Journal of Mobile Multimedia, Vol. 12, No. 1&2 (2016) 125–137

STRUCTURED-LIGHT-BASED DEPTH RECONSTRUCTION USING LOW-LIGHT PICO PROJECTOR

THOMAS RITTLER
Institute of Software Technology and Interactive Systems, Vienna University of Technology, Favoritenstrasse 9-11
1040 Vienna, Austria
thomas.rittler@tuwien.ac.at

FLORIAN SEITNER
emotion3D GmbH, Gartengasse 21/3
1050 Vienna, Austria
florian.seitner@emotion3d.tv

MARGRIT GELAUTZ
Institute of Software Technology and Interactive Systems, Vienna University of Technology, Favoritenstrasse 9-11
1040 Vienna, Austria
margrit.gelautz@tuwien.ac.at

In this work we investigate an infrared structured light prototype which is intended for 3D reconstruction in resource-restricted mobile applications. We explore the constraints on working range and pattern resolution that are imposed by the low-light property of our single-shot set-up. While focusing on the most light-sensitive steps of the decoding workflow, we suggest adaptations of image rectification, pattern generation and segmentation algorithms that are tailored to the specific spatial and radiometric requirements of our system. We incorporate two disparity estimation techniques based on codeword look-up and foreground/background segmentation into our system and analyze the effects of different algorithmic components on the overall runtime of our implementation.

Keywords: Structured light, low-light projector, image rectification, pattern segmentation

1 Introduction

Depth sensing finds its use in a wide range of applications such as 3D reconstruction/scanning, mobile multimedia applications and gaming, or automotive applications. In order to compute depth information from a given scene by stereo analysis, two main techniques can be distinguished: passive (e.g., [12, 1, 3]) and active stereo vision approaches (e.g., [5, 10, 2, 4]). A passive stereo matching algorithm tries to extract the 3D scene geometry via triangulation from two images that were acquired from slightly different viewpoints by identifying corresponding points in both images. The process of correspondence finding is known as the stereo matching problem and constitutes the key challenge in the stereo processing pipeline. Passive stereo matching algorithms typically encounter problems on textureless surfaces and in areas of repetitive textures, due to ambiguities among match point candidates. Unfavorable lighting conditions such as illumination differences between the left and right stereo view or poor ambient illumination with increased image noise can further complicate the determination of point correspondences. The resulting matching inaccuracies can significantly reduce the

quality of the retrieved depth information when passive techniques are used.

Contrary to passive stereo approaches, active Structured Light (SL) methods substitute one of the cameras by an active device, for example a projector or laser. This active device emits a light pattern - named as *SL illumination* - that is reflected by the scene and captured by the camera. The additionally projected pattern makes the determination of point correspondences more independent of surface texture and illumination characteristics and thus overcomes some important limitations of passive stereo techniques. The increased robustness and accuracy of SL techniques compared to passive stereo approaches is also reflected by the usage of SL-derived depth maps as ground truth data for the evaluation of passive stereo matching techniques [13].

In this paper, we focus on a near-infrared SL prototype which is intended to be used on resource-restricted mobile devices. The set-up consists of a single camera and pico projector. Specifically, we investigate the low-light properties of the projector regarding their effects on the image rectification, pattern codification, segmentation and disparity estimation. We address more literature related to the choice of our algorithms in the following section.

The rest of this paper is organized as follows. Section 2 takes a closer look at the algorithmic aspects including the rectification process between the captured camera image and the projected pattern image, strategies for pattern codification and extraction, as well as disparity estimation. In Section 3 the working range of the proposed SL system with its low projector brightness is evaluated and runtime results are presented. We summarize our findings in the concluding Section 4.

2 Algorithms

In this section, we discuss several algorithmic aspects of the proposed SL system under consideration of the constraints imposed by the low-light property and spatial arrangement of our set-up.

2.1 Image rectification

The first step of the SL decoding workflow comprises the rectification between the camera and projector image. Due to its simplicity and well-known accuracy, the method by Zhang [16] has become one of the most widespread approaches for camera calibration. Inspired by that earlier work, Moreno and Taubin [8] proposed a method for camera/projector calibration by modeling the projector as an inverse camera. A sequence of Gray code patterns is projected onto a static planar checkerboard placed within the working volume for finding correspondences between projector pixels and 3D world points. This approach requires a physical checkerboard to be placed in the scene at various positions, which has to remain static during projecting and capturing of the Gray code pattern sequence.

