J Sign Process Syst manuscript No.
(will be inserted by the editor)

Image Blending in a High Frame Rate FPGA-based Multi-Camera System

Vladan Popovic · Kerem Seyid · Abdulkadir Akin · Ömer Cogal · Hossein Afshari · Alexandre Schmid · Yusuf Leblebici

Received: date / Accepted: date

Abstract Panoptic is a custom spherical light field camera used as a polydioptric system where imagers are distributed over a hemispherical surface, each having its own vision of the surroundings and a distinct focal plane. The spherical light field camera records light information from any direction around its center. This paper revises previously developed Nearest Neighbor and Linear blending techniques. Novel Gaussian blending and Restricted Gaussian blending techniques for vision reconstruction of a virtual observer located inside the spherical geometry are presented. These new blending techniques improve the output quality of the reconstructed image with respect to the ordinary stitching techniques and simpler image blending algorithms. A comparison of the developed blending algorithms is also given in this paper. A hardware architecture based on Field Programmable Gate Arrays (FPGA) enabling the real-time implementation of the blending algorithms is presented, along with the imaging results and resource utilization comparison. A recorded omnidirectional video is attached as a supplementary material.

Keywords FPGA · Computational photography · Image blending · Omnidirectional imaging · Camera systems · Real-time systems

This research has been partly conducted with the support of the Swiss NSF under grant number 200021-125651 and Science and Technology Division of the Swiss Federal Competence Center (armasuisse).

Vladan Popovic · Kerem Seyid · Abdulkadir Akin · Ömer Cogal · Hossein Afshari · Alexandre Schmid · Yusuf Leblebici
Microelectronic System Laboratory, Ecole Polytechnique Fédérale de Lausanne EPFL, Station 11, 1015 Lausanne, Switzerland

E-mail: vladan.popovic@epfl.ch

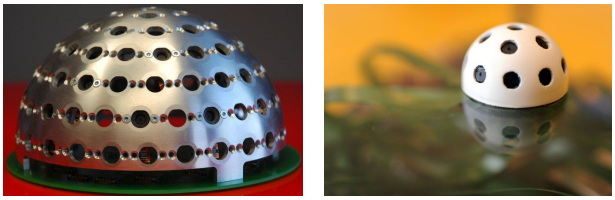
1 Introduction

A trend in constructing high-end computing systems consists of parallelizing large numbers of processing units. A similar trend is observed in digital photography, where multiple images of a scene are used to enhance performance of the capture process. The most common applications relate to increasing image resolution [1] and obtaining high dynamic range images [2, 3]. Virtualized reality and view interpolation for creating the illusion of a three-dimensional scene is another use of multi-view systems [4].

Early systems for capturing multiple views were based on a translating [5] or rotating [6–8] high-resolution camera for capturing and later rendering the scene. The advantage of a rotating camera is in its capability to acquire a high-resolution omnidirectional image, however at the cost of a long acquisition time. Therefore, it is difficult to use such systems to acquire a dynamic scene or a high frame rate video. Another disadvantage of these concepts is the limited vertical field-of-view, due to rotation around a single center. These concepts were later extended to a dynamic scene by using a linear array of still cameras [9].

An alternate approach to omnidirectional acquisition is a catadioptric system, which consists of a convex mirror placed above a single camera sensor [10]. Catadioptric systems have the advantage of real-time and high frame rate video acquisition. However, such systems are limited to the resolution of the sensor. Furthermore, their overall field-of-view (FOV) is limited, since it is restricted to the area below the sensor.

For capturing large data sets, researchers focused on arrays of video cameras. In addition to the synchronization of the cameras, very large data rates present new challenges for the implementation of these systems. The first camera array systems were built only for recording and later offline processing on Personal Computers (PC) [4]. Other such sys-



(a) (b)

Fig. 1 Side view of the fabricated Panoptic cameras with a hemispherical diameter of (a) $2r_{\odot} = 129$mm with 104 camera positions and (b) $2r_{\odot} = 30$mm with 15 camera positions.

tems [11–13] were built with real-time processing capability for low resolution and low frame rates. Another commercial omnidirectional camera was developed by Pointgrey [14]. This camera provides real-time capability, however using a limited number of sensors.

A general-purpose camera array system was built at Stanford University [15] with limited local processing at the camera level. This system was developed for recording large amounts of data and its intensive offline processing, and not for real-time operations. Recently, a camera system able to acquire an image frame with more than 1 gigapixel resolution was developed [16]. The camera uses a very complex lens system comprising of a parallel array of microcameras to acquire the image. Due to the extremely high resolution of the image, it suffers from a very low frame rate, even at low output resolution. A similar system using multiple sensors and a single ball lens is presented in [17]. This design also lacks the ability to process data with high frame rates.

Most developed camera-array systems are bulky and not easily portable platforms. Their control and operation depend on multi-computer setups. In addition, image sensors on camera arrays are usually mounted on planar surfaces which prohibits them from covering the full view of their environment. Full view or panoramic imaging finds application in various areas such as autonomous navigation, robotics, telepresence, remote monitoring and object tracking. Several solutions for acquiring omnidirectional images and their application have been presented in [18].

A new approach for creating a multi-camera system distributed over a spherical surface is presented in [19, 20]. This new multi-camera system is referred to as the Panoptic camera. The Panoptic camera is an omnidirectional imager capable of recording light information from any direction around its center. It is also a polydioptric system where each CMOS camera sensor has a distinct focal plane. Fig. 1 depicts two prototypes of a custom-made Panoptic camera.

The Panoptic camera is an image-based rendering system. Similarly to other such concepts [5, 21, 22], Panoptic acquires light ray informati on and interpolates it at rendering time. There are two main advantages of Panoptic over these systems: storage requirements and computation time. The light field/lumigraph methods require eight [22], five [5] or four [21] dimensional information in order to render the image. In contrast, Panoptic requires only the light ray intensity, since the rendering is based on a small set of calibration parameters. The rendering algorithm is explained in Section 2. This small set of parameters and the efficient rendering algorithm allow real-time high frame rate video reconstruction, which the previously mentioned concepts fail to achieve.

Reconstruction of the omnidirectional view using a multi-camera system can be regarded as creation of a photo-mosaic. A major issue in creating photo-mosaics resides in the fact that the original images do not have identical brightness levels. This may be caused by diverging camera orientations in space. Thus, cameras acquire more light in some of the shots. The problem manifests itself by the appearance of a visible seam in regions where the images overlap. Adequate image blending is required to handle the pixel instensity differences.

Blending is usually realized as a post-processing operation on a PC. However, real-time blending is often required in multi-camera systems, which can be a very challenging problem. The algorithms based on a weighted average between pixels in every image, *e.g.* , “Cut and paste” algorithm [23], are possible to implement in real-time. Furthermore, they can reduce or even completely remove the visible seams. However, the drawback of a weighted average lies in a high-frequency blurring in the presence of any small image alignment error. This work focuses on the real-time implementation of these algorithms and the comparison of their results. Additionally, new algorithms are presented which resolve high-frequency blurring without the need for complex processing hardware.

The omnidirectional vision reconstruction algorithm is presented in Section 2. Discussion of several additional abilities of Panoptic are discussed in the same section. An overview of the implemented blending algorithms is discussed in detail in Section 3. Hardware implementation of the system is given in Section 4. Imaging results and comparisons are presented in Section 5.

2 Omnidirectional Vision Reconstruction Algorithm

The omnidirectional vision of a virtual observer located anywhere inside the hemisphere of the Panoptic structure can be reconstructed by combining the information collected by each camera in the light ray space domain (or light field [5]).

