Next on Stage: 'MC ViSi' – a Machine Vision Simulation Framework

Johannes Meyer

Vision and Fusion Laboratory
Institute for Anthropomatics
Karlsruhe Institute of Technology (KIT), Germany
johannes.meyer@kit.edu

Technical Report IES-2016-06

Abstract: Machine vision systems are used in diverse kinds of industries. They are employed in automobiles for the detection and protection of pedestrians, in visual inspection systems to ensure the quality of produced goods etc. The parameters of such systems result in many degrees of freedom. Determining the optimal setting of these parameters usually represents a time-consuming task that has to be performed empirically. This article introduces a software framework consisting of multiple plugins for the physically based rendering suite Mitsuba. By means of combining the single framework components, the sensor signals of arbitrary machine vision systems can be simulated for synthetic scenes consisting of objects having complex 3D geometries and the design process of the whole system can be streamlined.

1 Introduction

In many industrial fields, thousands or even millions of high quality products are produced every day. In order to ensure the produced goods' quality, they are usually visually inspected. Often, the visual inspection is performed by human workers. However, since this is a fatiguing task for humans, they may oversee defects what could have disastrous consequences. Therefore there exists a huge demand for automated visual inspection systems.

A visual inspection system usually contains one or more sensors and illumination sources. In order to build an effective inspection system, the components have to be well adapted to the inspection problem on hand. For example, the focal length and the magnification of a camera lens, the size of the sensor, the angle, color and

intensity of the illumination sources and the relative placement and orientation of all components with respect to each other have to match in order to allow a successful processing of the acquired sensor images. Often, an expert determines the optimal parameters via time consuming experiments.

The procedure of adjusting the degrees of freedom of a visual inspection system can be streamlined using computer simulations. By employing accurate models of the employed sensors and light sources, a suitable physically based rendering framework and CAD models of the objects under test, the experiments can be simulated. Furthermore, quality metrics concerning the resulting sensor images, e.g., contrast, can be used to automatically evaluate batches of different parameter sets. In total, such a simulation framework is a potential basis of a method allowing to efficiently and accurately determine the optimal parameters of visual inspection systems.

This article presents *MC ViSi*, a novel bundle of plugins for the physically based rendering framework Mitsuba [Jak10]. Indeed, the Mitsuba framework already includes plugins like an entocentric camera, a telecentric camera, a collimated beam source etc.. However, the parameterization of these plugins is focused on computer graphics applications, e.g., by using terms like the field of view, focus distance and clipping planes. In contrast, for components employed in machine vision applications, parameters like the focal length, the image distance, the sensor element size, etc. are common. MC ViSi extends the Mitsuba framework by appropriately parameterized plugins modeling components of sensor and illumination classes that are often employed in machine vision systems for visual inspection. Particularly, the bundle includes concepts of sensors and light sources involved in light field imaging and processing. Recent research investigates the suitability of such components in novel approaches for the visual inspection of transparent objects [MLB16b, MLB16a, MGLB16]. For such applications, the direction of captured light rays can exhibit important information, e.g., about scattering defects present in a test object. Section 3 introduces the core components of the Mitsuba renderer and briefly explains how a sensor image is rendered.

2 Related work

The idea of using simulation frameworks to support the design of automated visual inspection systems has been adopted by different groups of researchers.

In [NZL16], Nürnberg et al. employed a rendering framework in order to optimize the parameters of a computational camera used for depth estimations.

The group of Mohammadikaji et al. [BM+16, MBI+16] proposed an uncertainty propagation framework for finding optimal arrangements in a laser triangulation setup by means of precisely simulating the laser's speckle effects and the reflectance properties of the investigated test object.

Irgenfried et al. [ITW11, IDW14] developed a software framework capable of optimizing the parameters of image processing routines involved in a visual inspection process. Therefore they simulated the inspection scene using both realtime and photorealistic renderers.

The suitability of different combinations of novel optical setups and subsequent image processing algorithms for finding defects in transparent materials has been shown by Meyer et al. by also employing a simulation framework [MGLB16, MLB16a, MLB16c, MLB16b].

