scaramuzza, D.; Martinelli, A.; Siegwart, R. A flexible technique for accurate omnidirectional camera calibration and structure from motion. In Proceedings of the Fourth IEEE International Conference on Computer Vision Systems (ICVS'06), New York, NY, USA, 5–7 January 2006; IEEE: Piscataway, NJ, USA, 2006; p. 45.
23. Möller, R. *A SIMD Implementation of the MinWarping Method for Local Visual Homing*; Technical Report; Computer Engineering Group, Faculty of Technology, Bielefeld University: Bielefeld, Germany, 2016.
24. Möller, R. *Design of a Low-Level C++ Template SIMD Library*; Technical Report; Computer Engineering Group, Faculty of Technology, Bielefeld University: Bielefeld, Germany, 2016.
25. Luus, R.; Jaakola, T.H.I. Optimization by direct search and systematic reduction of the size of search region. *AIChE J.* **1973**, *19*, 760–766. [CrossRef]