Machine Vision and Applications (2024) 35:58
https://doi.org/10.1007/s00138-024-01534-2

ORIGINAL PAPER

OmniGlasses: an optical aid for stereo vision CNNs to enable omnidirectional image processing

Julian B. Seuffert[1] · Ana C. Perez Grassi[2] · Hamza Ahmed[1] · Roman Seidel[1] · Gangolf Hirtz[1]

Received: 4 April 2023 / Revised: 26 January 2024 / Accepted: 6 March 2024 / Published online: 23 April 2024

Abstract
Stereo vision is a key technology for 3D scene reconstruction from image pairs. Most approaches process perspective images from commodity cameras. These images, however, have a very limited field of view and only picture a small portion of the scene. In contrast, omnidirectional images, also known as fisheye images, exhibit a much larger field of view and allow a full 3D scene reconstruction with a small amount of cameras if placed carefully. However, omnidirectional images are strongly distorted which make the 3D reconstruction much more sophisticated. Nowadays, a lot of research is conducted on CNNs for omnidirectional stereo vision. Nevertheless, a significant gap between estimation accuracy and throughput can be observed in the literature. This work aims to bridge this gap by introducing a novel set of transformations, namely *OmniGlasses*. These are incorporated into the architecture of a fast network, i.e., *AnyNet*, originally designed for scene reconstruction on perspective images. Our network, *Omni-AnyNet*, produces accurate omnidirectional distance maps with a mean absolute error of around 13 cm at 48.4 fps and is therefore real-time capable.

Keywords Epipolar geometry · Fisheye · Omnidirectional · Look up table · Stereo vision · View synthesis

1 Introduction

3D image processing is an important research area, that has gained a lot of attention in the past decades. It is essential for spatial reconstruction of a scene and therefore for many applications in the fields of robotics, autonomous driving and scene understanding. Such 3D reconstruction techniques even reached medicine, human pose estimation and human action recognition.

One prominent 3D reconstruction technique is stereo vision. Inspired by human 3D perception, stereo vision aims to recover the depth of a scene by aggregating the information of multiple cameras. Matching algorithms constitute the core of stereo vision. These algorithms look for corresponding points in the different images, that is, those points in the images that result from the projection of the same point in the real world. Finally, the difference in the position of these points allows the stereo methods to retrieve their distance.

Most of the research in this area is based on images that underlie the perspective camera model and are mainly free of distortion artifacts. For performance reasons, these approaches rectified the input images. Rectified images correspond to multi-camera setups with aligned viewing directions and collinear x-axes. As result, in all rectified images, the corresponding points are found on horizontal and collinear lines called epipolar lines. This represents an advantage for the matching algorithms whose search space is reduced to a one dimension given by this epipolar line.

Perspective images have a field of view (FOV) of usually less than 65°. In many applications, this limited FOV represents a big drawback by requiring the use of multiple

✉ Julian B. Seuffert
julian.seuffert@etit.tu-chemnitz.de

Ana C. Perez Grassi
ana-cecilia.perez-grassi@informatik.tu-chemnitz.de

Hamza Ahmed
hamzaahmed9305@gmail.com

Roman Seidel
roman.seidel@etit.tu-chemnitz.de

Gangolf Hirtz
g.hirtz@etit.tu-chemnitz.de

[1] Faculty of Electrical Engineering and Information Technology, Chemnitz University of Technology, Reichenhainer Str. 70, Chemnitz 09126, Saxony, Germany

[2] Faculty of Computer Science, Chemnitz University of Technology, Reichenhainer Str. 70, Chemnitz 09126, Saxony, Germany

calibrated cameras to cover the region of interest. For this reason, omnidirectional images from so-called fisheye cameras have gained a lot of attention in the last years. These cameras exhibit a much higher FOV of around 180° and therefore show much more content of the scene with only one sensor. However, the higher FOV of these cameras is associated with a high distortion in the omnidirectional images. For the stereo vision, this distortion results in a new and more complex search space for corresponding points. The same holds true for the so-called normalized omnidirectional images, which are, broadly speaking, a scaled version of the non-normalized counterparts. As shown in Fig. 1, the search space for stereo correspondences on (normalized) omnidirectional images is not more a horizontal line, as in the case of perspective images, but a curve. This curve is known as epipolar curve. In order to be able to use perspective matching algorithms, many stereo methods for omnidirectional images unwarp the images to cylindrical images [1–3] to recover the epipolar lines. However, these transformations reduce the FOV as well as the efficiency of the methods.

Neural networks have also reached stereo vision. Networks for perspective images, like *AnyNet* [4], reach excellent results and allow more than 30 fps on high resolution images. Also for omnidirectional stereo vision different networks has been proposed by Won et al. [5–7]. These networks use spherical sweeping in which the omnidirectional images are first projected on spherical images with high FOV, also known as equirectangular projection (ERP) images, and then mapped on concentric global spheres surrounding the cameras' rig center. Although this strategy gives high precision results, its structure and transformations make it computationally intensive and prevent it from achieving real-time performance.

In this work, we aim to bridge the gap between accuracy and fast processing time in omnidirectional stereo vision. This is accomplished with the following contributions:

- We propose *OmniGlasses*, a set of look up tables (LUTs) carefully designed for fast and incremental stereo correspondence search on omnidirectional images.
- We integrate *OmniGlasses* into *AnyNet* as part of our new network *Omni-AnyNet*. This demonstrates how fast networks can be modified to process omnidirectional images.
- We proof the efficiency of *Omni-AnyNet* and therefore *OmniGlasses* experimentally. We show that the integration of *OmniGlasses* comes with only low cost in terms of throughput while producing accurate scene reconstructions.
- All results are compared to the state-of-the-art network *OmniMVS*$^{+}$.

2 Related work

Depth estimation from omnidirectional images using neural networks was pioneered by Won et al. [5–7]. The input images of these networks come from a wide-baseline multi-view (four cameras) omnidirectional setup. The first of these works introduces SweepNet [5], a CNN that computes the matching costs of ERP image pairs warped from the omnidirectional images. The resulting cost volume is refined by applying a semi global matching (SGM) algorithm [9] and

(a) left normalized omnidirectional image

(b) right normalized omnidirectional image

Fig. 1 Search space and parametrization of images points. A point of interest is annotated with the triangle in (**a**) and with a filled circle in (**b**). Furthermore the triangle can also be found in (**b**) as a yellow triangle to see the different locations more clearly. Given an image point on the left omnidirectional image, the corresponding right point has to be searched along a so-called epipolar curve for 3D reconstruction purposes. These image points can be parameterized by the azimuth $\phi \in \{\phi_\mathrm{l}, \phi_\mathrm{r}\}$ and the elevation $\theta \in \{\theta_\mathrm{l}, \theta_\mathrm{r}\}$ (See Sect. 3.1.2). The images in this figure rely on a sample of the *THEOStereo* dataset [8]

finally the depth map is estimated. SweepNet presents a problem to manage occlusions, which are typical for the proposed wide-baseline omnidirectional setup. To overcome this problem, Won et al. propose *OmniMVS* [6], an end-to-end deep neural network consisting of three blocks: feature extractor, spherical sweeping and cost volume computation. *OmniMVS* was then extended to *OmniMVS*$^+$ [7] by incorporating an entropy boundary loss for a better regularization of the cost volume computation. Furthermore, *OmniMVS*$^+$ improves the efficiency of its predecessor in terms of memory consumption and run time. This is achieved by merging opposite camera views into the same ERP image.

Almost parallel to *OmniMVS*, Wang et al. [10] developed 360SD-Net. This network takes as input a pair of ERP images from two cameras aligned in a top-bottom manner. The features extracted from the images are concatenated with those extracted from a polar angle map, in order to introduce the geometry information in the model. Atrous-Spatial Pyramid Pooling (ASPP) modules are proposed to enlarge the receptive field followed by a learnable cost volume (LCV). The final disparity is regressed by using a Stacked-Hourglass module.

Komatsu et al. [11] present IcoSweepNet for depth estimation from four omnidirectional images. IcoSweepNet based on an icosahedron representation, spherical sweeping and a 2D/3D CNN architecture called CrownConv, which is specially designed for extracting features of icosahedrons. By considering the extrinsic camera parameters, this network achieves more robust results against camera misalignments than *OmniMVS*.

Córdova-Espaza et al. [12] combine a deep learning matching algorithm and stereo epipolar constraints to reconstruct 3D scenes from a stereo catadioptric system. After converting the images to panoramic ones, the matching points pairs proposed by a DeepMatching algorithm [13] are filtered according to the defined epipolar constraints. This method requires a tradeoff between 3D-point sparsity and reconstruction error, which is achieved by adjusting a threshold on the distance between the proposed points and their corresponding epipolar curves.

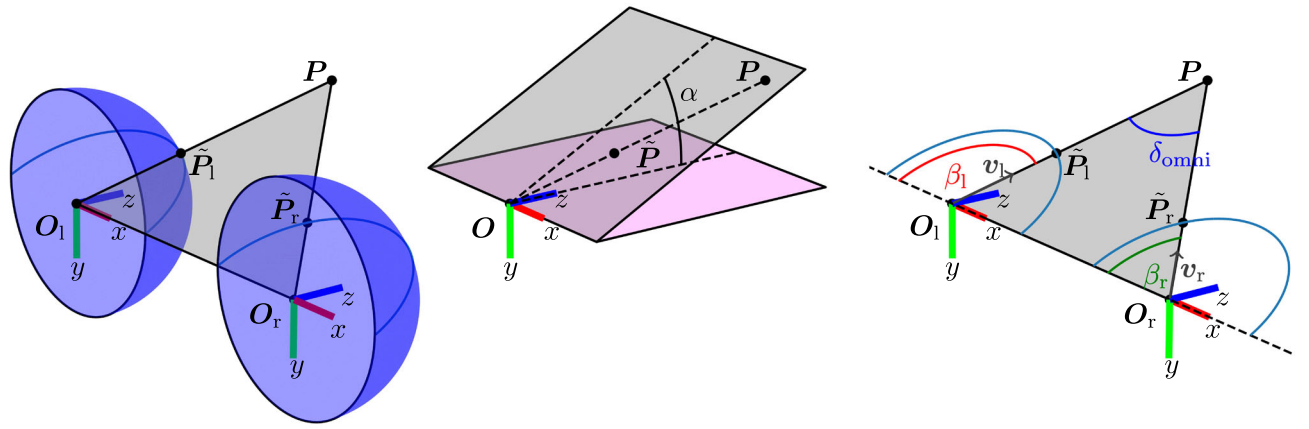
In [14], Lee et al. propose a semi-supervised learning method by expanding *OmniMVS*$^+$ with a second loss function. The pixel-level loss selects a supervised loss or a unsupervised re-projection loss according to the availability or not of ground truth information. This combination of loss functions allows to consider the common sparsity of real depth ground truth generated by LIDAR. This work achieves better results than *OmniMVS*$^+$ in presence of sparsity and calibration errors, making the network more robust to work with real data.