In the case of low projector brightness the accurate determination of chessboard corner locations is diminished or may not be possible at all due to blurry delineations of chessboard tiles and an increased presence of image noise. Hence, since the vertical alignment between projector and camera image denotes a crucial step in order to extract depth information efficiently, we propose a computationally fast method for uncalibrated image rectification. Instead of placing the chessboard physically into the scene, a chessboard image is projected onto a planar surface.

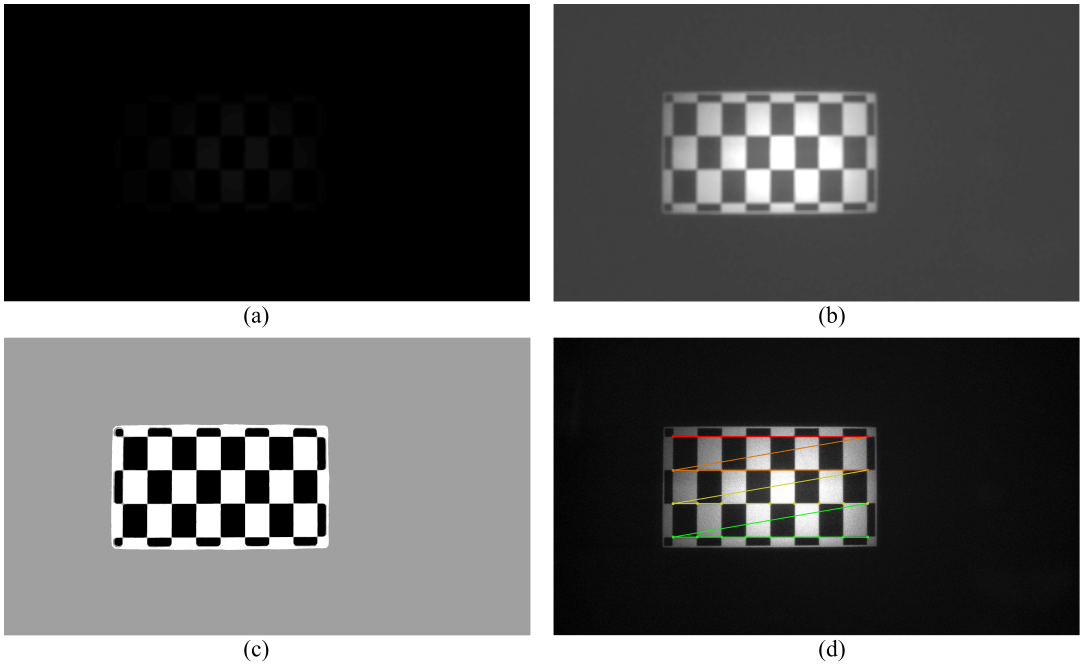

Fig. 1. Chessboard-based image rectification: (a) Initial camera image. (b) Camera image after contrast enhancement and denoising. (c) Camera image with binary chessboard pattern formed after combining the regular and inverse projected chessboard patterns. (d) Detected chessboard corners overlaid on contrast enhanced input image.

In our processing chain, we first enhance the contrast of the camera image and reduce the amount of image noise by local filtering operators (see Figure 1 b). To increase the robustness of the corner detection process, the inverse chessboard is projected additionally and both captured camera images are combined to form a binary image of the chessboard pattern as illustrated in Figure 1 c. This binary image serves as the basis to fit the initial chessboard pattern (i.e. projector image) into the camera image via cross-correlation by adapting the scale and orientation of the pattern image. Finally, the artificially created camera image is used to compute the chessboard corner correspondences. After the inner chessboard corners have been detected, a mapping between coordinates of projector and camera pixels is established based on the chessboard corner correspondences. This mapping is then extrapolated in the area of the outer chessboard tiles.

As the correspondences between chessboard corners of the projector image and the camera image are established based on the inner chessboard corner points, the size of the exterior chessboard tiles is reduced so that the set of inner corner points covers a larger part of the projector image, as can be seen in Figure 1.

2.2 Pattern codification strategy

A structured light system is based on the projection of a single pattern or a set of patterns onto a scene, which is then viewed by a single camera. The patterns are specially designed so that every pixel has its own codeword to establish a direct mapping from the codewords to

the corresponding coordinates of the pixels in the pattern. The codification strategies of such patterns can be classified into two main categories: time-multiplexing and neighborhood codification [11]. In case of time-multiplexing, a set of patterns (e.g., Gray codes) is successively projected onto the surface to be measured. Such techniques can achieve high accuracy in the measurements [13], but are not suitable in case of dynamic scenes as the bits of a codeword are multiplexed in time.