In this process, the omnidirectional view is estimated on a discretized spherical surface S_d of directions. The surface of this sphere is discretized into an equiangular grid with N_θ latitudes and N_ϕ longitudes samples, where each sample represents one pixel. Fig. 2(a) shows a pixelized sphere with

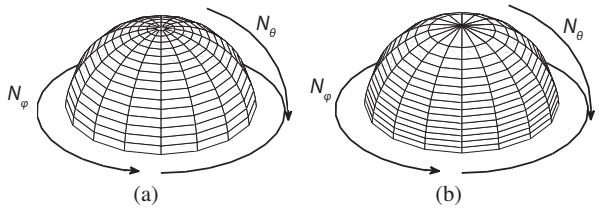


Fig. 2 Pixelized hemispherical surfaces S_d with $N_\theta = 16$ latitude pixels and $N_\phi = 16$ longitude pixels (total of 256 pixels) using a) equiangular and b) constant pixel density pixelization.

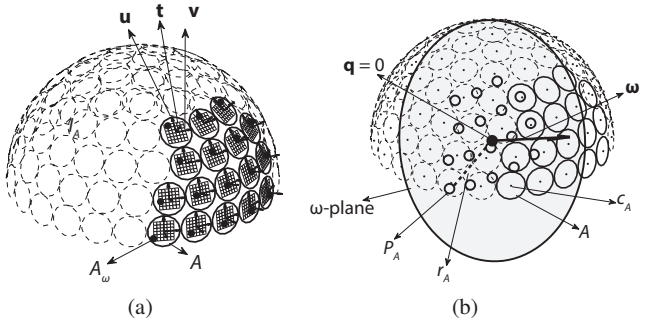


Fig. 3 (a) Cameras contributing to the direction ω with their contributing pixels in the respective image frames, (b) projections of camera centers contributing in direction ω onto planar surface normal to ω.

sixteen pixels for N_θ and N_ϕ each. A unit vector $\omega \in S_d$, represented in the spherical coordinate system $\omega = (\theta_\omega, \phi_\omega)$, is assigned to the position of each pixel. Possible pixel distributions over the sphere are discussed in Section 2.1.

The construction of the virtual omnidirectional view $\mathscr{L}(\mathbf{q}, \omega) \in \mathbb{R}$, where $\mathbf{q}$ determines the location of the observer, is performed in two steps. The first step consists of finding a pixel in each camera image frame that corresponds to the direction defined by ω. The second step consists of blending all pixel values corresponding to the same ω into one. The result is the reconstructed light ray $\mathscr{L}(\mathbf{q}, \omega)$.

To reconstruct the omnidirectional view, all the cameras having an ω in their angle-of-view are first determined. To extract the light intensity in that direction for each contributing camera, a pixel in the camera image frame has to be found. Due to the rectangular sampling grid of the cameras, the ω does not coincide with the exact pixel grid locations on the camera image frames. The pixel location is chosen using the nearest neighbor method, where the pixel closest to the desired direction is chosen as an estimate of the light ray intensity. The process is then repeated for all ω and results in the estimated values $\mathscr{L}(c_i, \omega)$, where c_i is the radial vector directing to the center position of the i^{th} contributing camera's circular face. Fig. 3(a) shows an example of the contributing cameras for a random pixel direction ω depicted in Fig. 3(b). The contributing position A_ω of the camera A, providing $\mathscr{L}(c_A, \omega)$ is also indicated in Fig. 3(a).

The second reconstruction step is performed in the space of light rays given by direction ω and passing through the camera center positions. Under the assumption of Constant Light Flux (CLF), the light intensity remains constant on the trajectory of any light ray. Following the CLF assumption, the light ray intensity for a given direction ω only varies in its respective orthographic plane. The orthographic plane is a plane normal to ω. Such plane is indicated as the "ω-plane" in Fig. 3(b), and represented as a gray-shaded circle (the boundary of the circle is drawn for clarity purposes). The light ray in direction ω recorded by each contributing camera intersects the ω-plane in points that are the projections of the cameras focal points on this plane. The projected focal points of the contributing cameras in ω direction onto the ω-plane are highlighted by hollow points in Fig. 3(b). Each projected camera point P_{c_i} on the planar surface is assigned the intensity value $\mathscr{L}(c_i, \omega)$, that is calculated in the first step.

As an example, the projected focal point of camera A onto the ω-plane (*i.e.* P_A) in Fig. 3(b) is assigned the intensity value I_A. The virtual observer point inside the hemisphere (*i.e.* $\mathbf{q}$) is also projected onto the ω-plane. The light intensity value at the projected observer point (*i.e.* $\mathscr{L}(\mathbf{q}, \omega)$) is estimated by one of the blending algorithms, taking into account all $\mathscr{L}(\mathbf{q}, \omega)$ values or only a subset of them. In the given example, each of the seventeen contributing camera positions shown with bold perimeter in Fig. 3(b) provides an intensity value which is observed into direction ω for observer position $\mathbf{q} = 0$. The observer is located in the center of the sphere and indicated by a bold dot. A single intensity value is resolved among the contributing intensities through a blending procedure on its respective ω-plane. The implemented blending algorithms are discussed in Section 3.

2.1 Sphere Pixelization Schemes

The pixel directions ω shown in Fig. 2(a) derive from an equi-angular segmentation of longitude and latitude coordinates of a unit sphere into N_ϕ and N_θ segments, respectively. This pixelization enables the rectangular presentation of the reconstructed image suitable for ordinary displays but results in a non-equal contribution of the Panoptic's cameras. The density of the pixel directions close to the poles of the sphere is higher compared to the equator of the sphere in the equi-angular pixelization scheme. Hence, the cameras positioned closer to the poles of the sphere contribute to more pixels in comparison to the other cameras of the system. The equi-angular pixelization derives mathematically from (1):

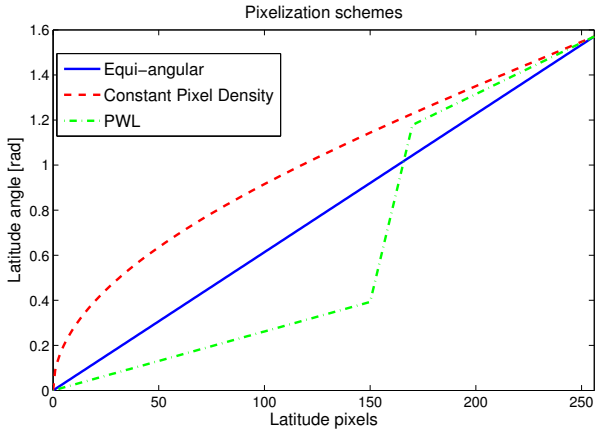


Fig. 4 Latitude angle distribution for $N_\theta = 256$ latitude pixels using three different pixelization schemes.

$$\begin{aligned} \phi_\omega(i) &= \frac{2\pi}{N_\phi} \times i, \quad 0 \le i < N_\phi \\ \theta_\omega(j) &= \frac{\pi}{2N_\theta} \times (j + \frac{1}{2}), \quad 0 \le j < N_\theta \end{aligned} \tag{1}$$

A constant density pixelization scheme resulting in an approximately even contribution of the cameras is devised for the Panoptic system. The scheme is based on enforcing a constant number of pixels per area, as expressed in (2). Compared to the equi-angular pixelization, the change is observed in latitude angles.

$$\frac{N_\phi \times j}{\int_0^{2\pi} d\phi \int_0^{\theta_\omega(j)} \sin\theta \, d\theta} = \frac{N_\phi \times N_\theta}{2\pi}, \quad 0 \le j < N_\theta \tag{2}$$

The pixelization scheme expressed in (3) is derived by solving the integral in (2). The illustration of constant pixel density pixelization is shown in Fig. 2(b).

$$\begin{aligned} \phi_\omega(i) &= \frac{2\pi}{N_\phi} \times i, \quad 0 \le i < N_\phi \\ \theta_\omega(j) &= \arccos(1 - \frac{j}{N_\theta}) + \theta_0, \quad 0 \le j < N_\theta. \end{aligned} \tag{3}$$

The offset value θ_0 is added to the latitude pixelization in (3) to avoid repetition of pixel direction for the $j = 0$ case.