Li et al. [15] introduce the Spherical Convolution Residual Network (SCRN) for omnidirectional depth estimation. This network processes ERP images as inputs, which are sampled in spherical meshes. In this way, the non-linear epipolar constrains in the plain are converted to linear constraints in the sphere. The SCRN is then followed by a planar refinement network (PRN) to go back to a 2D representation. The full architecture is called Cascade Spherical Depth Network (CSDNet).

While the architectures intended to estimate depth maps from perspective images have reached real-time conditions [4], the analogous architectures developed for omnidirectional images are still far from these levels of efficiency. Outside the field of neural networks and machine learning, Meuleman et al. [16] present a deterministic real-time sphere-sweeping stereo method. This work was developed for a 360° field of view setup consisting of 4 omnidirectional cameras. The proposed adaptive spherical matching runs directly on the input images but it considers only the best cameras pairs for each correspondence, which allows to reduce the computation time. A fast inter-scale bilateral cost volume filtering allows the method to reach 29 fps. This method performs better and faster than *OmniMVS* and CrownConv, however, it lacks the generalization power of the learning methods that makes the results robust to changes in the input data. Our work adapts the sweeping method from [16] making it part of the learning process of a neural network. As part of this integration, we also present an optimization process in order to save computational time without loosing depth resolution. Moreover, as explain in Sect. 3, the matching process in our approach is performed on features instead on intensity values and the considered setup is a stereo one.

3 Omnidirectional stereo vision

In perspective stereo vision, one common way to retrieve the depth (z-distance) of a scene is to determine the so-called disparity map between two images $\boldsymbol{I}_\mathrm{l}$ and $\boldsymbol{I}_\mathrm{r}$ captured by two cameras at different positions. In the case of horizontally aligned cameras, the subindexes l and r denote left and right, respectively. For performance reasons, the images $\boldsymbol{I}_\mathrm{l}$ and $\boldsymbol{I}_\mathrm{r}$ are usually rectified. This means they do not present any distortion and their corresponding x- and y-axes are respectively parallel. Moreover, the x-axes are collinear. Under these conditions, a real world point $\boldsymbol{P}$ captured by both cameras is projected on the rectified images at pixels with the same y-coordinate on collinear horizontal lines, called epipolar lines. The difference between the x-coordinates x_l and x_r of this projected point on the left and the right image gives the disparity value $d_\mathrm{persp}(x_\mathrm{l}, y) = x_\mathrm{l} - x_\mathrm{r}$. The x_l- and x_r-coordinates of corresponding points are extracted along the epipolar lines using stereo matching techniques, e.g., Block Matching [17]. Finally, a disparity map $D_\mathrm{persp}(x_\mathrm{l}, y) = x_\mathrm{l} - x_\mathrm{r}$ is generated, where each disparity value is inverse proportional to the searched depth value $z(x_\mathrm{l}, y)$.



(a) projection on image hemispheres (b) pitch angle α (c) relationship between β and δ_{omni}

Fig. 2 Epipolar Geometry. In (**a**), the projection points $\tilde{\boldsymbol{P}}_\text{l}$ and $\tilde{\boldsymbol{P}}_\text{r}$ of $\boldsymbol{P}$ on the image hemispheres are located in the so-called epipolar plane spanned by the camera centers $\boldsymbol{O}_\text{l}$ and $\boldsymbol{O}_\text{r}$ and the world point $\boldsymbol{P}$. (**b**) shows the parametrization of the epipolar plane by the angle α. (**c**) depicts δ_{omni} and the yaw angles β_l and β_r of $\boldsymbol{v}_\text{l}$ and $\boldsymbol{v}_\text{r}$. These are used together with α to triangulate $\boldsymbol{P}$. Unlike (**a**) and (**c**), the system in (**b**) is not camera-specific and applies to any camera coordinate system at $\boldsymbol{O} \in \{\boldsymbol{O}_\text{l}, \boldsymbol{O}_\text{r}\}$

Nowadays stereo vision networks for perspective images [4, 18–20] perform stereo matching on feature maps rather than on the original input images. These maps are computed from both rectified input images by using a feature extractor, e.g., U-Net [21]. Then one of the feature maps is horizontally shifted and compared with the other camera's feature map for each shift. This shift in x-direction (along the epipolar line) for a value $d_\text{persp} \in [0, d_\text{max}]$ generates a pixel-wise cost volume of size $H \times W \times D$, where H and W describe the height and width of the feature map and D the number of considered disparity values between 0 and d_max. This way, networks like *AnyNet* [4] summarize the costs for all possible disparity assumptions. Finally from this cost volume a regression module retrieves the optimal disparity value for each pixel.

However, this method of disparity estimation by horizontally shifting feature maps along the epipolar lines is not valid for omnidirectional images. In the case of using omnidirectional cameras, the world point $\boldsymbol{P}$ is projected onto an image hemisphere rather than an image plane [1–3] as shown in Fig. 2a. As a result, corresponding points on omnidirectional images are located along a so-called epipolar curve instead of along a line, as in the case of perspective images (See Fig. 1). Therefore, the disparity cannot be considered anymore as an offset in x-direction since two corresponding pixels in the left and right images may also differ in the y-coordinate. In this case, the correspondence search and the following disparity calculation should be based on an epipolar geometry for canonical stereo configurations, which follows an omnidirectional camera model. As described in the next subsection, our work describes omnidirectional cameras using the equiangular camera model.

3.1 Epipolar geometry for omnidirectional stereo vision

We first describe, in Sect. 3.1.1, the search space for stereo correspondences along the epipolar curves on the image hemispheres (See Fig. 2a). Then we link this search space to its corresponding search space on the omnidirectional images in Sect. 3.1.2. Finally, in Sect. 3.1.3, we propose *OmniGlasses*, a set of LUTs designed for searching stereo correspondences in omnidirectional image pairs.

3.1.1 Relationship of world points and their projection on the image hemisphere

Let $\tilde{\boldsymbol{P}}_\text{l}$ and $\tilde{\boldsymbol{P}}_\text{r}$ be the projection points of a world point $\boldsymbol{P}$ on the left and right image hemisphere of a canonical stereo setup, as shown in Fig. 2a. After bringing them into a joint coordinate system, both projection points, the camera centers $\boldsymbol{O}_\text{l}$ and $\boldsymbol{O}_\text{r}$ and the world point $\boldsymbol{P}$ itself sit on a so-called epipolar plane. As a consequence, given a reference point $\tilde{\boldsymbol{P}}_\text{l}$ on the left image hemisphere, the orientation of the epipolar plane determines the valid search space for $\tilde{\boldsymbol{P}}_\text{r}$ on the right image hemisphere. The orientation of the epipolar plane can be described through the pitch angle α between the epipolar plane and the plane spanned by the camera's x- and z-axis (See Fig. 2b):

$$\alpha = \arctan 2(\tilde{p}_y, \tilde{p}_z), \tag{1}$$

where $\tilde{p}_y$ and $\tilde{p}_z$ are the components in y- and z-direction of a point $\tilde{\boldsymbol{P}} \in \{\tilde{\boldsymbol{P}}_\text{l}, \tilde{\boldsymbol{P}}_\text{r}\}$. Furthermore, the position of $\tilde{\boldsymbol{P}}_\text{l}$ and $\tilde{\boldsymbol{P}}_\text{r}$ is determined by their corresponding light rays. Each

light ray can be described with the help of the unit vector $\boldsymbol{v} = \tilde{\boldsymbol{P}}/\|\tilde{\boldsymbol{P}}\|$, where $\boldsymbol{v} \in \{\boldsymbol{v}_\mathrm{l}, \boldsymbol{v}_\mathrm{r}\}$ (See Fig. 2c) points in the opposite direction to the given light ray. Finally, scaling $\boldsymbol{v}$ by the value of the focal length f results in the projection point $\tilde{\boldsymbol{P}} = f \cdot \boldsymbol{v}$.