In contrast to temporal coding approaches, spatial neighborhood techniques densify the coding scheme into a single pattern image. In particular, the codeword of a specific position is extracted from its surrounding points which allows to calculate depth information for each frame individually, and thus enables the use of these approaches for dynamic scenes. However, as a spatial neighborhood is required to perform decoding for each pixel, such methods lose spatial resolution and perform poorly around depth discontinuities [14]. Moreover, the decoding stage becomes more difficult as the spatial neighborhood cannot always be identified. Nevertheless, spatial neighborhood coding is the method of choice for our set-up because we are targeting mobile applications.

Since our aim is to construct a SL system using a near infrared light source (i.e. invisible to the human eye), color encoding of the pattern image as, for example, suggested in [15] is not an option. Additionally, to minimize the computational complexity and to ease the decoding process in case of low projector brightness, the encoding of codewords based on different shapes (e.g., [6]) appears not to be a good strategy. Thus, we have chosen a binary pseudorandom array using circles as shape primitives (hereinafter also referred to as dot pattern) as the proposed SL pattern.

A widespread algorithm in the computer vision community for constructing such a pseudorandom array is given by Morano et al. [7]. The authors propose an iterative process based on a brute-force approach that allows to define the length of the alphabet, the size of the window, the dimensions of the array and the Hamming distance between different windows to allow error correction. Hence, this approach is easily adjustable to different resolutions of the pattern image. Since the initial Morano algorithm was designed for generating color patterns, an additional connectivity constraint was introduced to ensure that those pixel positions that are adjacent (in an 8-neighbourhood) to an 'on' pixel are set to an 'off' status in our binary patterns. This isolation of 'on' pixels constitutes a important requirement to robustly segment a single dot during the decoding stage.

We use a dot pattern of size 1280×720 pixels which is based on an 80×45 binary pseudorandom array, 9×9 window property and minimum Hamming distance of 3. Currently, finer resolutions are not applicable as smaller dot sizes will further reduce the overall image brightness and as dot segmentation is prone to fail due to reduced sharpness of the pattern image.

2.3 Pattern segmentation

A crucial step in the workflow of a SL system is the segmentation of the pattern image. After the scene has been illuminated by the projector, the aim is to extract the altered pattern from the camera image in order to compute correspondences between the projected and the captured image pattern. In the following, we discuss the segmentation of the individual dot regions, as shown in Figure 2, in order to extract the distorted pseudorandom pattern. Two

methods based on absolute image intensities and image gradients are presented.

2.3.1 Intensity based pattern segmentation

The first approach is based on the idea that the pattern projected onto the objects to be measured is significantly brighter than the basic brightness of the scene. The goal is now to find an appropriate gray level threshold to separate the illumination pattern from its surrounding background. A common approach for image binarization is based on Otsu's method [9]. This algorithm assumes that the image contains two classes of pixels, i.e. foreground and background pixels, and calculates the optimum threshold separating the two classes by maximizing the inter-class variance of the corresponding gray level distribution. In the proposed approach, the threshold resulting from Otsu's method is used as a starting point. In particular, the mean size of the connected components, also referred to as Binary Large OBjects (BLOBs), of the binary image according to the calculated threshold is compared to the size of the dots in the initial pattern image. Depending on the deviation of the dot size in the camera image compared to the original dot size, the threshold is incremented or decremented iteratively until the average difference between the dot sizes lies beneath a given confidence level.

However, the use of a global threshold can lead to poor binarization results due to vignetting artifacts of the camera sensor, uneven illumination of the projector or varying reflectivity properties of the measured objects. Thus, the image is divided into smaller blocks and local thresholds are computed for each block center with missing thresholds obtained by bilinear interpolation to avoid blocking-artifacts in the resulting binary image.

2.3.2 Gradient based pattern segmentation

Instead of using absolute intensity levels, the second approach is based on intensity changes in the input image. As sharp brightness changes typically occur at object boundaries, an edge detection algorithm is applied to segment the individual dot regions. In particular, due to its low error rate and its accurate edge point localization, the well-known Canny edge detector is selected for our processing framework. In our experiments the gradient-based segmentation has proven to be more robust in low-light environments compared to intensity thresholding due to significant presence of image noise (see Figure 2 c and d).