Latitudal pixelization does not need to be a linear or a trigonometrical function. Moreover, it can be any function, including Piece-Wise Linear (PWL) ones. PWL functions are of special interest, since the pixel emphasis can be placed on several places on the sphere. To achieve such pixelization, the full latitudal FOV of $\pi/2$ is divided into M pieces of arbitrary FOV_i, where each piece is linearly pixelized. A

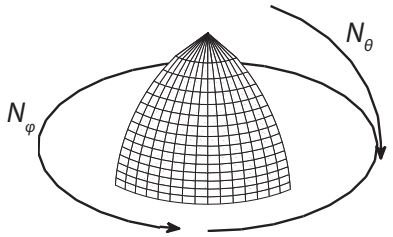


Fig. 5 Refined pixelization scheme with $N_\theta = 16$ latitude pixels and $N_\phi = 16$ longitude pixels. Longitudal FOV is reduced to a quarter of the hemisphere.

number of pixels p_i is chosen for each of the pieces, separately, based on the desired application and view specifications. The latitude angles in each segment are linearly generated with an angular slope expressed in (4):

$$\Delta\theta_i = \frac{FOV_i}{p_i}, \quad 1 < i \le M \tag{4}$$

A comparison between the presented pixelization schemes is shown in Fig. 4. An arbitrary PWL function comprising $M = 3$ pieces is taken for illustration purposes. This function results in denser pixelization near the pole and around the equator. The constant pixel density scheme provides more pixels around the equator, *i.e.* when latitude angles are higher. Finally, the equi-angular pixelization provides linearly distributed pixels around the hemisphere.

Apart from region selectivity, the PWL scheme is used for approximation of functions such as logarithms or exponentials, which is needed for easier hardware implementation. These functions can be used when more detail is required around the pole or the equator, respectively.

2.2 Grid Refinement

The presented pixelization schemes can be regarded as sampling grids of the surrounding light field. The total number of acquired pixels linearly increases with the number of cameras. Thus, the light field can become oversampled using several low-resolution cameras. Light field information is obtained at the subpixel scale as a benefit of this particular light field oversampling. Hence, Panoptic system acquires images in fine detail. In addition to the fact that the resolution of the reconstructed image can be significantly smaller than the total number of acquired pixels, this creates an excess of pixels that are not used in the reconstruction process.

Nevertheless, the acquisition of the excess pixels can be useful. If an ω direction in the reconstructed image is observed by more than one camera, *i.e.* parallax exists in each point in space, Panoptic achieves subpixel resolution.

As presented in Section 2.1, Panoptic has the ability to change pixelization schemes. Additionally, the desired FOV is also programmable. Hence, a constant output resolution with the reduced FOV results in a grid refinement effect. The example of the refined pixelization is shown in Fig. 5, where an increased pixel density can be noticed in the desired FOV.

The effect observed in the reconstructed image is similar to the effect of digital zoom. However, the subpixel data is taken from the real and previously unused data, and is not calculated in an interpolation process as in digital zooming. Hence, grid refinement provides more truthful light field rendering than digital zoom.

2.3 Vignetting Correction

Vignetting is an adverse effect observed in cameras, where the pixels located close to the image frame borders are significantly darker than the pixels located in the center. Vignetting also affects the reconstructed omnidirectional view; thus, pixel intensities in the reconstructed image alternatively vary, *i.e.* certain regions are darker and others are brighter.

Several methods are proposed in literature for modeling the vignetting effect and its correction. The chosen model for Panoptic camera is the Kang–Weiss model [24]. The Kang–Weiss model takes into account the pixel position in the camera image frame, the camera focal length and a camera constant named the vignetting factor. All pixels in each camera frame are corrected by multiplying the sampled pixel intensity with a correction factor. The corrected pixel intensity is expressed as:

$$I'(u,v) = I(u,v)(1-\alpha d)\frac{1}{(1+(d/f)^2)^2} \tag{5}$$

where α is the vignetting factor, f is the focal length, $I(u,v)$ is the original pixel intensity at coordinates (u,v) and $d = \sqrt{u^2+v^2}$.

3 Implemented Blending Techniques

The first step in omnidirectional vision construction discussed in Section 2 consists of determining contributing pixels from camera image frames and their respective intensities, $\mathscr{L}(c_i,\omega)$. The obtained values may significantly vary due to diverging camera orientations and misalignment of the pixels. Even though the vignetting correction equalizes brightness of the individual camera's image, the reconstructed image quality mostly depends on the blending algorithm.

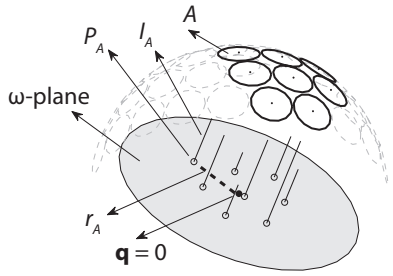


Fig. 6 Projections of camera centers onto the orthographic plane. P_A represents the projected focal point of camera A and I_A represents the set of pixel intensities.

3.1 Nearest Neighbour Blending

When applying the Nearest Neighbor (NN) technique in the second reconstruction step, the light intensity at the virtual observer point for each ω direction is set to the light intensity value of the best observing camera for that direction. The NN technique is expressed in (6) in mathematical terms:

$$\begin{aligned} j &= \text{argmin}_{i\in I}(r_i) \\ \mathscr{L}(\mathbf{q},\omega) &= \mathscr{L}(c_j,\omega) \end{aligned} \tag{6}$$

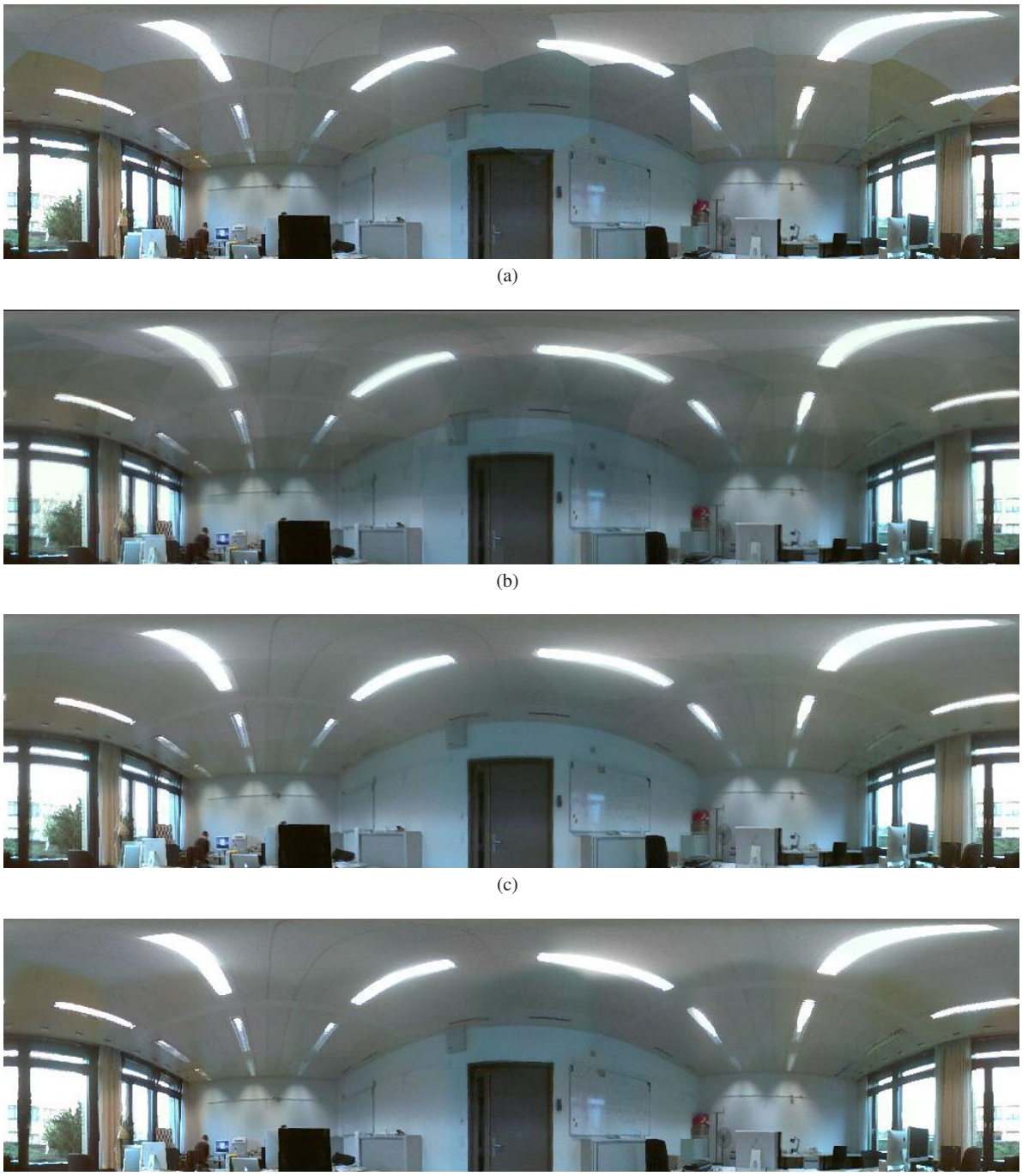
where $I = \{i | \omega \cdot \mathbf{t_i} \geq \cos(\frac{\alpha_i}{2})\}$ is the index of the subset of contributing cameras for the pixel direction ω. A pixel direction ω is assumed observable by the camera c_i if the angle between its focal vector $\mathbf{t_i}$ (see Fig. 3(a)) and the pixel direction ω is smaller than half of the minimum angle of view α_i of camera c_i. The length r_i identifies the distance between the projected focal point of camera c_i and the projected virtual observer point on the ω-plane. The camera with the smallest r distance to the virtual observer projected point on the ω-plane is considered the best observing camera. As an illustration, such distance is identified with r_A and depicted by a dashed line for the contributing camera A in Fig. 6.