The angle δ_omni between the light rays described by $\boldsymbol{v}_\mathrm{l}$ and $\boldsymbol{v}_\mathrm{r}$ is denoted as normalized disparity by Li et al. [22, 23]. This angle can also be defined in relation to the angles β_l and β_r between the vectors $\boldsymbol{v}_\mathrm{l}$ and $\boldsymbol{v}_\mathrm{r}$ and the unit vector $-\boldsymbol{e}_x$ pointing in the negative direction of the x-axis (See Fig. 2c):

$$\delta_\mathrm{omni} = \beta_\mathrm{l} - \beta_\mathrm{r} \tag{2}$$

Finally, by expressing the vector $\boldsymbol{v}$ in terms of the angles $\beta \in \{\beta_\mathrm{l}, \beta_\mathrm{r}\}$ and α, a projection point $\tilde{\boldsymbol{P}}$ can be defined as follows:

$$\begin{aligned}\tilde{\boldsymbol{P}}(\alpha, \beta) &= f \cdot \boldsymbol{R}_x(-\alpha) \cdot \boldsymbol{R}_y(\beta) \cdot (-\boldsymbol{e}_x) \\ &= f \cdot \begin{pmatrix} -\cos\beta \\ \sin\alpha \sin\beta \\ \cos\alpha \sin\beta \end{pmatrix} = f \cdot \boldsymbol{v},\end{aligned} \tag{3}$$

where $\boldsymbol{R}_x$ and $\boldsymbol{R}_y$ denote rotation matrices around x- and y-axis, respectively. Equation 3 describes the rotation of $-\boldsymbol{e}_x$ around the camera's y-axis (on the xz-plane) to account for the yaw angle β. Then, the introduction of the pitch α by rotating around the x-axis by $-\alpha$. The result is the unit vector $\boldsymbol{v}$, which is finally scaled by f to obtain the projection point $\tilde{\boldsymbol{P}}$.

The goal of stereo matching is to find the projection point $\tilde{\boldsymbol{P}}_\mathrm{r}$ in the right image hemisphere, that corresponds to a given projection point $\tilde{\boldsymbol{P}}_\mathrm{l}$ in the left one. As both points belong to the same epipolar plane, the value of α can be calculated from $\tilde{\boldsymbol{P}}_\mathrm{l}$ (Conf. Equation 1). According to Eq. 3, given α, the search space for $\tilde{\boldsymbol{P}}_\mathrm{r}$ is defined by all possible values of β_r or, what is the same, by all possible values of δ_omni (Conf. Equation 2). Iterating over all possible disparity values $\hat{\delta}_\mathrm{omni} \in \left[0, \hat{\delta}_\mathrm{omni,max}\right]$ results in a set of angles $\hat{\beta}_\mathrm{r}(\hat{\delta}_\mathrm{omni}) = \beta_\mathrm{l} - \hat{\delta}_\mathrm{omni}$ (Eq. 2), which together with α define all possible correspondence points $\hat{\boldsymbol{P}}_\mathrm{r}(\hat{\delta}_\mathrm{omni})$ for $\tilde{\boldsymbol{P}}_\mathrm{l}$ (Eq. 3).

The purpose of stereo matching is then to find from all candidates $\hat{\boldsymbol{P}}_\mathrm{r}(\hat{\delta}_\mathrm{omni})$, which one best represents the observed projection $\tilde{\boldsymbol{P}}_\mathrm{r}$ and with it the best value of $\hat{\delta}_\mathrm{omni}$. This disparity value is then the last key to spatially reconstruct the scene as described in the next section.

3.1.2 Relationship of world points and their projection on the omnidirectional image

A projection point $\tilde{\boldsymbol{P}} \in \{\tilde{\boldsymbol{P}}_\mathrm{l}, \tilde{\boldsymbol{P}}_\mathrm{r}\}$ on an image hemisphere corresponds to a point on the resulting omnidirectional image. In order to avoid the conversion between omnidirectional image and image hemisphere during runtime, the search of stereo correspondences is performed directly on the omnidirectional images according to the restrictions derived in Sect. 3.1.1. Cameras following the equiangular projection model project an incoming light ray onto the omnidirectional image depending on the elevation θ and azimuth ϕ angles of the corresponding vector $\boldsymbol{v}$ [24]. The elevation angle θ describes the angle between $\boldsymbol{v}$ and the optical axis $\boldsymbol{e}_z$:

$$\theta = \arccos(v_\mathrm{z}) = \arccos(\cos\alpha \sin\beta) \tag{4}$$

The azimuth angle ϕ stands for the angle between the x-axis and the projection of $\boldsymbol{v}$ onto the xy-plane:

$$\begin{aligned}\phi &= \arctan 2(v_\mathrm{y}, v_\mathrm{x}) \bmod 2\pi \\ &= \arctan 2(\sin\alpha \sin\beta, -\cos\beta) \bmod 2\pi\end{aligned} \tag{5}$$

The modulo operator ensures that $\phi \in [0, 2\pi[$ and avoids negative values. By equiangular projection, $r = \theta$, where r is the distance between the projected point and the image distortion center as show in Fig. 1. With the help of these polar coordinates, the light ray can be projected onto the normalized omnidirectional image at

$$\begin{pmatrix} x_\mathrm{norm} \\ y_\mathrm{norm} \end{pmatrix} = \theta \cdot \begin{pmatrix} \cos\phi \\ \sin\phi \end{pmatrix} = r \cdot \begin{pmatrix} \cos\phi \\ \sin\phi \end{pmatrix} \tag{6}$$

and on the omnidirectional image itself at:

$$\begin{pmatrix} x \\ y \end{pmatrix} = f \cdot \begin{pmatrix} x_\mathrm{norm} \\ y_\mathrm{norm} \end{pmatrix} + \begin{pmatrix} c_\mathrm{x} \\ c_\mathrm{y} \end{pmatrix} \tag{7}$$

Here we assume equal focal length f for x- and y-direction. The vector $(c_\mathrm{x}, c_\mathrm{y})^T$ describes the coordinates of the image distortion center.

Now θ, ϕ and finally $\boldsymbol{v}$ can be restored from the pixel locations in the normalized image itself. The elevation θ is calculated as:

$$\theta = r = \left\| \begin{pmatrix} x_\mathrm{norm} \\ y_\mathrm{norm} \end{pmatrix} \right\| \tag{8}$$

Note that an explicit conversion of radians to pixels an vice versa is not necessary in Eqs. 6 and 8 as both pixels and radians are dimensionless. The azimuth ϕ can be retrieved from the normalized image as:

$$\phi = \arctan 2(y_\mathrm{norm}, x_\mathrm{norm}) \bmod 2\pi \tag{9}$$

The relationship between image points on normalized image pairs and their parameters $\phi \in \{\phi_\mathrm{l}, \phi_\mathrm{r}\}$ and $r \in \{r_\mathrm{l}, r_\mathrm{r}\}$ are visualized in Fig. 1 for both left and right image.

Finally, the vector $\boldsymbol{v}$ that links the omnidirectional images with the search space on the image hemispheres can be restored:

$$\boldsymbol{v} = \boldsymbol{R}_z(\phi)\boldsymbol{R}_y(\theta)\boldsymbol{e}_z = \begin{pmatrix} \cos\phi \sin\theta \\ \sin\phi \sin\theta \\ \cos\theta \end{pmatrix} \tag{10}$$

Note that the equations in this section are based on the equidistant camera model and may differ for real world omnidirectional dioptric lenses. However, adapting these equations to other similar camera models is straightforward.

3.1.3 Searching strategy on omnidirectional images

By applying Eqs. 8 and 9, the polar coordinates $(r_\mathrm{l}, \phi_\mathrm{l})^T$ for each pixel in the left omnidirectional image $\boldsymbol{I}_\mathrm{l}$ can be calculated. Then the vector $\boldsymbol{v}_\mathrm{l}$ and the angles α and β_l can be derived for each pixel in $\boldsymbol{I}_\mathrm{l}$ by using Eqs. 10, 1 and 3.

As mentioned before, during the correspondence search, a set of plausible disparity values $\hat{\delta}_\mathrm{omni} \in \left[0, \hat{\delta}_\mathrm{omni,max}\right]$ and their corresponding β_l are assumed for each projection point $\tilde{\boldsymbol{P}}_\mathrm{l}$. The omnidirectional disparity $\hat{\delta}_\mathrm{omni}$ of Li et al. [22, 23] describes an angle represented by a floating-point number. For sake of implementation, $\hat{\delta}_\mathrm{omni}$ needs to be redefined as a discrete variable by sampling its valid range with step size $S = \hat{\delta}_\mathrm{omni,max}/D$. This results in $\hat{\delta}_\mathrm{omni} \in \{\hat{\delta}_0, \ldots, \hat{\delta}_s, \ldots, \hat{\delta}_{D-1}\}$, with $0 \leq s \leq D-1$, where D is the number of considered disparities values.

Each value of $\hat{\delta}_s$ results in an angle $\hat{\beta}_\mathrm{r}(\hat{\delta}_s)$ (Conf. Equation 2). Given a pixel $(\theta_\mathrm{l}, \phi_\mathrm{l})^T$ in the left image, its correspondence candidates $(\hat{\theta}_\mathrm{r}, \hat{\phi}_\mathrm{r})^T$ in the right image can then be found by substituting β by $\hat{\beta}_\mathrm{r}(\hat{\delta}_s)$ in Eqs. 4 and 5. Finally, the coordinates of the correspondence candidates $\left(\hat{x}_\mathrm{r}, \hat{y}_\mathrm{r}\right)^T$ on the right image can be obtained by using Eqs. 6 and 7 for each disparity hypothesis $\hat{\delta}_s$.

In order to implement a stereo matching process, the coordinates $\left(\hat{x}_\mathrm{r}, \hat{y}_\mathrm{r}\right)^T$ derived from each hypothetical disparity value $\hat{\delta}_s$ are used to project the right image on the left one. The resulting transformed image $\hat{I}_\mathrm{r}^{\hat{\delta}_s}(x_\mathrm{l}, y_\mathrm{l})$ can be defined as follows:

$$\hat{I}_\mathrm{r}^{\hat{\delta}_s}(x_\mathrm{l}, y_\mathrm{l}) = I_\mathrm{r}\left(\hat{x}_\mathrm{r}, \hat{y}_\mathrm{r}\right) \tag{11}$$

This view transformation can be easily implemented as a backward-projection with look up tables (LUTs). These LUTs store for each pixel location $(x_\mathrm{l}, y_\mathrm{l})^T$ in the left image and each value of $\hat{\delta}_s$ the resulting location of the correspondence point $\left(\hat{x}_\mathrm{r}, \hat{y}_\mathrm{r}\right)^T$ in the right image. We name the volume comprising all LUTs for all disparity hypothesis *Full OmniGlasses*. These LUTs have the size $D \times H \times W \times 2$, where D is the number of hypothetical disparity values, $H \times W$ is given by the size of the picture and the 2 refers to the two coordinates $\hat{x}_\mathrm{r}$ and $\hat{y}_\mathrm{r}$. A sparse version of *OmniGlasses* will be introduced in Sect. 3.2.