3 Basic rendering framework

The plugins of the MC ViSi package are based on the Mitsuba rendering framework. Figure 3.1 shows the core components of Mitsuba and how they interact with each other. In order to render an image, the main component, i.e., the renderer, passes a sensor sample and, if required, also an aperture sample to the sensor component. For example, the aperture sample is needed by plugins modeling conventional cameras, where every pixel integrates incident light rays coming from multiple directions. The sensor component determines the ray of sight corresponding to the two samples. The renderer traces the ray of sight through the scene. All objects and light sources that should be rendered are contained in the scene. The ray of sight might get reflected, scattered or absorbed while being traced. Whenever the ray hits a light source, the renderer passes the 3D position of the intersection together with the direction of incidence to the respective component. The involved light source plugin determines and returns the spectrum and intensity of the light that it emits into the queried direction. The renderer calculates the final radiance corresponding to the current sensor sample by propagating the spectrum back along the traversed optical path and by taking the reflectance characteristics of all involved surfaces into account. The film component successively aggregates all pairs of sensor samples and radiances and transforms them into the final sensor image as soon as the rendering process has finished.

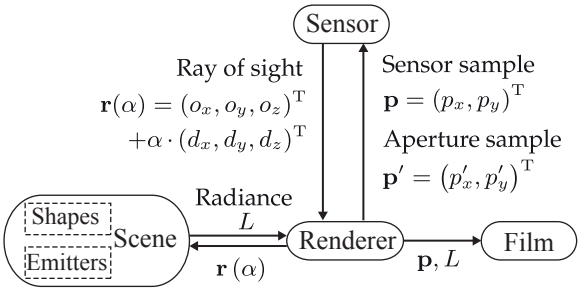

Figure 3.1: The core components of the Mitsuba renderer and how they interact with each other.

4 MC ViSi framework extensions

This section describes some of the major extensions to the Mitsuba renderer that are currently included in the MC ViSi framework.

4.1 Light sources

As mentioned in Sec. 3, a Mitsuba light source component has to provide one functionality: for a given intersection $\mathbf{q}_i$ of a traced ray of sight and the light source and the corresponding direction of incidence $\mathbf{q}_d$, there has to be a function that calculates the resulting spectrum of the emitted light. The following sections describe all the light sources contained in the proposed framework and explain how the spectra are calculated for a given query from the renderer.

4.1.1 Parallel emitter

The parallel emitter plugin realizes a telecentric light source. Such a light source emits light only into directions inside a cone having a certain angle β with respect to the surface normal $\mathbf{q}_n$ at the queried intersection $\mathbf{q}_i$ (see Fig. 4.1). The plugin has to be attached to an arbitrary shape (i.e., a geometric object) that provides the surface information. If the angle between the query direction $\mathbf{q}_d$ and the surface normal $\mathbf{q}_n$ is greater than β, a spectrum with all elements set to zero is returned—otherwise, the specified spectrum is returned.

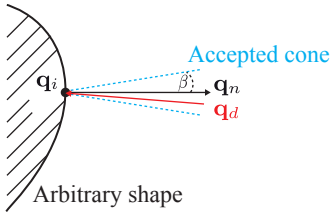

Figure 4.1: Schematic concept of the parallel emitter plugin: the red arrow denotes the queried direction $\mathbf{q}_d$ and $\mathbf{q}_n$ represents the surface normal at the queried surface point $\mathbf{q}_i$. Since the angle between $\mathbf{q}_d$ and $\mathbf{q}_n$ is less than the angle β of the accepted cone, light is emitted in the queried direction.

4.1.2 Telecentric projector

The telecentric projector plugin allows to simulate a spatially programmable area light source that consists of single pixels of the same size all emitting individual spectra. Figure 4.2 shows the principal setup of the plugin. The spectra can be conveniently defined by the user by providing an RGB image. The plugin has to be attached to a rectangular shape that determines the projector's overall size, position and orientation in the simulated scene. Similar to the parallel emitter plugin mentioned in the previous section, the pixels of the telecentric projector plugin emit light only inside a definable cone angle.

4.1.3 Light field emitter

Basically, the light field emitter extends the telecentric projector by also allowing to define individual spectra for different emission directions. By this means, the emission of a four dimensional light field can be simulated. This plugin is also attached to a rectangular 3D shape and also requires a an RGB image that provides the information about the emitted spectra. Additional parameters specify the light field emitter's spatial and angular resolution. The light field information is spatially multiplexed in the the RGB image as shown in Fig. 4.3.