Reconstructed image using the NN blending is given in Fig. 7(a).

3.2 Linear Blending

The issues resulting from different brightness levels between cameras and misalignment can be resolved to a certain extent using a linear blending algorithm.

The linear blending scheme incorporates all the cameras contributing into a selected ω direction through a linear combination [19]. This is conducted by aggregating the weighted intensities of the contributing cameras. The weight of a contributing camera is the reciprocal of the distance between its projected focal point and the projected virtual observer point on the ω-plane, *i.e.* r_A in Fig. 6. The weights are also normalized to the sum of the inverse of all the contributing cameras distances.



(a) (b) (c) (d)

Fig. 7 A computer laboratory at the Swiss Federal Institute of Technology in Lausanne (EPFL, ELD227). Panoramic construction with a pixel resolution of $N_\phi \times N_\theta = 1024 \times 256$ (a) using the nearest neighbor technique, (b) using linear blending, (c) using Gaussian blending with $\sigma_d = 100$ and (d) using Restricted Gaussian blending with $\sigma_d = 100$ and $\sigma_r = 1/30$.

The linear blending is expressed in (7) in mathematical terms.

$$\mathscr{L}(\mathbf{q},\omega) = \frac{\sum_{i\in I} w_i \cdot \mathscr{L}(c_i,\omega)}{\sum_{i\in I} w_i}$$
$$w_i = \frac{1}{r_i} \tag{7}$$

An image resulting from the linear blending algorithm is shown in Fig. 7(b).

3.3 Gaussian Blending

The NN and linear blending present several issues. An image reconstructed using the NN method shows clear boundaries between the fields of view of different cameras. Although some brightness differences are reduced by the vignetting correction, the boundaries are still visible and create an unpleasant effect to the human eye.

Linear blending solves the problem of sharp boundaries to a certain extent. Pixels in the regions where cameras' fields of view overlap are blended using a weighted average, as expressed in (7). The intensity difference is reduced, but it is still existant. Moreover, the main disadvantage lies in the appearance of blurred edges in the image due to the misalignment and linearly chosen weights.

Distributing the weights according to a Gaussian function with respect to the pixel distance from the frame center appears to be an appropriate solution to further limit the brightness difference. The new weights in the weighted average expression are:

$$w_{i,j} = \frac{1}{r_i} \cdot \mathscr{G}(d_j,\sigma_d)$$
$$\mathscr{G}(d_j,\sigma_d) = e^{-\frac{d_j^2}{2\sigma_d^2}} \tag{8}$$

where r_i is the same distance as in (7), d_j is the distance of the j^{th} pixel in the camera image frame from its center and σ_d is the variance of the Gaussian distribution function $\mathscr{G}$.

By adding the Gaussian factor to the weighted average expression, the borders between cameras are not visible any more, as shown in Fig. 7(c). Furthermore, the Gaussian blending reduces the difference in brightness in the images from different cameras and the overlapping regions are equalized with their respective surroundings. High-frequency blur is also reduced compared to the linear blending. The value of variance was empirically determined and set to $\sigma_d = 100$.

3.4 Restricted Gaussian Blending

The NN blending proves to be suitable for processing the pixels which are close to the camera center. Towards the boundaries of the camera's FOV, Gaussian blending is favorable thanks to the brightness equalization and reduction of effects originating from the camera misalignment. The Restricted Gaussian (RG) blending technique aims to restrict the Gaussian blending to the areas where the reconstructed pixels are not close to the center of a single camera's FOV. The NN blending is used in the areas close to the mentioned centers. Hence, this method benefits from the advantages of both Gaussian and NN blending.

One way of implementing this method consists of simultaneously constructing the two views and blending them for the output display. However, the hardware supporting this method is extremely resource-demanding. The method doubles the resource usage, since both NN and Gaussian blending should be operated in parallel. The implementation of this method on the current Panoptic prototypes is practically infeasible, since the required resources vastly exceed the capacity of the utilized FPGA.

A resource efficient implementation of RG blending is proposed. A new confidence factor is introduced which is related to each camera's observation of a given ω direction. For that purpose, a dot product of the ω and the focal vector $\mathbf{t}$ (see Fig. 3(a)) is taken as a reference metric.

In the blending phase of the reconstruction, a Gaussian confidence factor with respect to its $\omega \cdot \mathbf{t_i}$ is multiplied with the previously calculated $w_{i,j}$ of each camera c_i obtained from the Gaussian blending technique. By expanding (8), the RG blending weight and the Gaussian confidence factor are expressed in mathematical terms:

$$\tilde{w}_{i,j} = \frac{1}{r_i} \cdot \mathscr{G}(d_j,\sigma_d) \cdot \mathscr{C}(\omega,c_i)$$
$$\mathscr{C}(\omega,c_i) = e^{-\frac{(\omega\cdot\mathbf{t_i}-1)^2}{\sigma_r^2}} \tag{9}$$

where $\tilde{w}_{i,j}$ represents the new blending weight for j^{th} pixel in the i^{th} camera frame and $\mathscr{C}$ represents the RG confidence factor.