By applying these LUTs to the right image, D transformed images $\hat{I}_\mathrm{r}^{\hat{\delta}_s}(x_\mathrm{l}, y_\mathrm{l})$ are obtained. The optimal value of $\hat{\delta}_s$ for each coordinate $(x_\mathrm{l}, y_\mathrm{l})$ in the left image is the one that maximizes the similarity between the intensity $\hat{I}_\mathrm{r}^{\hat{\delta}_s}(x_\mathrm{l}, y_\mathrm{l})$ and $I_\mathrm{l}(x_\mathrm{l}, y_\mathrm{l})$. In order to determine this value, a measurement of the similarity between both images, the left one and the transformed right one, is performed. In our work, we used the L_1 norm, for similarity measurements, which is the cost metric used by *AnyNet*, as explain in the next section. A cost volume $\boldsymbol{C}$, of size $D \times H \times W$ stores all resulting $C^s(x_\mathrm{l}, y_\mathrm{l})$ with:

$$C^s(x_\mathrm{l}, y_\mathrm{l}) = \left| I_\mathrm{l}(x_\mathrm{l}, y_\mathrm{l}) - \hat{I}_\mathrm{r}^{\hat{\delta}_s}(x_\mathrm{l}, y_\mathrm{l}) \right| \tag{12}$$

The final disparity value can be determined with the help of the softargmin on the cost values for each pixel separately [19] and refined by a disparity refinement module [4]. The softargmin function gives the index of the optimal disparity value. Moreover, this function allows to obtain a subindex precision by weighting and integrating the cost volume results. This local oversampling results in a subindex s' between two given indexes $s \leq s' \leq s+1$ and a final estimated disparity $\hat{\delta}' = s' \cdot S$, with $\hat{\delta}_{\lfloor s' \rfloor} \leq \hat{\delta}' \leq \hat{\delta}_{\lceil s' \rceil}$. Finally, following [22, 23], the Euclidean distance $\hat{\rho}_\mathrm{l}$ bewteen world point $\boldsymbol{P}$ and left camera $\boldsymbol{O}_\mathrm{l}$ is given by

$$\hat{\rho}_\mathrm{l} = b \cdot \frac{\sin\hat{\beta}_\mathrm{r}}{\sin\hat{\delta}'} = b \cdot \frac{\sin(\beta_\mathrm{l} - \hat{\delta}')}{\sin\hat{\delta}'}, \tag{13}$$

with b being the baseline of the stereo camera, i.e., the distance between $\boldsymbol{O}_\mathrm{l}$ and $\boldsymbol{O}_\mathrm{r}$.

3.2 Integration of OmniGlasses into AnyNet

AnyNet [4] is a network for disparity estimation with state-of-the-art results on perspective images. Unlike what is described in the previous sections, *AnyNet* does not perform stereo matching on the input image but rather on feature maps extracted from them. Designed to achieve a good computing time, *AnyNet* estimates the disparity in a hierarchical way. The network is organized into four stages, where each stage increases the resolution of the disparity map generated in the previous one. Stage 1, takes feature maps of $1/16$ of the full image resolution. Stages 2 and 3 increase this resolution to $1/8$ and $1/4$ of the original resolution respectively. Finally, the last stage estimates the full resolution disparity map.

In the first stage, $D = 12$ values are considered for the disparity estimation. In stages 2 and 3, *AnyNet* takes the disparity estimation of the preceding stage (rounded to integer)

as initial value and predicts a residual disparity instead of undertaking a full estimation. Here it is assumed that, the disparity $\hat{d}^i_{\text{persp}}$ estimated for a pixel (x_1, y) in stage $i \in \{2, 3\}$ does not differ by more than two pixels from the previous prediction, i.e., $\hat{d}^i_{\text{persp}} \in [2\hat{d}^{i-1}_{\text{persp}} - 2, 2\hat{d}^{i-1}_{\text{persp}} + 2]$. This means that the new stage needs to consider only 5 values of disparity at maximum: The one received by the previous stage, two values higher and two lower. This incremental improvement per stage saves much computational time and enables the real-time capability of *AnyNet*. In this work, i refers to the stage of *AnyNet* or its adaptions always. However, its range can be refined individually to make more specific statements.

AnyNet is designed for perspective images captured from a canonical camera setup. Therefore, it takes advantage of the parallelism and collinearity of the epipolar lines on the left and right rectified images. Hence, the disparity $d_{\text{persp}}(x_1, y) = x_1 - x_r$ results from an offset only in x-direction. Therefore, the cost volume $C^{\hat{d}_{\text{persp}}}(x_1, y_1)$ can be generated by overlapping the right feature map on the left one and horizontally shifting it for each considered value $\hat{d}^i_{\text{persp}}$. For this purpose, the L_1 norm serves as cost measure to evaluate each assumed disparities value. The final disparities for stages 1–3 are then estimated as a weighted softargmin on the cost volume and refined through a network. Stage 4 has the task of refining the disparity maps of stage 3 and upsampling it to the original resolution. For this, it uses an SPNet module [25] together with the left image as a guide.

As shown in [8], the displacement in x-direction cannot adequately model perspective disparity arranged in an omnidirectional manner. In this case, the subsequent disparity refinement module can only partially correct the disparity estimates. In this work, the original *AnyNet* process for generating the cost volume is replaced by the proposed *OmniGlasses*.

By considering the 2D displacement in the x- and y-directions, *OmniGlasses* can follow epipolar curves (as shown in Fig. 1) during the stereo matching process. Modelling this 2D displacement, which describes the distortion in omnidirectional images, is the main challenge for depth estimation. We will show in Sect. 5 that *OmniGlasses* successfully overcome this challenge and significantly reduces the error compared to applying stereo matching along epipolar lines.

In this way, an appropriate omnidirectional view synthesis, as described in Sects. 3.1.1, 3.1.2, 3.1.3, is incorporated into *AnyNet*. Each stage from 1 to 3 has its own parameters D_i, S_i and therefore its own valid set of assumed disparities $\hat{\delta}_{\text{omni}} \in \{\hat{\delta}_0, \ldots, \hat{\delta}_{s_i}, \ldots, \hat{\delta}_{D_i-1}\}$, with $0 \leq s_i \leq D_i - 1$ and $i \in \{1, 2, 3\}$.

Analogous to the original *AnyNet*, $D_1 = 12$ is selected for the first stage. The resulting disparity map is then refined in each successive stage by estimating a residual value for the upsampled version. In the same way as with perspective images, the residual calculation reduces the number of view synthesis transformations to the predefined number of residual values. We conserve the original number of residual values by considering a disparity range between two disparity indexes lower and two higher than the one received from previous stage. This results in five transformations for each feature vector given by $\hat{\delta}^i_{\text{omni}} \in \{\hat{\delta}^i_0, \cdots, \hat{\delta}^i_2, \cdots, \hat{\delta}^i_4\}$, with $i \in \{2, 3\}$, $\hat{\delta}^i_2$ is the estimated disparity from the previous stage, $\hat{\delta}^i_{s_i} = \hat{\delta}^i_2 + S_i(s_i - 2)$.

As a consequence, we first generated three *OmniGlasses* of shape $D_i \times H_i \times W_i \times 2$, referred to as *Full OmniGlasses*, for the first three stages before runtime and further reduced the shape of the LUT of stage $i \in \{2, 3\}$ to $5 \times H_i \times W_i \times 2$ during runtime. The reduced versions of *OmniGlasses* are hereinafter denoted as *Sparse OmniGlasses*. The values inside the *Sparse OmniGlasses* depend on the predictions of the previous stage for each feature vector independently, as shown in Fig. 3.

Analogously to *AnyNet*, the resulting disparity values from one stage are rounded to integers and upscaled in order to incorporate them into the next stage. The shapes of all *OmniGlasses* are documented in Table 1. We hereinafter refer to this version of *AnyNet* leveraging *OmniGlasses* as *Omni-AnyNet*.

4 Experiments

We propose three groups of experiments to demonstrate the effectiveness of our approach. First, we show qualitative results of *OmniGlasses* as a standalone module. This experiment aims to prove the correctness of the proposed LUTs in the generation of transformed omnidirectional images for different disparity values. Furthermore, we determine a set of measurements of different error metrics to show the accuracy of *OmniGlasses* as part of *Omni-AnyNet*. The second group of experiments presents an ablation study to compare the performance of *AnyNet* with and without the proposed adaptation. Moreover, this study shows the importance of choosing the right disparity metric. Finally, we compare *Omni-AnyNet* with the state-of-the-art network *OmniMVS*$^+$ [7].