For a given query by the renderer, at first the emitter's corresponding macro pixel is determined. Then, the intersection $\mathbf{i}$ of a virtual ray originating in the query direction from the origin $\mathbf{o}$ of the found macro pixel with a hemisphere of radius 1 located underneath the emitter plane is calculated. The position $\mathbf{i}$ is then projected on the emitter plane and the relative coordinates with respect to $\mathbf{o}$ allow to determine the angular component of the query, i.e., the actual spectrum to be

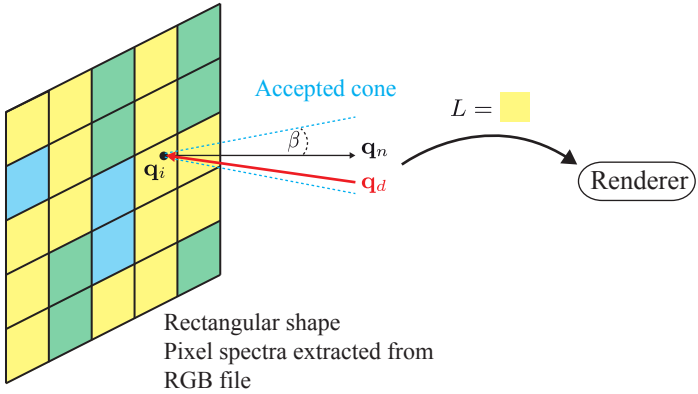

Figure 4.2: Schematic concept of the telecentric projector plugin: similar to the parallel emitter, every pixel of the simulated programmable area light source emits light inside a definable cone only. Since in the shown case the queried surface position lies inside a yellow pixel of the light source, the respective spectrum is returned to the renderer.

read from the RGB image and to be returned to the renderer. This concept is visualized in Fig. 4.4.

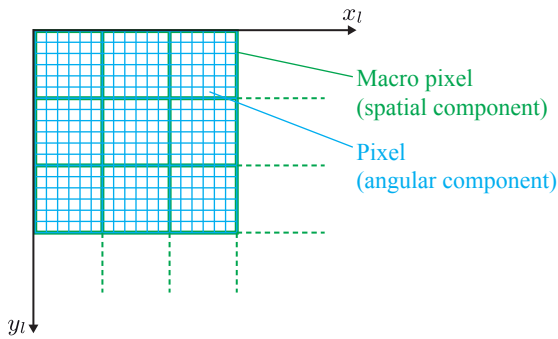

Figure 4.3: Spatial multiplexing of the light field emitter plugin for representing the spatial and angular component of the light field data in a two-dimensional RGB file. Single pixels are divided into macro pixels according to the specified angular resolution. The macro pixels carry the spatial component of the light field data and the underlying pixels represent the light field's angular component.

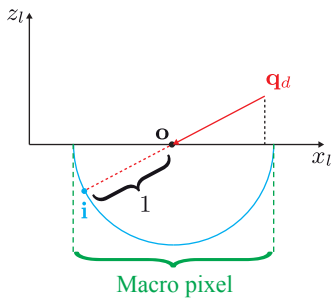

Figure 4.4: Schematic concept of the light field emitter: $\mathbf{o}$ denotes the center of the macro pixel, i.e., the spatial component of the light field data corresponding to the queried surface position and direction. The intersection $\mathbf{i}$ of the ray originating from $\mathbf{o}$ in direction $\mathbf{q}_d$ allows to obtain the correct pixel inside the determined macro pixel and to finally return the respective spectrum.

4.2 Sensors

Besides the described light source plugins, the MC ViSi framework also adds sensor plugins to the Mitsuba renderer. A sensor plugin has to provide a function that calculates the parameters of the ray of sight $\mathbf{r}(\alpha)$ corresponding to a pixel sample $\mathbf{p} = (p_x, p_y)^{\mathrm{T}}$ and—if applicable—an aperture sample $\mathbf{p}' = (p'_x, p'_y)^{\mathrm{T}}$. The following Sections are dedicated to MC ViSi's major sensor plugins and provide details about the respective steps needed to calculate the ray of sight.

4.2.1 Telecentric camera

Although Mitsuba already has a telecentric camera plugin, MC ViSi introduces its own implementation that uses types of parameters that are more common in the field of industrial machine vision. The provided plugin is based on a simple model of a telecentric camera as shown in Fig. 4.5. The required parameters are the sensor's focal length f, the image plane distance b, the pixel dimensions $\mathbf{l} = (l_x, l_y)$, the diameter D_t of the telecentric stop, the diameter D_l of the main lens and the number of pixels (M, N) in x- and y-direction.