The RG blending favors very high values of $\omega \cdot \mathbf{t}$ for a single camera. They represent pixels which are positioned around the center of the camera frame. These pixels are considered to be more reliable than the ones located on the borders of the frame. The majority of ω have one dominant camera, *i.e.* these pixels will be around the frame center of only one camera. Thus, the RG blending should neutralize the effects of all other cameras by assigning them a very low confidence factor and keeping only the dominant camera, similar to NN blending. In cases when an ω has more than one high value of $\omega \cdot \mathbf{t}$, the confidence factors allow blend-

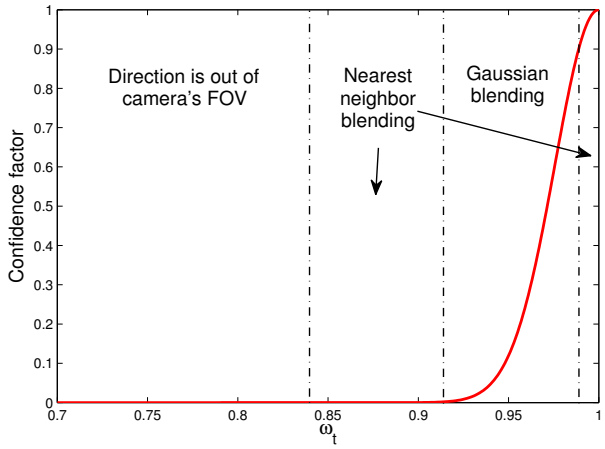


Fig. 8 Confidence factor based on $\omega_t = \omega \cdot \mathbf{t}$. Gaussian blending is applied only in the region where the confidence factor is lower than 0.9.

ing using weighted average with more than one contributing camera, resembling the Gaussian blending.

The proposed RG blending implementation does not visually differ from the approach consisting of creating two views. Furthermore, the standard deviation of the confidence factor can be manually adapted to obtain the best possible image quality.

An example of the RG blending is shown in Fig. 7(d), using an empirically determined σ_r set to $\frac{1}{30}$. The curve used for the confidence factor determination is shown in Fig. 8. The regions drawn over the curve depict the restrictions imposed on the Gaussian blending to obtain the RG blending. NN blending is applied in regions where the confidence factor is higher than 0.9, or almost 0, while Gaussian blending is applied in the transition region. This division reduces the influence of low-confidence over high-confidence pixels. Thus, the reconstructed image is sharp in areas close to a single cameras' center, while the camera overlapping regions located on the periphery are blended using a Gaussian weight distribution.

4 Hardware Implementation

A custom FPGA board has been designed using a XILINX Virtex5 XC5VLX50-1FF1153C FPGA as a core processing unit in order to capture and process the video streams produced by the cameras in real-time. This board interfaces with twenty PIXELPLUS PO4010N single-chip Common Intermediate Format (CIF, 352×288) cameras with 66° minimum angle of view. They provide output data in 16-bit RGB format with selectable frame rate. The cameras of the Panoptic system have been calibrated for their true geometrical position in the world space, and lens distortion parameters are obtained. The extraction of their intrinsic parameters is also done *a priori* [25]. Even though the camera calibration is precise within certain error bounds, the spherical arrangement of the cameras, *i.e.* diverging camera directions, emphasize misalignment problems. This misalignment can be as large as a few pixels; hence, appropriate blending algorithms are still needed.

The number of cameras connected to a single board is limited by the user I/O pin availability of the chosen FPGA chip. To support higher number of camera interfaces, multiple identical boards of the same kind are stacked. For scalability and extension purposes, the designed board also contains high-speed Low-Voltage Differential Signaling (LVDS) serial links and extension connectors. The board is also equipped with a Universal Serial Bus (USB) 2.0 device chipset for external access and high-speed data transfer. The FPGA board contains two Zero Bus Turn around (ZBT) Static Random Access Memories (SRAM) with 36 Mb capacity and an operating bandwidth of 167 MHz, for each. The maximum achievable throughput using this SRAM is approximately 3 Gbps.

4.1 Top-level FPGA Architecture

The architecture of the FPGA is depicted in Fig. 9(a). The FPGA design consists of five major blocks. The arrow lines depicted in Fig. 9(a) show the flow of image data inside the FPGA. Image data streaming from the cameras enters the FPGA via the Camera input channel block. A time-multiplexing mechanism is implemented to store the incoming frame data from all the camera modules into one of the single data port SRAMs. Hence, the Data transmit multiplexer block time-multiplexes the data received by the Camera input channel block and transfers it to the Memory controller block for storage in one of the SRAMs. The SRAMs are partitioned into twenty equal segments, one for each camera. The Memory controller block interfaces with two external SRAMs available on the board. The Memory controller block provides access for storing/retrieving the incoming/previous twenty frames in/from the SRAMs. The SRAMs swap their roles (*i.e.* one is used for writing and one for reading) with the arrival of each new image frame from the cameras. The Image processing and application unit block is in charge of signal processing and basic functionalities such as single video channel streaming, all channels image capture and omnidirectional view reconstruction. Thist block accesses the SRAMs via the Memory controller block and transfers the processed image data to the Data link and control unit block. The Data link and control unit block provides transmission capability over the external interfaces available on the board such as high-speed LVDS serial links or the USB 2.0 link. The Cameras control block is in charge of programming and synchronizing the cameras connected to the FPGA board.

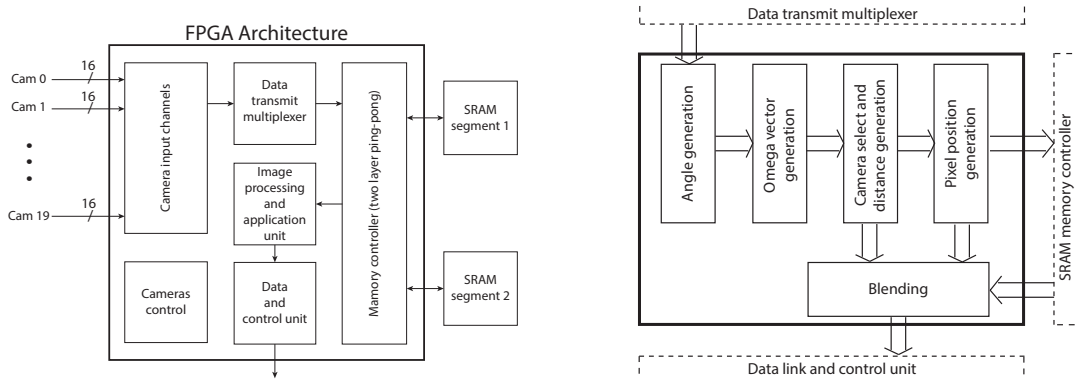


Fig. 9 (a) System-level architecture, (b) block diagram of the light field reconstruction unit inside the Image processing and application block.

4.2 Image Reconstruction Hardware

The reconstruction algorithm is implemented inside the Image processing and application unit. The block diagram is shown in Fig. 9(b). This image processing entity comprises five modules, which are thoroughly discussed in the following sections.

4.2.1 Angle and Omega Vector Generation

The Angle generation module generates the spherical coordinates, *i.e.* $(\theta_\omega, \phi_\omega)$, of the ω directions which are of interest for the reconstruction. It has the ability of generating angles for both equi-angular and constant pixel density pixelization schemes from (1) and (3). The span and resolution of the output view is selectable within this module. It is possible to reconstruct a smaller portion of the light field with an increased resolution, due to the initial oversampling of the light field (*i.e.* the cameras record more samples than the reconstructed image has), as explained in Section 2.2. Hence, a more detailed image with a limited field of view can be reconstructed while keeping the same frame rate. Furthermore, higher resolutions can be achieved by trading off the frame rate. Since the coordinate angles are represented by 13 bits, the maximum reconstruction resolution for a hemisphere is 32M pixels at 0.2 frames per second. The 13-bit representation leaves enough margin for truthful representation, considering the used CIF imagers and the total amount of the acquired pixels.

The Omega vector generation module calculates the radial unit vector pertaining to the spherical position $(\theta_\omega, \phi_\omega)$ received from the Angle generation module. The vectors are generated according to the following equation:

$$\omega = \sin\theta_\omega \cos\phi_\omega \,\mathbf{x} + \sin\theta_\omega \sin\phi_\omega \,\mathbf{y} + \cos\theta_\omega \,\mathbf{z}. \tag{10}$$

Detailed hardware implementation of the Angle generation and Omega vector generation modules can be found in [26].