There are few datasets for omnidirectional stereo vision. Won et al. [6] published the datasets *OmniThings* and *OmniHouse* for training the inverse distance of a scene. These datasets present images from a system of four cameras with not aligned viewing directions. In contrast, our system is based on a canonical stereo setup (aligned viewing directions). To the best of our knowledge, *THEOStereo* [8] is the only dataset with rendered samples for depth estimation with an canonical omnidirectional stereo setup. Therefore, all

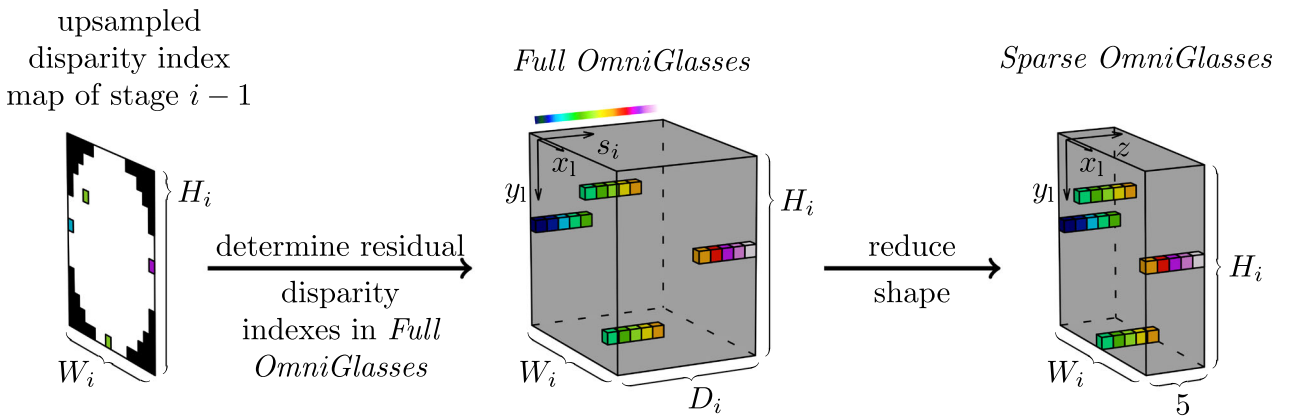


Fig. 3 Shape reduction of *OmniGlasses*. The disparity index maps of the stages 1 and 2 are upsampled to $H_i \times W_i$, with $i \in \{2, 3\}$. Each disparity index is used to determine five residual disparity indexes s_i. These indexes constitute the color-coded positions of five small cubes per pixel along the s_i-axis in a *Full OmniGlasses* LUT. Note that the center cube of each sequence in *Full OmniGlasses* has the same color as the disparity map value of the corresponding pixel. Each small cube denotes a look up tuple $(\hat{x}_\mathrm{r}, \hat{y}_\mathrm{r})$. In *Sparse OmniGlasses*, only these five cubes / look up tuples are stored for each pixel

Table 1 Shapes of *OmniGlasses* and disparity index mapping for stages 1–3 in *Omni-AnyNet*

Stage i	$H_i/H = W_i/W$	Index for $\hat{\delta}_{\mathrm{omni,max}}$	Highest index	Shape of *Full OmniGlasses*	Shape of *Sparse OmniGlasses*
1	1/16	11	11	$12 \times H_1 \times \cdot W_1 \times 2$	*not necessary*
2	1/8	$11 \cdot 2 = 22$	$11 \cdot 2 + 2 = 24$	$25 \times H_2 \times \cdot W_2 \times 2$	$5 \times H_2 \times W_2 \times 2$
3	1/4	$22 \cdot 2 = 44$	$24 \cdot 2 + 2 = 50$	$51 \times H_3 \times \cdot W_3 \times 2$	$5 \times H_3 \times W_3 \times 2$

The index reflecting the maximum disparity $\hat{\delta}_{\mathrm{omni,max}}$ in stages 1–3 is twice times as big as the corresponding index of the previous stage. However, a maximum residual value of $+2$ can theoretically further increase the estimable disparity as the index is higher than that of the $\hat{\delta}_{\mathrm{omni,max}}$. In *AnyNet*, no LUTs and hence no *OmniGlasses* are required in the last stage

evaluated networks in this work are trained and tested with *THEOStereo*. *THEOStereo* comprises 31,250 stereo image pairs together with their ground truth depth maps (distance in z-direction). Images and depth maps were rendered using the Unity3D game engine. As Unity3D does not provide shaders for omnidirectional camera models, the authors of [8] used the handcrafted shaders of [26], which merge four perspective images or depth maps according to the fusion method of Bourke et al. [27, 28]. In addition to RGB images, the shaders generated relative depth values between zero and one, which were then scaled to the given absolute distance (z-direction). For the experiments on *(Omni-)AnyNet* and *OmniMVS*$^+$, these depth maps are used as ground truth by first converting them to point clouds and then to Euclidean distance and disparity maps. Training, validation and testing subsets are partitioned in a ratio of 80%/10%/10%. We downsampled *THEOStereo's* images and ground truth to $H \times W = 1024 \times 1024$ pixels.

For a proof of concept of the LUT as a standalone module (without CNN layers), we first built up a *Full OmniGlasses* LUT of shape $201 \times H \times W \times 2$, with $H \times W$ given by the full image resolution. We reduced the shape to $1 \times H \times W \times 2$ by using the ground truth disparity. The transformations for each pixel of the right image given by this optimal version of the LUT are based in the correct disparity value. The right image is then transformed with the help of this LUT and compared with the left image. A high agreement by this comparison indicates that the transformation proposed by *Full OmniGlasses* as described in Sect. 3 is correct.

The quantitative evaluation involves error measurements on both disparity and Euclidean distance. All the output maps (Li's disparity, perspective disparity or inverse distance) of the considered approaches were converted into Euclidean distance maps to facilitate a comparison between them. The mean absolute error (MAE) is calculated by averaging the L_1 norm of each error. For perspective images, the bad-e error (abbreviated by $\Delta > e$) describes the ratio of disparity errors greater than e pixels along the epipolar line [29]. This error metric is, however, not directly applicable on omnidirectional images. In this case, we defined the bad-e error in relation to the disparity index. For omnidirectional images, $\Delta > e$ describes the ratio of disparities errors $\epsilon_i(x_1, y_1) = \frac{1}{S_i}\|\hat{\delta}'^i(x_1, y_1) - \delta(x_1, y_1)\|$ that exceeds e disparity indices, where $\hat{\delta}'^i(x_1, y_1)$ is the final estimated disparity of stage i (Conf. Sect. 3.1.3) and $\delta(x_1, y_1)$ is the ground truth

disparity. Here, we upsampled $\hat{\delta}'^{i}(x_1, y_1)$ to match the resolution of $\delta(x_1, y_1)$. The use of the disparity index (given by the division of the disparity values by the sampling step size S_i) by the error calculation allows to evaluate the performance of the network regardless of the predefined sampling rate given by S_i. The three pixel error (3PE) adds a new constraint to the bad-3 error, considering that also the relative error between the estimated disparity and the ground truth should exceed a certain threshold, in this case 5%. We used a 3PE analogous to Wang et al. [4] which is similar to that of *KITTI Stereo Dataset 2015* [30, 31]. Our 3PE for omnidirectional images is defined as follows:

$$E_{3\mathrm{PE}_i} = \frac{1}{N_i} \sum_{x_1, y_1} f_i(x_1, y_1), \tag{14}$$

with:

$$f_i(x_1, y_1) = \begin{cases} 1 \quad , & \text{if } \epsilon_i(x_1, y_1) > 3 \ \& \\ & \epsilon_i(x_1, y_1) / \frac{\delta(x_1, y_1)}{S_i} > 5\% \\ 0 \quad , & \text{otherwise} \end{cases} \tag{15}$$

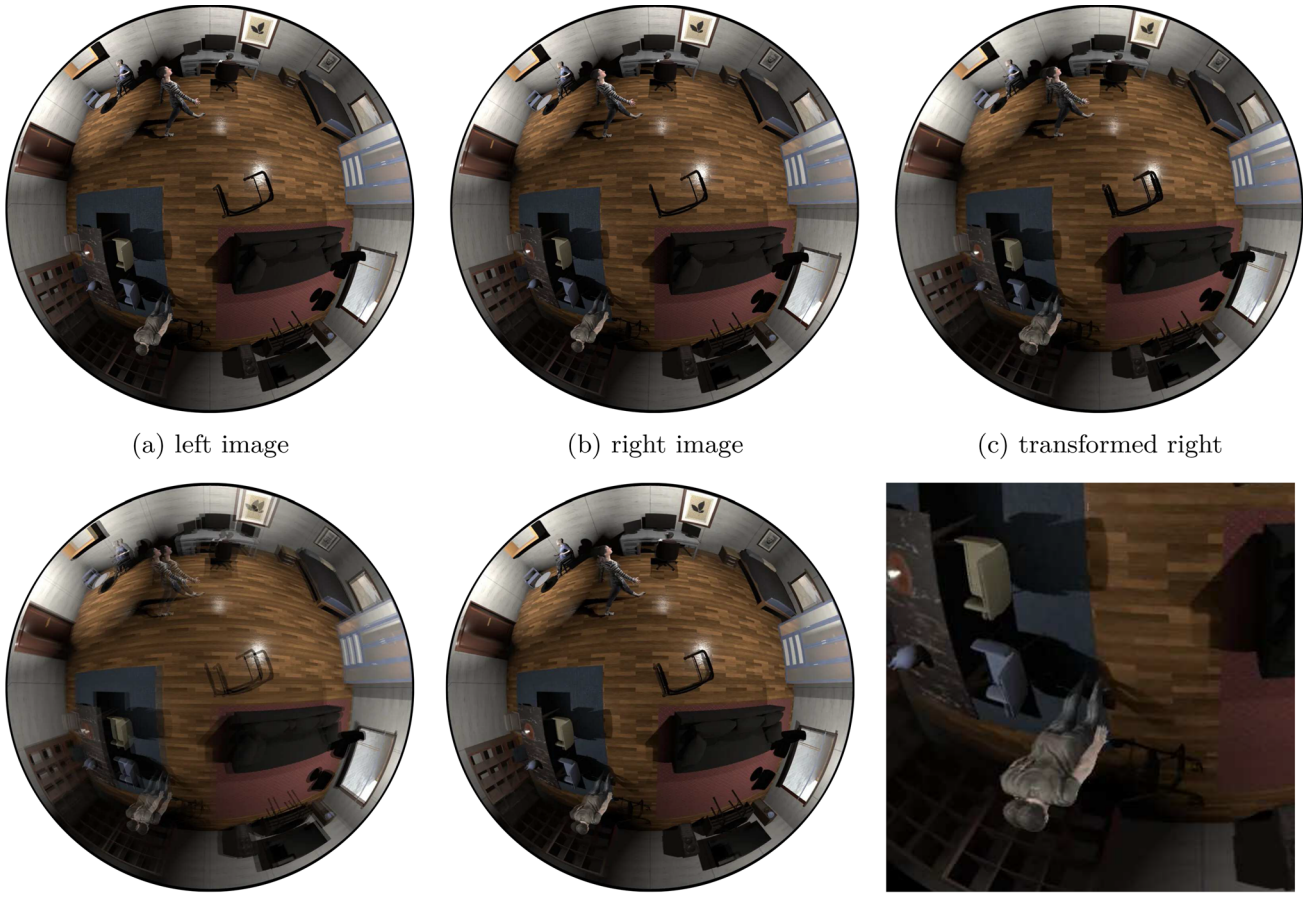
For all considered metrics, 3PE, bad-e and mean absolute error (MAE), only valid regions are considered, which reduced the pixel number from $H_i \cdot W_i$ to N_i. For sake of comparability between approaches and stages of *Omni-AnyNet*, some pixels of the results are discarded in the evaluation of disparity maps. This is the case, for example, for those pixels whose final disparity exceeds the index $H_i / H_1 \cdot (D_1 - 1)$. This allows to ignore those estimated disparities values that, because of the residual strategy of *AnyNet*, exceed the maximum disparity value $\hat{\delta}_{\mathrm{omni,max}}$ defined for the setup. Unlike the evaluation strategy followed by the original version of *AnyNet*, those pixels that do not belong to regions captured by both cameras (and therefore do not present valid values for stereo methods) are also discarded in the whole quantitative evaluation. This causes pixels to be ignored, mostly along the border of the FOV, but avoids including monocularly estimated values in the evaluation.