For a given pixel sample $\mathbf{p}$ and an aperture sample $\mathbf{p}'$, the corresponding ray of sight $\mathbf{r}(\alpha)$ is calculated as follows: the samples $\mathbf{p}$ and $\mathbf{p}'$ are mapped to the respective positions $\mathbf{s} = (s_x, s_y)$ on the sensor plane and $\mathbf{a} = (a_x, a_y)^{\mathrm{T}}$ on the plane of the telecentric stop (i.e., the aperture plane) respectively. The intersection $\mathbf{o}$ of a ray running through $\mathbf{s}$ and $\mathbf{a}$ with the plane of the main lens denotes

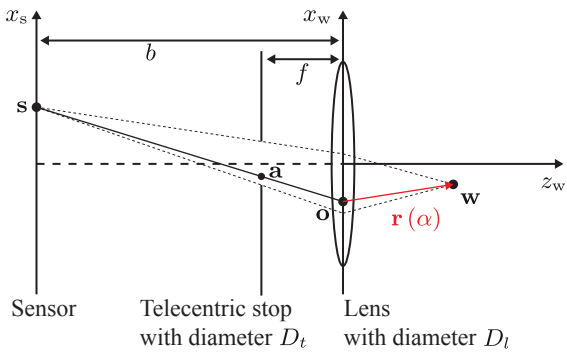

Figure 4.5: Schematic concept of the telecentric camera: visualization of the calculation of the ray of sight $\mathbf{r}(\alpha)$ corresponding to the position $\mathbf{s}$ on the sensor plane and the position $\mathbf{a}$ on the aperture plane.

the origin of the ray of sight. If $\mathbf{o}$ lies outside the main lens, the spectrum corresponding to the respective ray of sight will be weighted with zero. The focused world point $\mathbf{w} = (w_x, w_y, w_z)^{\mathrm{T}}$ can be calculated by means of the thin lens formula [BLF15]:

$$\frac{1}{f} = \frac{1}{w_z} + \frac{1}{b} \Leftrightarrow \left(\frac{1}{f} - \frac{1}{b}\right) \tag{4.1}$$

and the intercept theorem

$$\begin{pmatrix} w_x \\ w_y \end{pmatrix} = -\frac{1}{b} \begin{pmatrix} w_z s_x \\ w_z s_y \end{pmatrix} . \tag{4.2}$$

The direction $\mathbf{d} = (d_x, d_y, d_z)^{\mathrm{T}}$ of the ray of sight can be obtained via

$$\mathbf{d} = \mathbf{w} - \mathbf{o} .$$

The presented implementation also covers depth of field effect as it accounts for the size of the sensor elements and for the aperture diameter.

4.2.2 Light field camera

A light field camera is able to not only resolve the (x, y)-position of captured light bundles but also their two-dimensional direction of incidence (θ, φ). By most commercially available light field cameras this is achieved by replacing the

sensor by a micro lens array and by placing the actual sensor behind the micro lens array. The single elements of the micro lens array represent the system's spatial component. The micro lenses realize a mapping between the direction of the incoming light bundles and the sensor pixels that are located behind the respective micro lens. The set of pixels that correspond to one micro lens is usually called a macro pixel.

A light field camera of the described kind is modeled by the respective plugin of the MC ViSi framework. The plugin's parameters are the focal length f of the main lens, the image plane distance b, the diameter of the main lens D_l, the spatial resolution (M, N), the angular resolution (J, K) and the pixel dimensions $\mathbf{l} = (l_x, l_y)$.

In order to reduce the overall number of parameters, a more simplified light field camera model is used (see Fig. 4.6). In order to obtain the ray of sight $\mathbf{r}(\alpha)$

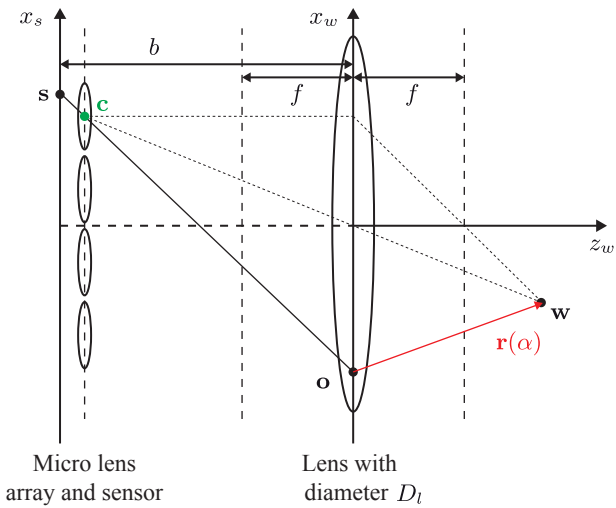

Figure 4.6: Schematic concept of the light field camera: visualization of the calculation of the ray of sight $\mathbf{r}(\alpha)$ corresponding to the position $\mathbf{s}$ on the sensor plane and the center position $\mathbf{c}$ of the respective micro lens.