4.2.2 Camera Selection and Distance Generation

The Camera select and distance generation module identifies which cameras contribute (*i.e.* observe) to the construction of the pixel in ω direction. Concurrently, this module computes the distance between the focal point projection and the virtual observer projection on the ω-plane, for each contributing camera c_i in direction $\omega_\mathbf{j}$, as expressed in (11):

$$r_{i,j} = \left|(\mathbf{q}-\mathbf{t_i}) - ((\mathbf{q}-\mathbf{t_i})\cdot\omega_\mathbf{j})\,\omega_\mathbf{j}\right| \tag{11}$$

When processing the NN blending, the module searches for the minimum distance through all the calculated distances for one ω. The index of the closest camera is provided at the output. When processing any other blending methods which are based on a weighted average, the module provides all contributing cameras' indices and their distances $r_{i,j}$ from the virtual observer. The pseudo-code of the module's operation is provided as Algorithm 1.

Algorithm 1 Camera Select and Distance Generation

```
r_min ← 1
for all cameras do
    ω_t ← ω · t
    r ← (q − t) − ((q − t) · ω) ω
    if (ω_t > cos(α/2)) then
        if interpolation == nearest_neighbor then
            if (|r| < r_min) then
                r_min ← |r|
                STORE camera_index
            end if
        else
            r ← |r|
            index ← camera_index
        end if
    end if
end for
if interpolation == nearest_neighbour then
    r ← r_min
    index ← camera_index
end if
```

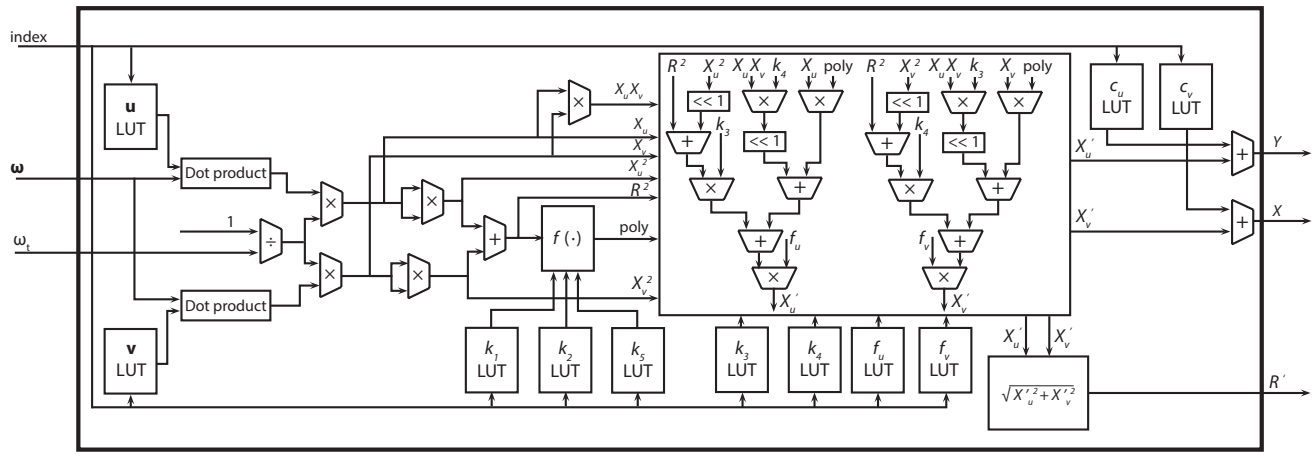


Fig. 10 Block diagram of the Pixel position generation module.

4.2.3 Pixel Position Generation

The Pixel position generation module calculates the true pixel position in the image frame of the cameras selected in the Camera selection block. This goal is achieved using a pinhole camera model [27] to obtain the two-dimensional position (X_u, X_v) on the camera image plane (which is identified by the vectors **u** and **v**) expressed as:

$$(X_u, X_v) = -(\omega \cdot \mathbf{u} \times \frac{f_u}{\omega \cdot \mathbf{t}}, \omega \cdot \mathbf{v} \times \frac{f_v}{\omega \cdot \mathbf{t}}), \tag{12}$$

where f_u and f_v represent the focal lengths in **u** and **v** coordinates of the camera image plane.

In reality, the mapping of a 3-D scene onto an observed 2-D plane of a camera image frame is a complex problem, which is only coarsely represented by (12). The intrinsic parameters of the camera, categorized in two classes, characterize the mapping between a 3-D scene and the observed 2-D plane. The first class is the linear homography, defined by a 3×4 camera matrix mapping of 3-D points coordinates into 2-D pixel coordinates [27]. The second class models the non-linear effects such as lens distortion. These parameters are estimated through a calibration process [25] and stored for each camera in a Look-Up Table (LUT). A detailed block diagram of this module is shown in Fig. 10. The pipeline registers are omitted for clarity purpose.

The Pixel position generation module interfaces with the SRAM memory controller to retrieve the pixel value of the contributing cameras upon calculation of the true pixel position. The camera index originating from the Camera select module is used to access the correct segment of the SRAM, *i.e.* the segment where the image frame of the selected camera is stored. The true pixel position is used to access the target address within each segment inside the SRAM. The Pixel position module also calculates the distance of the selected pixel in the image frame from the image center. This distance is represented as R' in Fig. 10 and it is further used for Gaussian blending and the vignetting correction.

4.2.4 Blending

The Blending module receives the pixel light intensity values from all contributing cameras along with the two distances $r_{i,j}$ and R', detailed in Section 4.2.2 and 4.2.3. The distances are used to calculate the contributing weights using one of the blending expressions from (7), (8), (9). The weights are later used in the blending process discussed in Section 3. As a result, the block estimates a single light intensity value for each ω direction.

A straightforward implementation of the blending algorithms is very resource demanding. Multipliers and dividers would be required for each color channel separately. It is benefitial to share resources which are common for all channels, *e.g.* implementation of the denominator in (7). For the implementation purpose, (7) is expressed differently in (13):

$$\begin{aligned} \mathscr{L}(\mathbf{q}, \omega) &= \sum_{i \in I} a_i \cdot \mathscr{L}(c_i, \omega) \\ a_i &= \frac{w_i}{\sum\limits_{k \in I} w_k} \end{aligned} \tag{13}$$

The pseudo-code of the blending module's operation is shown as Algorithm 2. The following notation is used in the algorithm: I_{RGB} represents color intensities of the contributing pixels; $w_{acc} = \sum\limits_{k \in I} w_k$; $G(R'_i)$ is the Gaussian blending factor; $C(a,b)$ is the Restricted Gaussian confidence factor.

4.3 Scalability

Each FPGA board can interface with 20 cameras. To support a higher number of cameras and increase the throughput

Algorithm 2 Blending

if interpolation == nearest_neighbor **then**
  $\mathscr{L}_{RGB} \leftarrow I_{RGB}$
else
  $w_{acc} \leftarrow \sum_{i \in I} \frac{1}{r_i}$
  for all $i \in I$ **do**
    $a_i \leftarrow \frac{1}{r_i} \cdot \frac{1}{w_{acc}}$
    if interpolation == gaussian **then**
      $a_i \leftarrow a_i \cdot G(R'_i)$
    end if
    if interpolation == restricted **then**
      $a_i \leftarrow a_i \cdot G(R'_i) \cdot C(a,b)$
    end if
  end for
  for all color_channels **do**
    $\mathscr{L}_{RGB} \leftarrow \sum_{i \in I} I_{RGB} \cdot a_i$
  end for
end if

of the Panoptic camera, multiple FPGA boards must be incorporated. Hence, the omnidirectional view reconstruction workload is distributed and the algorithm operates in parallel on all FPGA boards. Thus, a central FPGA is required to receive the output data from all FPGA boards, apply the final blending process and transfer the result to a PC for display.