We integrate *OmniGlasses* into *AnyNet* as described in Sect. 3.2 and train *Omni-AnyNet* for 300 epochs on the training set of *THEOStereo*. As *THEOStereo* provides depth maps as ground truth, we convert them to disparity maps. Adam [32] with default parameters ($\beta_1 = 0.9$ and $\beta_2 = 0.999$) is chosen as an optimizer. The training is carried out with a learning rate of $1 \cdot 10^{-3}$ that decays to zero at the end of the training following the cosine annealing strategy [33] with a single decay period (no warm restarts). A smooth $L1$ loss with a threshold of 2.0 serves as training loss function for each stage. The final loss constitutes a weighted sum of the loss values of all stages with the same weights used for the original *AnyNet* implementation: 0.25 for the first, 0.5 for the second and 1 for the third and fourth stage.

For the ablation study, we train *AnyNet* without *OmniGlasses* using the same described hyperparameters. Here, we distinguish between *AnyNet* trained on omnidirectional disparities after [22, 23] and *AnyNet* trained on perspective disparities arranged in an omnidirectional manner analog to [8]. Both experiments use the original architecture of *AnyNet*, but with different ground truth arrangements. We refer to the first model as *AnyNet(Li)* and the second one as *AnyNet(Persp.)*. The number of evaluated pixels N_i may slightly vary for the different approaches: *AnyNet(Persp.)*, *AnyNet(Li)* and *Omni-AnyNet*. For sake of comparability a joint mask is applied during the evaluation to calculate the error maps only on that regions that are valid for all three approaches.

OmniMVS$^+$ was designed to reconstruct a 3D scene from four cameras with four different viewing directions. The number of cameras can hardly be changed in the official implementation without changing the architecture significantly. Therefore, we fed the stereo image pair from *THEOStereo* into *OmniMVS*$^+$ and kept the images of the remaining two cameras black. As the training routine was not provided by the authors[1], we built up a standard training pipeline on PyTorch. We excluded the entropy loss as no such training code was provided by the authors. However, it turned out that *OmniMVS*$^+$ still produces accurate results without this optimization as demonstrated in the next section. We split the network into two parts after Layer `conv4-11` of the Unary Feature Extractor (See Table 1 of [7]) and run the network on two GPUs in sequence. The first part was executed by an NVIDIA GeForce GTX 1080, the remaining part was processed on an NVIDIA Quardo P6000. The high GPU-RAM utilization (29132 MiB / 32768 MiB) did not allow to train the network on only one of our training GPUs nor to increase batch size or the resolution of the inverse distance maps. Hence, we kept a batch size of one and the original resolution of 160×640 of the distance maps. However, the same input images with resolution 1024×1024 were fed into *OmniMVS*$^+$. We used *OmniMVS*$^+$ with interleaved spheres which was essential to reduce the memory consumption and allows training on the mentioned GPUs. Furthermore, we chose the default number of channels, i.e., 32. We used the weights obtained from the pretraining on *OmniThings* [7] to initialize the network. The minimum distance parameter of *OmniMVS*$^+$ was adjusted to the minimal observable distance in *THEOStereo*, i.e., 0.78 AU. The inference and training times obtained for *OmniMVS*$^+$ are approx. $0.8 - 2.0$ fps (See Table 5) and 0.3 fps, respectively. As a result, it was not feasible to train the network for 300 epochs like for *Omni-AnyNet*. Our learning rate schedule for *OmniMVS*$^+$ therefore imitates the original schedule depending on the number of processed samples rather then the

[1] https://github.com/hyu-cvlab/omnimvs-pytorch.



(a) left image (b) right image (c) transformed right

(d) superposition left and right image (e) superposition left and transformed right image (f) occlusion artifacts, excerpt of (c)

Fig. 4 Proof of concept: View synthesis on RGB images via *Omni-Glasses*. The right image of a sample of *THEOStereo* is transformed using the proposed LUT and the disparity values of the ground truth. A superposition of the original left (**a**) and right image (**b**) shows the presence of disparity as a blurred effect (**d**). In contrast, by superposing the left image (**a**) and transformed right image (**c**) the location of the objects in both image coincides and the results looks sharper (**e**). This evidences that the coordinate transformations for the given disparities in the proposed LUTs are correct. As shown in (**f**), occlusion artifacts cannot be avoided. Here, part of the person's shape appears twice. The images in this figure rely on a sample of the *THEOStereo* dataset [8]

Table 2 Full Evaluation of *Omni-AnyNet*. Metrics relying on the Euclidean distance are given in arbitrary units of the *THEOStereo* dataset whereas 1 AU $\approx$ 50 cm. Disparity values are given in radians

Network	$\Delta > 1$	$\Delta > 2$	$\Delta > 4$	3PE	MAE disp. index		MAE Euc. dist.	
					abs.	rel.	abs.	rel.
Omni-AnyNet	13.72%	3.77%	1.02%	1.68%	0.59	3.27%	0.25	3.25%

processed epochs. In [7], *OmniMVS*$^+$ was trained for 20 epochs on *OmniThings* with a learning rate of $3 \cdot 10^{-3}$ and for further 10 epochs with a learning rate of $3 \cdot 10^{-4}$. A training for 20 or 30 epochs on *OmniThings* roughly processes as much training samples as in seven or 11 epochs on *THEOStereo*. Hence, we trained *OmniMVS*$^+$ for seven epochs with the initial learning rate of $3 \cdot 10^{-3}$ and for further four epochs with the reduced learning rate of $3 \cdot 10^{-4}$. In order to compare the results of *Omni-AnyNet* and *OmniMVS*$^+$, the error maps should coincide in their projection model as well as in their resolution. With this objective, the error maps of *Omni-AnyNet* were converted to the ERP model with the same resolution as in *OmniMVS*$^+$. Analogue to our ablation study, we masked out regions in the error maps of *Omni-AnyNet* and *OmniMVS*$^+$, that are not valid in both approaches. Due to the high memory footprint, it was not feasible to train both networks *Omni-AnyNet* and *OmniMVS*$^+$ under equal conditions. Hence, the juxtaposition of both approaches can only be seen as a coarse comparison.

Table 3 Comparison of *Omni-AnyNet* and *AnyNet*. Absolute metrics are given in arbitrary units of *THEOStereo* (1 AU ≈ 50 cm)

Network		MAE Euc. dist.	
	disp.	abs.	rel. (%)
Omni-AnyNet	omni	0.25	3.16
AnyNet(Li)	omni	0.33	4.38
AnyNet(Persp.)	persp	0.46	6.34