corresponding to a continuous pixel sample $\mathbf{p}$, at first the center $\mathbf{c} = (c_x, c_y)^{\mathrm{T}}$ of the respective micro lens, i.e., the macro pixel is determined, to which $\mathbf{p}$ belongs. The focused world point $\mathbf{w} = (w_x, w_y, w_z)^{\mathrm{T}}$ is calculated via the thin

lens formula (4.1) and the intercept theorem (4.2) modified as

$$\begin{pmatrix} w_x \\ w_y \end{pmatrix} = -\frac{1}{b} \begin{pmatrix} w_z c_x \\ w_z c_y \end{pmatrix} .$$

All pixel positions of the same macro pixel are assumed to share the same focused world point $\mathbf{w}$, but to look at it from different directions. In the employed simplified model, the relative position of the pixel sample $\mathbf{p}$ with respect to its macro pixel is linearly mapped to the corresponding position $\mathbf{o}$ on the main lens:

$$\begin{pmatrix} o_x \\ o_y \\ o_z \end{pmatrix} = \begin{pmatrix} -\left(\frac{p_x \mathrm{mod} J}{J} \cdot 2 - 1\right) \cdot \frac{D}{2} \\ -\left(\frac{p_y \mathrm{mod} K}{K} \cdot 2 - 1\right) \cdot \frac{D}{2} \\ 0 \end{pmatrix} .$$

Finally, the ray of sight is given by

$$\mathbf{r}(\alpha) = \mathbf{o} + \alpha \cdot (\mathbf{w} - \mathbf{o}) .$$

5 Experiments

This section provides an example showing how two of the introduced plugins can be employed to simulate a novel machine vision system for the visual inspection of transparent objects as proposed in [MLB16b]. The system's principle optical setup is shown in Fig. 5.1. The parallel emitter plugin (see Sec. 4.1.1) can be used to simulate the collimated illumination and the light field camera plugin (see Sec. 4.2.2) allows to simulate the employed sensor. The described scene has been simulated for a defect-free test object instance and for a test object affected by a scattering material defect, i.e., an enclosed air bubble. In order to visually enhance material defects present inside the test objects, the authors propose the calculation of a special kind of image gradient that is based on the so-called Earth Mover's distance [RTG98] and that is suitable for light field images. Figure 5.2 shows the resulting gradient images visualized in pseudo colors.

Since the simulated experiments showed promising results for the method proposed in [MLB16b], the authors are currently setting up a real prototype for conducting further experiments.

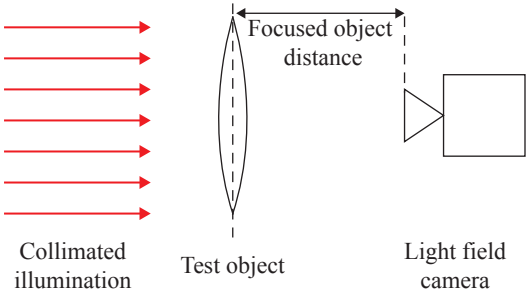

Figure 5.1: Optical setup of the simulated machine vision system: parallel light beams illuminate the test object, a double-convex lens, and a light field camera serves as the sensor.