A scalable FPGA-based system is devised, using the designed FPGA board, to support the application development of the Panoptic camera. The devised system consists of four layers: 1) image sensors, 2) FPGA boards handling local image processing, 3) one central FPGA board for control, external access and last stage image processing, 4) a PC in charge of the applicative layer consisting of displaying the operation results transmitted from the central FPGA board. The designed central board supports up to five layer-2 FPGAs. Fig. 11 depicts the devised architecture for a typical Panoptic system.

The layer-2 FPGA boards implement the architecture presented in Section 4.1. The outputs of these boards carry the value related to locally blended pixel values and their corresponding weight. These two 16-bit values are streamed to the central unit for the final blending step via an LVDS link. The LVDS link is implemented in the Data and control unit shown in Fig. 9(a). The 16-bit pixel value and its weight are split into the most significant byte (MSByte) and the least significant byte (LSByte). Xilinx embedded serializer blocks are used to serialize the bytes and transfer them to the central FPGA. The byte order is as follows: 1) LSByte of the pixel value, 2) MSByte of the pixel value, 3) LSByte of the blending weight, 4) MSByte of the blending weight.

The full-resolution frame is transferred via LVDS, irrespective of the FOV of the cameras connected to the observed layer-2 FPGA. In practice, this means that the pixels in the reconstructed image which are not observed by the connected cameras are also transferred. In such cases, both the pixel value and the weight are set to zero, *i.e.* the pixel is considered purely black and as such the least influential in the final blending operation.

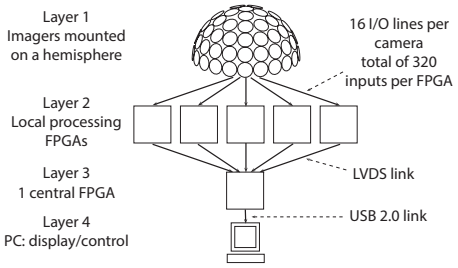


Fig. 11 Architecture of the multi-layer Panoptic system.

Furthermore, the LVDS link is used to transfer commands issued from the central to the slave FPGAs. The implemented commands are "start/stop video stream", "capture a single snapshot" and "reset the whole system".

The central FPGA architecture consists of two main parts: input buffers that store data from the slave FPGAs and the image processing unit. The input buffers deserialize the incoming data and recover pixel values with its respective weights. All slave boards are synchronized a with maximum of one clock cycle latency. Hence, short input FIFOs are used as input buffers and memory storage is avoided. The processing unit of the central FPGA is significantly simpler than in the slave FPGA, as it only contains the Blending module. It calculates the final results based on the pixel values and the weights calculated in the slave FPGAs. The final values are sent to the PC for display, via a USB link.

5 Results and Discussion

5.1 System Performance

The Panoptic system with thirty embedded cameras presented in [26] is used for real-time image extraction and evaluation. The thirty-camera system contains two FPGA boards for camera interfacing and one central FPGA.

The operating frequency of the design implemented in the FPGA is 133 MHz, which allows the system to output 25 frames per second video stream of 1024×256 pixels resolution. Hence, the output video streaming rate of each FPGA board is 6.6M pixels per second. The total latency of the system is 132 clock cycles, which is less than 1 μs, using 133 MHz frequency. The power consumption of each FPGA board in operation is only 5W.

The discussed blending methods were separately implemented on the FPGA in order to compare the resource utilization of a single FPGA board. The summary is presented in Table 1. Gaussian and RG blending infer additional LUTs and multipliers compared to NN and Linear blending. This is observed through the increase of the used BlockRAMs

Table 1 FPGA resource usage comparison

Blending	Nearest Neighbor	Linear	Gaussian	Restricted Gaussian		
Resource	Used				Available	Largest Utilization [%]
Slices	4070	4653	4607	4816	7200	67
Slice Registers	9351	10069	10127	10196	28800	35
BlockRAMs/FIFOs	17	17	21	22	48	46
DSPs	37	47	48	48	48	100

Table 2 Comparison with related camera systems in terms of system performance

	Panoptic	Panoptic-100	Ladybug2	Ladybug3	Cockpit [13]	Aware2 [16]	Yang [12]
Number of Cameras	30	100	6	6	8	94	64
Camera Resolution	352×288	352×288	1024×768	1600×1200	320×240	4384×3288	320×240
System Throughput [Mbit/s]	1216	4055	566	600	40	526	66
Output Resolution	1024×256	1024×256	–	–	640×480	–	320×240
Output Frame Rate [fps]	25	25	15	6.5	20	0.05	18
FOV	$360^\circ \times 90^\circ$	$360^\circ \times 90^\circ$	$360^\circ \times 150^\circ$	$360^\circ \times 150^\circ$	$150^\circ \times 110^\circ$	$120^\circ \times 50^\circ$	–
Power Consumption [W]	5	< 10	11.2	7.2	–	430	–

and logic slices. However, the increase of resource usage compared to the linear blending and NN is very small and is not an influential factor in the overall utilization.

The Panoptic system is compared to the omnidirectional camera systems and the summary is given in Table 2. The first column corresponds to the implemented Panoptic camera with 30 imagers. The second column is an estimate of the system performance if all five slave boards are used to implement a 100 cameras system. The estimated results give a notion of the full capabilities of the Panoptic system. Ladybug2 and Ladybug3 cameras [14] are included in the comparison, as an example of off-the-shelf systems with similar goal. The remaining three cameras are scientific cameras and the data shown in the table is taken from the original publications. The fields marked with "–" represent data which is either not available (power consumption and FOV) or scene-dependent (output resolution).

The Panoptic system and architecture enable the highest data throughput or "system-level data throughput". Even though the presented prototype uses relatively low-resolution low-cost cameras, the system architecture is able to process huge amount of data in real-time. Other systems, such as Aware2, have higher number of acquired pixels, but they are unable to process it and have to lower the output frame rate. Furthermore, Panoptic has the lowest power consumption within the compared systems, thanks to its customized FPGA architecture.

5.2 Image Quality Discussion

Four captured snapshots of the same scene from the real-time output (*i.e.* 25 frame per second) of the Panoptic device with thirty embedded cameras are shown in Fig. 7. The horizontal and vertical directions in the shown panoramic constructions correspond to ϕ and θ spherical coordinates, respectively. Constant pixel density pixelization is used in the reconstruction. During the image acquisition session, cameras were set to automatic mode, *i.e.* exposure settings, gamma correction and white balancing were provided by the sensor. As the shooting took place indoors, settings were different, thus a difference in color tones is observable in Fig. 7, for several cameras in the setup. Fig. 7(a) corresponds to the panoramic scene constructed for a virtual observer located at the center of the sphere using the NN technique. No automatic gain compensation or radiometric calibration has been used for the cameras. Hence, the boundaries between the cameras are apparent and high intensity changes are visible in Fig. 7(a). The linear blending technique improves the color intensity variations as observed in Fig. 7(b) and provides a scene with less sharp color transitions. However, it also results in a high-frequency blur, also known as the ghosting effect, altering the objects that are close to the Panoptic system. The ghosting manifests itself as the duplication of the object's edges. Fig. 7(c) shows the omnidirectional view reconstruction using the Gaussian blending. The color transitions in the overlapping regions are significantly reduced as a benefit of the applied Gaussian factor. The Gaussian blending is not a filtering operation, thus it does not reduce the image sharpness as it only affects the inter-camera brightness differences. Fig. 7(d) shows the result of the Restricted Gaussian blending. The edges in the image are sharper compared to the linear and Gaussian blending, as the ghosting effect is almost completely neutralized. This is especially noticeable in the areas around the desk lamp and some of the ceiling lights. However, the background bright-

Fig. 12 Detailed image parts obtained using Gaussian blending with grid refinement: a lamp magnified 8x, the books magnified 32x, a desk magnified 8x.

ness level is less equalized compared to the Gaussian blending due to differently selected weights. Hence, RG blending requires an additional pre-processing step such as Gain Compensation [28] to produce high-quality images.