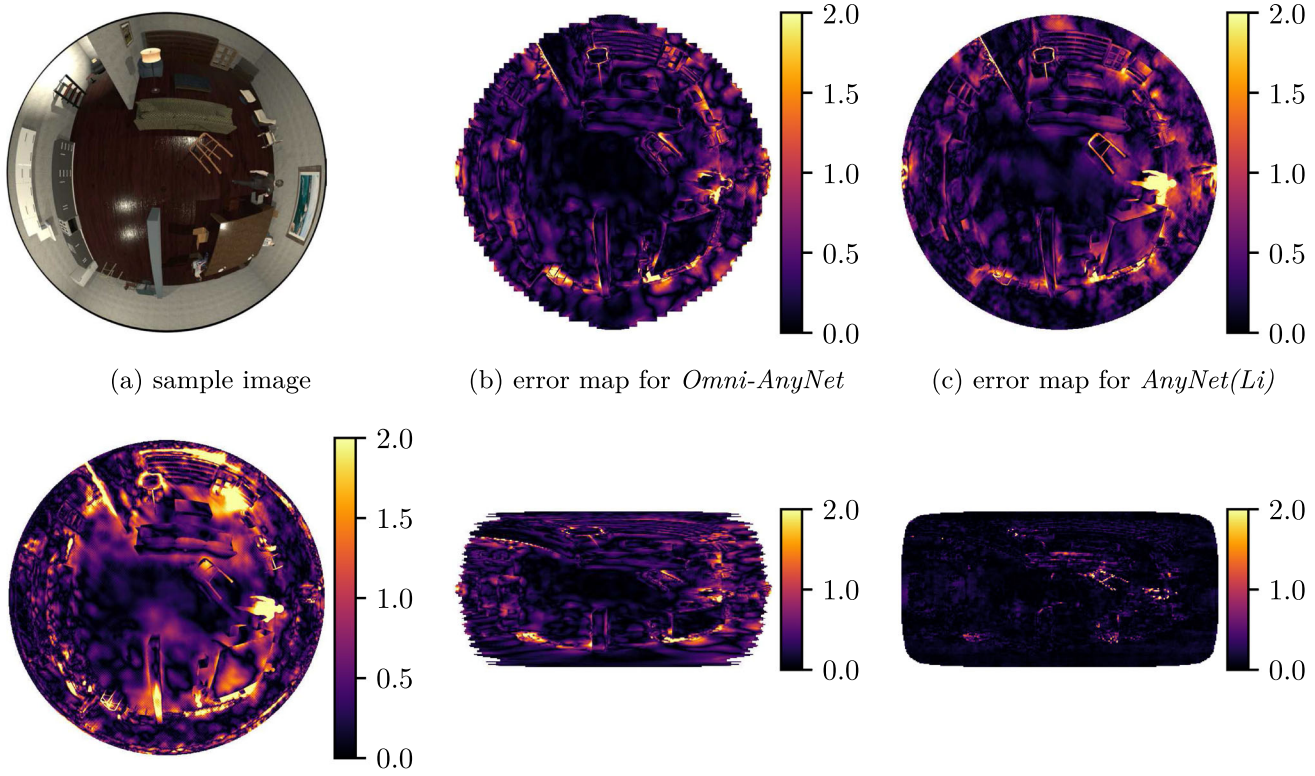
Table 4 Comparison of *Omni-AnyNet* and *OmniMVS*$^+$. Absolute metrics are given in arbitrary units of *THEOStereo* (1 AU ≈ 50 cm)

Network	MAE Euc. dist.	
	abs.	rel. (%)
Omni-AnyNet	0.24	2.98
OmniMVS$^+$	0.12	0.56

5 Results

Figure 4 depicts our proof of concept. In contrast to a simple superposition of left and right image (See Fig. 4d), the superposition of the left and transformed right image (See Fig. 4e) appears significantly sharper. This indicates that both the left and the transformed right image mainly coincide, which signalizes that the view synthesis was successful. However, some occlusion artifacts are visible in the transformed right image, which cannot be diminished by *OmniGlasses* as a standalone module without CNN layers. Figure 4f zooms one of this artifacts, where a part of the person's shape appears a second time on the left side of the person. For this particular image area, the left camera captures a part of the floor and wall shelves. However, this part of the background is occluded by the person for the right camera. This occluding texture is then copied instead of the floor or wall shelves texture from the right image to the transformed version where the mainly the floor texture has been expected. As a consequence, the person partially appears a second time.

Table 2 shows the result of *Omni-AnyNet* on the testing partition of *THEOStereo*. The bad-*e* error is remarkably low. The MAE for the disparity index as well as the MAE for the Euclidean distance give very satisfying results, that are inside the tolerance ranges for many applications. It can be



(a) sample image (b) error map for *Omni-AnyNet* (c) error map for *AnyNet(Li)*

(d) error map for *AnyNet(Persp.)* (e) ERP error map for *Omni-AnyNet* (f) ERP error map for *OmniMVS*$^+$

Fig. 5 Comparison of *Omni-AnyNet* with *AnyNet(Li)*, *AnyNet(Persp.)* and *OmniMVS*$^+$ showing the absolute error maps that correspond to the Euclidean distance for a sample of *THEOStereo* [8] (**a**). Figures (**b–d**) show qualitative results of our ablation study. *AnyNet(Li)* produces better results on omnidirectional disparity values (**c**) than *AnyNet(Persp.)* (**d**). However, *Omni-AnyNet* (**b**) produces more promising results. On the other hand, the high throughput of *Omni-AnyNet* comes with the cost of accuracy if compared with *OmniMVS*$^+$ (Conf. (**e**) and (**f**))

Table 5 Throughput measurements on *THEOStereo* during inference (batch size 1). All measurements are given in frames per second

	Quad. P6000	Titan RTX	GTX 1080
Omni-AnyNet	37.6	48.4	32.4
AnyNet(Li)	40.0	48.4	37.7
AnyNet(Persp.)	40.8	49.0	40.6
OmniMVS$^+$	1.2	2.0	0.8

seen that the estimated disparity index does not differ more than one step from the ground truth in average. An MAE for the Euclidean distance of 0.25 AU represents an error of around 12.5 cm in *THEOStereo*.

Table 3 summarizes our ablation study. As the disparity metrics differ between the proposed algorithms, only the absolute and the relative MAE of the Euclidean distance have been chosen for comparison. The errors are averaged over the dataset. As aforementioned, in each output sample we mask out regions that are not valid for all the three approaches. It can be seen that the incorporation of Li's disparity into *AnyNet* (*AnyNet(Li)*) considerably increases the performance in comparison with *AnyNet* using perspective disparity values arranged in an omnidirectional geometry (*AnyNet(Persp.)*). Moreover, *Omni-AnyNet*, which replaces the original *AnyNet's* look up tables with the proposed *OmniGlasses*, significantly reduced the absolute MAE by around 0.08 AU to 0.25 AU. In order to visualize these results, Fig. 5b–d present maps of the absolute errors by measuring the Euclidean distance, where the brighter the color the higher the error. It can be seen that *Omni-AnyNet* produces an error map with only a few bright spots indicating high MAEs (See Fig. 5b). Moreover, these bright spots are located nearby edges or fine objects like the wheeled walker, which might be explained by (self) occlusion artifacts. *AnyNet(Li)* has a lower performance than *Omni-AnyNet*, which is specially visible by the standing human's shape in the image. Finally, *AnyNet(Persp.)* results in larger high-error spots.

To complete our comparison, we present the results of *OmniMVS*$^+$. As this network uses the EPR model to present their results, Therefore, in order to facilitate the comparison, Figs. 5e and 5f show the outputs of *Omni-AnyNet* and *OmniMVS*$^+$ under ERP, respectively. *OmniMVS*$^+$ produces more accurate Euclidean distance maps than *Omni-AnyNet* (See Table 4), however, at the price of much more computational time (See Table 5). The pixelated borders of the estimation of *Omni-AnyNet* in Figs. 5b and 5e stem from intentionally deactivating monocular disparity estimation as mentioned in Sect. 4.

The throughput measurements for all discussed networks are documented in Table 5. *Omni-AnyNet* exhibits high frame rates of up to 48.4 fps. In contrast, *OmniMVS*$^+$, with maximum 2 fps, is an order of magnitude slower than *Omni-AnyNet*.

Experiments on the NVIDIA GTX 1080 as well as the NVIDIA Quadro P6000 were conducted on a deep learning machine with an Intel® Core™ i7-6900K CPU @ 3.20GHz (8 cores, 16 threads). The experiments on the NVIDIA TITAN X were performed on a second deep learning workstation with an Intel® Core™ i9-9960X CPU @ 3.10GHz (16 cores, 32 threads). Both machines have 128 GiB of RAM. It should be noted that the original version of AnyNet [4] achieved 10 fps on images with a resolution of 1242 × 375 on an NVIDIA Jetson TX2. This, together with the moderate RAM, GPURAM and CPU utilization of *Omni-AnyNet*, indicates that it is also suitable for real-time inference in embedded systems, delivering high quality results.

6 Conclusion and future work

In this work, we derive a search space for stereo correspondences in omnidirectional image pairs captured by canonical stereo cameras. We plan to extend *OmniGlasses* to consider other projection models for real-world lenses by refining the constraints for epipolar geometry, in particular Eq. 8. Moreover a corresponding search strategy, similar to Meuleman et al. [16], is proposed. These derive in a set of LUTs named *OmniGlasses*, which can be easily combined with machine learning methods, like neural netowrks. In contrast to [7] and [16], *OmniGlasses* search for the disparity instead of a distance or inverse distance. It is therefore, to some extent, reminiscent of classical stereo vision retrieving disparity values rather than estimating properties of the scene (the distance of 3D points to the camera) directly. We concentrated on a canonical camera system. This system maximizes the area of the scene that is visible by both cameras and is therefore optimal for binocular stereo setup. We integrated *OmniGlasses* into *AnyNet* and proved the efficiency of our approach. We call the resulting network *Omni-AnyNet*. This achieves remarkable reconstruction results with a low MAE of around 13 cm (Euclidean distance) at up to 48.4 fps and outperforms *OmniMVS*$^+$, a state-of-the-art CNN for depth reconstruction with omnidirectional images, in terms of speed. As a consequence, *OmniGlasses* successfully diminish the gap between reconstruction accuracy and high throughput rates. As a large number of networks for perspective stereo vision like *AnyNet* exist, we believe that *OmniGlasses* can open up many opportunities to develop fast networks for omnidirectional vision. We derived *OmniGlasses* for the equiangular projection model. This model could be replaced by camera models more suitable for real-world images in the future.

Acknowledgements We gratefully acknowledge the donation of a Quadro P6000 graphics card by NVIDIA Corporation for training and evaluation purposes. This work is funded by the European Regional Development Fund (ERDF) as well as the Free State of Saxony under the grant number 100-241-945.

Funding Open Access funding enabled and organized by Projekt DEAL. NVIDIA Corporation donated a NVIDIA Quadro P6000 for scientific purposes within the NVIDIA GPU Grant Program. This work is funded by the European Regional Development Fund and the Free State of Saxony under the grant number 100-241-945.

Data availability Fig. 1, 4 and 5 contain images that are based on samples of the *THEOStereo* dataset [8]. This dataset was published under the CC-BY 4.0 license (See https://creativecommons.org/licenses/by/4.0/).

Code Availability We published our code at https://github.com/hamza9305/Omni-AnyNet and https://github.com/hamza9305/OmniGlasses.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose. The authors have no Conflict of interest to declare that are relevant to the content of this article.