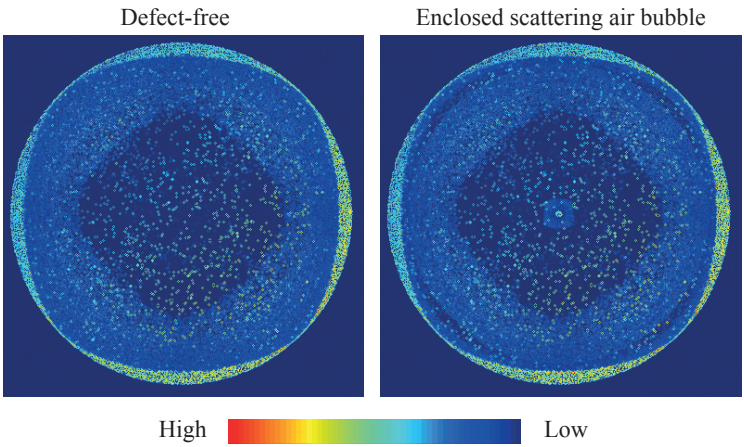

Figure 5.2: Pseudo color visualization of the output of the image gradient method based on the Earth Mover's distance applied to the light field images resulting from the simulation. In the right-hand side image, the scattering defect present in the test object's center is clearly visible.

6 Conclusion

This article introduced the machine vision simulation framework MC ViSi. The framework consists of several plugins that extend the physically based rendering software Mitsuba. Some of the proposed framework's main components, i.e., the

parallel emitter, the telecentric projector, the light field emitter, the telecentric camera and the light field camera have been explained in detail. Furthermore, an example has been provided, where the introduced plugins have been employed in a simulation of a machine vision system for the inspection of transparent objects.

As future steps, the authors plan to extend the framework by further components, e.g., a plugin allowing to model retroreflective object surfaces and a sensor plugin modeling a laser scanner sensor.

Bibliography

[BLF15] Jürgen Beyerer, Fernando Puente León, and Christian Frese. *Machine Vision - Automated Visual Inspection: Theory, Practice and Applications*. Springer Berlin Heidelberg, 2015.

[BM+16] Stephan Bergmann, Mahsa Mohammadikaji, , Stephan Irgenfried, Heinz Wörn, Jürgen Beyerer, and Carsten Dachsbacher. A phenomenological approach to integrating gaussian beam properties and speckle into a physically-based renderer. In Matthias Hullin, Marc Stamminger, and Tino Weinkauf, editors, *Vision, Modeling Visualization*. The Eurographics Association, 2016.

[IDW14] Stephan Irgenfried, Frank Dittrich, and Heinz Wörn. Realization and evaluation of image processing tasks based on synthetic sensor data: 2 use cases. In *Forum Bildverarbeitung 2014*, pages 35–46, 2014.

[ITW11] Stephan Irgenfried, Igor Tchouchenkov, and Heinz Wörn. CADaVISION: A simulation framework for machine vision prototyping. In *Proceedings of the 2nd International Conference on Computer Modelling and Simulation CSSim 2011*, pages 59–67, 2011.

[Jak10] Wenzel Jakob. Mitsuba renderer, 2010. http://www.mitsuba-renderer.org.

[MBI+16] Mahsa Mohammadikaji, Stephan Bergmann, Stephan Irgenfried, Jürgen Beyerer, Carsten Dachsbacher, and Heinz Wörn. A framework for uncertainty propagation in 3d shape measurement using laser triangulation. In *2016 IEEE International Instrumentation and Measurement Technology Conference Proceedings*, pages 1–6, 2016.

[MGLB16] Johannes Meyer, Robin Gruna, Thomas Längle, and Jürgen Beyerer. Simulation of an inverse schlieren image acquisition system for inspecting transparent objects. *Electronic Imaging*, 2016(19):1–9, 2016.

[MLB16a] Johannes Meyer, Thomas Längle, and Jürgen Beyerer. About acquiring and processing light transport matrices for transparent object inspection. *tm-Technisches Messen*, 2016.

[MLB16b] Johannes Meyer, Thomas Längle, and Jürgen Beyerer. About the acquisition and processing of ray deflection histograms for transparent object inspection. In *Irish Machine Vision and Image Processing Conference*, pages 9–16. Irish Pattern Recognition and Classification Society, 2016.

[MLB16c] Johannes Meyer, Thomas Längle, and Jürgen Beyerer. Acquisition and processing of light transport matrices for automated transparent object inspection. In *Forum Bildverarbeitung 2016*, pages 75–86, 2016.

[NZL16] Thomas Nürnberg, Christian Zimmermann, and Fernando Puente León. Simulationsgestützte Optimierung einer Computational-Kamera zur dichten Tiefenschätzung. *tm - Technisches Messen*, 83(9), January 2016.

[RTG98] Yossi Rubner, Carlo Tomasi, and Leonidas Guibas. A metric for distributions with applications to image databases. In *Computer Vision, 1998. Sixth International Conference on*, pages 59–66. IEEE, 1998.