Furthermore, Fig. 12 depicts the ability of Panoptic to refine the pixelization grid in a selected portion of space. This results in increased detail, while keeping the same output image resolution. In Fig. 12 a lamp, a desk and books are shown in increased resolution. The quality of the magnified parts is proportional to the number of observing cameras.

A 30 seconds long video is provided as a supplementary material showing a video stream record from Panoptic system with 15 cameras shown in Fig. 1(b). The video shows the entrance hall of ELA building in EPFL campus.

6 Conclusion

The abilities of the Panoptic camera system, *e.g.* change of pixelization and use of light field oversampling for zooming, are explained. Several blending techniques enabling the omnidirectional view reconstruction of the Panoptic camera are discussed. The introduced Gaussian blending algorithm decreases the high light intensity variations in the reconstructed image. Its enhancement version, the Restricted Gaussian blending, bounds the region where Gaussian blending is applied only to the parts where one camera is not dominant over the others. In the remaining areas, NN blending is used. However, the ghosting effect for the close objects is still noticeable in a few regions. To further improve the output of the Panoptic camera, the real-time implementation of the multi-band blending technique [28],[29] is considered.

The architecture of an FPGA based system supporting the real-time deployment of the reconstruction algorithm is presented in detail. Snapshots of the real-time output of the Panoptic system are presented, along with a recorded example video. Furthermore, the ability to display image parts in finer detail is also presented.

Future work related to the Panoptic device focuses on Gigapixel resolution real-time light field reconstruction, High-dynamic-range video, real-time 3-D cinematography and Application-Specific Integrated Circuit (ASIC) design of the current system.

Acknowledgements The authors thank S. Hauser, P. Bruehlmeier and E. Erdede for their contribution. The authors gratefully acknowledge the support of XILINX, Inc., through the XILINX University Program.

References

1. Szeliski, R. (1994). Image Mosaicing for Tele-Reality Applications. In *Proceedings of the Second IEEE Workshop on Applications of Computer Vision*, (pp. 44 –53).
2. Mann, S., & Picard, R.W. (1995). On Being 'Undigital' with Digital Cameras: Extending Dynamic Range by Combining Differently Exposed Pictures. In *Proceedings of IS&T*, (pp. 442–448).
3. Debevec, P.E., & Malik, J. (1997). Recovering High Dynamic Range Radiance Maps from Photographs. In *Proceedings of the 24th Conference on Computer Graphics and Interactive Techniques*, (pp. 369–378).
4. Rander, P., Narayanan, P.J., & Kanade, T. (1997). Virtualized Reality: Constructing Time-Varying Virtual Worlds From Real World Events. In *Proceedings of IEEE Visualization '97*, (pp. 277–284).
5. Levoy, M., & Hanrahan, P. (1996). Light Field Rendering. In *Proceedings of the 23rd annual Conference on Computer Graphics and Interactive Techniques*, (pp. 31–42). doi:10.1145/237170.237199.
6. Schechner, Y., & Nayar, S. (2001). Generalized Mosaicing. In *Proceedings of IEEE International Conference on Computer Vision*, (pp. 17–24).
7. Sarachik, K.B. (1989). Characterising an Indoor Environment with a Mobile Robot and Uncalibrated Stereo. In *Proceedings of IEEE International Conference on Robotics and Automation*, (pp. 984–989). doi:10.1109/ROBOT.1989.100109.
8. Shum, H.Y., & He, L.W. (1999). Rendering with Concentric Mosaics. In *Proceedings of the 26th annual conference on Computer graphics and interactive tech-*

niques, SIGGRAPH '99, (pp. 299–306). New York, NY, USA: ACM Press/Addison-Wesley Publishing Co. doi:10.1145/311535.311573.

9. Taylor, D. (1996). Virtual Camera Movement: The Way of the Future? *American Cinematographer*, *77*(8), 93–100.
10. Nayar, S.K., & Peri, V. (1999). Folded Catadioptric Cameras. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, (pp. 217–223).
11. Zhang, C., & Chen, T. (2004). A Self-Reconfigurable Camera Array. In *Eurographics Symposium on Rendering*, (pp. 243–254).
12. Yang, J.C., Everett, M., Buehler, C., & McMillan, L. (2002). A Real-Time Distributed Light Field Camera. In *Proceedings of the 13th Eurographics Workshop on Rendering*, (pp. 77–86).
13. Tang, W.K., Wong, T.T., & Heng, P.A. (2005). A System for Real-Time Panorama Generation and Display in Tele-immersive Applications. *IEEE Transactions on Multimedia*, *7*(2), 280–292.
14. Ladybug, Pointgrey. URL `http://www.ptgrey.com/products/spherical.asp`. Accessed on Sep. 16, 2013.
15. Wilburn, B., Joshi, N., Vaish, V., & et al. (2005). High Performance Imaging Using Large Camera Arrays. *ACM Trans. Graph.*, *24*, 765–776. doi:10.1145/1073204.1073259.
16. Brady, D.J., Gehm, M.E., Stack, R.A., Marks, D.L., Kittle, D.S., Golish, D.R., Vera, E.M., & Feller, S.D. (2012). Multiscale Gigapixel Photography. *Nature*, *486*(7403), 386–389.
17. Cossairt, O.S., Miau, D., & Nayar, S.K. (2011). Gigapixel Computational Imaging. In *Proceedings of IEEE International Conference on Computational Photography*, (pp. 1–8).
18. Yagi, Y. (1999). Omni Directional Sensing and Its Applications. *IEICE Transactions on Information and Systems*, *E82-D*(3), 568–579.
19. Afshari, H., Akin, A., Popovic, V., Schmid, A., & Leblebici, Y. (2012). Real-Time FPGA Implementation of Linear Blending Vision Reconstruction Algorithm Using a Spherical Light Field Camera. In *IEEE Workshop on Signal Processing Systems*, (pp. 49–54). doi:10.1109/SiPS.2012.49.
20. Popovic, V., Afshari, H., Schmid, A., & Leblebici, Y. (2013). Real-time Implementation of Gaussian Image Blending in a Spherical Light Field Camera. In *Proceeding of IEEE International Conference on Industrial Technology*, (pp. 1173–1178).
21. Gortler, S.J., Grzeszczuk, R., Szeliski, R., & Cohen, M.F. (1996). The Lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, (pp. 43–54). New York, NY, USA: ACM. doi:10.1145/237170.237200.
22. Szeliski, R., & Shum, H.Y. (1997). Creating Full View Panoramic Image Mosaics and Environment Maps. In *Proceedings of the Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, (pp. 251–258). New York, NY, USA: ACM. doi:http://dx.doi.org/10.1145/258734.258861.
23. Peleg, S., & Herman, J. (1997). Panoramic mosaics by manifold projection. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, (pp. 338 –343). doi:10.1109/CVPR.1997.609346.
24. Kang, S.B., & Weiss, R.S. (2000). Can We Calibrate a Camera Using an Image of a Flat, Textureless Lambertian Surface? In *Proceedings of the 6th European Conference on Computer Vision - Part II*, (pp. 640–653).
25. Bouguet, J. (2010). Camera Calibration Toolbox for Matlab. URL `http://www.vision.caltech.edu/bouguetj/calib_doc`. Accessed on Dec. 7, 2011.
26. Afshari, H., Jacques, L., Bagnato, L., & et al. (2013). The PANOPTIC Camera: A Plenoptic Sensor with Real-Time Omnidirectional Capability. *Journal of Signal Processing Systems*, *70*(3), 305–328.
27. Hartley, R.I., & Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition.
28. Brown, M., & Lowe, D. (2007). Automatic Panoramic Image Stitching Using Invariant Features. *International Journal of Computer Vision*, *74*(1), 59–73.
29. Burt, P.J., & Adelson, E.H. (1983). A Multiresolution Spline with Application to Image Mosaics. *ACM Trans. Graph.*, *2*(4), 217–236. doi:10.1145/245.247.