References

1. Findeisen, M., Hirtz, G.: Trinocular spherical stereo vision for indoor surveillance. In: 2014 Canadian Conference on Computer and Robot Vision, pp. 364–370. IEEE, New York (2014). https://doi.org/10.1109/CRV.2014.56
2. Findeisen, M., Meinel, L., Hirtz, G.: A trinocular omnidirectional stereo vision system for high-precision RGB-D acquisition. In: Proceedings ELMAR-2014, pp. 1–4. IEEE, New York (2014). https://doi.org/10.1109/ELMAR.2014.6923350
3. Findeisen, M., Meinel, L., Richter, J., Hirtz, G.: An omnidirectional stereo sensor for human behavior analysis in complex indoor environments. In: 2015 IEEE international conference on consumer electronics (ICCE), pp. 17–19. IEEE, New York (2015). https://doi.org/10.1109/ICCE.2015.7066302
4. Wang, Y., Lai, Z., Huang, G., Wang, B.H., van der Maaten, L., Campbell, M., Weinberger, K.Q.: Anytime stereo image depth estimation on mobile devices. In: 2019 International Conference on Robotics and Automation (ICRA), pp. 5893–5900. IEEE, New York (2019). https://doi.org/10.1109/ICRA.2019.8794003
5. Won, C., Ryu, J., Lim, J.: SweepNet: wide-baseline omnidirectional depth estimation. In: 2019 International Conference on Robotics and Automation (ICRA), pp. 6073–6079. IEEE, New York (2019). https://doi.org/10.1109/ICRA.2019.8793823
6. Won, C., Ryu, J., Lim, J.: OmniMVS: end-to-end learning for omnidirectional stereo matching. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 8986–8995. IEEE, New York (2019). https://doi.org/10.1109/ICCV.2019.00908
7. Won, C., Ryu, J., Lim, J.: End-to-end learning for omnidirectional stereo matching with uncertainty prior. IEEE Trans. Pattern Anal. Mach. Intell. **43**(11), 3850–3862 (2021). https://doi.org/10.1109/TPAMI.2020.2992497
8. Seuffert, J.B., Perez Grassi, A.C., Scheck, T., Hirtz, G.: A Study on the Influence of Omnidirectional Distortion on CNN-based Stereo Vision. In: Proceedings of the 16th international joint conference on computer vision, imaging and computer graphics theory and applications, VISIGRAPP 2021, Volume 5: VISAPP, pp. 809–816. SciTePress, Setúbal (2021). https://doi.org/10.5220/0010324808090816
9. Hirschmüller, H.: Accurate and efficient stereo processing by semi-global matching and mutual information. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 2, pp. 807–814. IEEE, New York (2005). https://doi.org/10.1109/CVPR.2005.56
10. Wang, N.-H., Solarte, B., Tsai, Y.-H., Chiu, W.-C., Sun, M.: 360SD-Net: 360° Stereo depth estimation with learnable cost volume. In: 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 582–588. IEEE, New York (2020). https://doi.org/10.1109/ICRA40945.2020.9196975
11. Komatsu, R., Fujii, H., Tamura, Y., Yamashita, A., Asama, H.: 360° Depth estimation from multiple fisheye images with origami crown representation of icosahedron. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 10092–10099. IEEE, New York (2020). https://doi.org/10.1109/IROS45743.2020.9340981
12. Córdova-Esparza, D.-M., Terven, J., Romero-González, J.-A., Ramírez-Pedraza, A.: Three-dimensional reconstruction of indoor and outdoor environments using a stereo catadioptric system. Appl. Sci **10**(24), 85 (2020). https://doi.org/10.3390/app10248851
13. Revaud, J., Weinzaepfel, P., Harchaoui, Z., Schmid, C.: DeepMatching: hierarchical deformable dense matching. Int. J. Comput. Vision **120**(3), 300–323 (2016). https://doi.org/10.1007/s11263-016-0908-3
14. Lee, J., Park, D., Lee, D., Ji, D.: Semi-supervised 360° depth estimation from multiple fisheye cameras with pixel-level selective loss. In: ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2290–2294. IEEE, New York (2022). https://doi.org/10.1109/ICASSP43922.2022.9746232
15. Li, M., Hu, X., Dai, J., Li, Y., Du, S.: Omnidirectional stereo depth estimation based on spherical deep network. Image Vis. Comput, **114**, 85 (2021). https://doi.org/10.1016/j.imavis.2021.104264
16. Meuleman, A., Jang, H., Jeon, D.S., Kim, M.H.: Real-time sphere sweeping stereo from multiview fisheye images. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 11418–11427. IEEE, New York (2021). https://doi.org/10.1109/CVPR46437.2021.01126
17. Konolige, K.: Small vision systems: hardware and implementation. In: Robotics Research - The Eighth International Symposium, pp. 203–212. Springer, London (1998). https://doi.org/10.1007/978-1-4471-1580-9_19
18. Tonioni, A., Tosi, F., Poggi, M., Mattoccia, S., Stefano, L.D.: Real-time self-adaptive deep stereo. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 195–204. IEEE, New York (2019). https://doi.org/10.1109/CVPR.2019.00028
19. Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., Bry, A.: End-to-end learning of geometry and context

for deep stereo regression. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 66–75. IEEE, New York (2017). https://doi.org/10.1109/ICCV.2017.17
20. Lipson, L., Teed, Z., Deng, J.: RAFT-stereo: multilevel recurrent field transforms for stereo matching. In: 2021 International Conference on 3D Vision (3DV), pp. 218–227. IEEE, New York (2021). https://doi.org/10.1109/3DV53792.2021.00032
21. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) Medical image computing and computer-assisted intervention - MICCAI 2015. Lecture Notes in Computer Science, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
22. Li, S.: Real-time spherical stereo. In: 18th International Conference on Pattern Recognition (ICPR'06), vol. 3, pp. 1046–1049. IEEE, New York (2006). https://doi.org/10.1109/ICPR.2006.968
23. Li, S.: Trinocular spherical stereo. In: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4786–4791. IEEE, New York (2006). https://doi.org/10.1109/IROS.2006.282350
24. Kannala, J., Brandt, S.S.: A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. IEEE Trans. Pattern Anal. Mach. Intell. **28**(8), 1335–1340 (2006). https://doi.org/10.1109/TPAMI.2006.153
25. Liu, S., Mello, S.D., Gu, J., Zhong, G., Yang, M.-H., Kautz, J.: Learning affinity via spatial propagation networks. In: Guyon, I., Luxburg, U.v., Bengio, S., Wallach, H.M., Fergus, R., Vishwanathan, S.V.N., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, vol. 30, pp. 1520–1530. Curran Associates, Inc., Red Hook (2017)
26. Scheck, T., Seidel, R., Hirtz, G.: Learning from THEODORE: a synthetic omnidirectional top-view indoor dataset for deep transfer learning. In: 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), Snowmass Village, Colorado, USA, pp. 932–941 (2020). https://doi.org/10.1109/WACV45572.2020.9093563
27. Bourke, P.: iDome: Immersive gaming with the Unity game engine. In: CGAT 09: computer games. Multimedia and Allied Technology 09: Proceedings, pp. 265–272. Research Publishing Services, Singapore (2009)
28. David Bourke, P., Quintanilha Felinto, D.: Blender and immersive gaming in a hemispherical dome. GSTF Int J Comput **1**(1), 280–284 (2010)
29. Scharstein, D., Szeliski, R., Zabih, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In: Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001), pp. 131–140. IEEE, New York (2001). https://doi.org/10.1109/SMBV.2001.988771
30. Menze, M., Heipke, C., Geiger, A.: Joint 3D estimation of vehicles and scene flow. ISPRS Ann. Photogr. Remote Sens. Spat. Inf. Sci. **2**, 427–434 (2015). https://doi.org/10.5194/isprsannals-II-3-W5-427-2015
31. Menze, M., Heipke, C., Geiger, A.: Object scene flow. ISPRS J. Photogramm. Remote. Sens. **140**, 60–76 (2018). https://doi.org/10.1016/j.isprsjprs.2017.09.013
32. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings, San Diego, CA, USA (2015). https://doi.org/10.48550/arXiv.1412.6980
33. Loshchilov, I., Hutter, F.: SGDR: stochastic gradient descent with warm restarts. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings (2017). https://doi.org/10.48550/arXiv.1608.03983

Julian B. Seuffert studied computer science at the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU). He received his M.Sc. in 2017 and worked as a research assistant at FAU in the field of image forensics between April and September 2017. From December 2017 to December 2021, he was employed as a research assistant at Chemnitz University of Technology. Since then, he has been conducting research on omnidirectional stereo vision.

Ana C. Perez Grassi is a research assistant at Chemnitz University of Technology. She received her Ph.D. in Computer Vision from the Karlsruhe Institute of Technology. Her current research interests include image processing, deep learning, and information fusion.

Hamza Ahmed a computer vision enthusiast, focuses his research in the realms of 3D reconstruction, training/testing novel versions of object detection networks, and image processing. He holds a Master's degree from Chemnitz University of Technology in the field of Embedded Systems.

Roman Seidel is a postdoctoral researcher at the Faculty of Electrical Engineering and Information Technology at Chemnitz University of Technology. In his Ph.D. thesis he worked on fine-grained activity analysis of persons in fisheye images based on optical flow and on the generation of synthetic data. As a post-doc, he is currently working on the generation of semi-synthetic data with diffusion models.

Gangolf Hirtz studied electrical engineering at the University of Saarland, Germany. After receiving his Ph.D. from the University of Dortmund in 1989, he joined industry. Here he worked in the field of television technology and finally in the automotive industry before accepting a position at Chemnitz University of Technology in 2008. Since then, he has been a full professor at the Department of Digital and Circuit Technology (DST). His research areas are in the field of transmission technology and machine learning.