2.4 Disparity computation

The final step of the SL decoding workflow is the actual computation of pixel disparities. Disparities are inversely proportional to corresponding depth values and denote the offset between codeword positions in the initial pattern image and their respective positions in the camera image captured of the illuminated scene. After the dots have been segmented according to one of the approaches presented in the previous section, incorrectly detected BLOBs are filtered out based on the size of the BLOB area as well as the length of and ratio between the axes of the corresponding bounding box. Next, the center position of each detected BLOB is calculated and mapped to the nearest grid location of the respective pseudorandom array grid. These offsets between the initial center positions and the corresponding grid positions - hereinafter also referred to as delta offset - can be used to account for subpixel accuracy in the disparity map. In the following, we present two methods using a codeword look-up technique and foreground/background segmentation, respectively, to compute pixel disparities for the

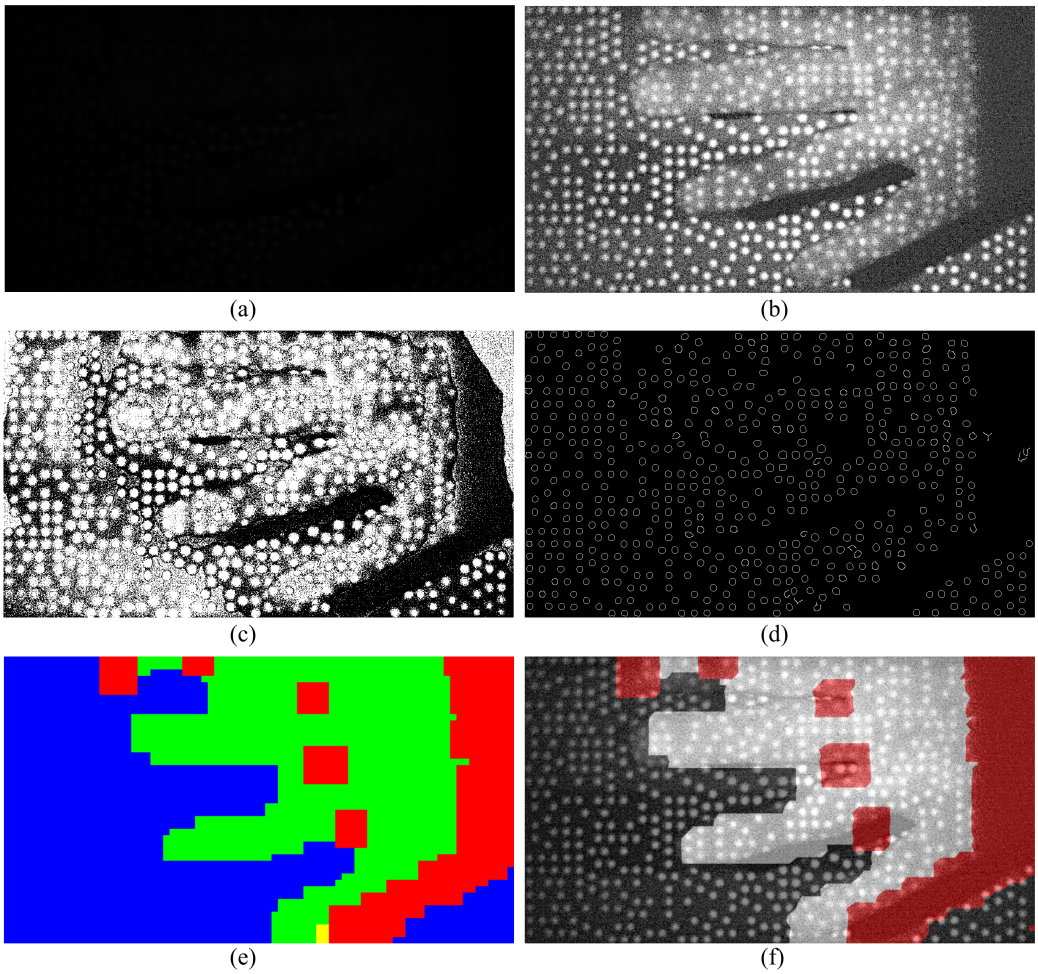

Fig. 2. Illustration of results obtained from different processing steps: (a) Input camera image. (b) Input image preprocessed. (c) Intensity thresholding. (d) Gradient-based edge detection. (e) Classification after foreground/background segmentation (green: foreground; blue: background; yellow: uncertain; red: undefined). (f) Computed final disparity map in gray value encoded representation, with initial dot pattern superimposed. Undefined regions are marked red.

reconstructed pseudorandom array.

2.4.1 Codeword look-up

The first approach is based on the actual decoding of the codewords contained in the pseudorandom array. According to the window property, for each window in the reconstructed array the corresponding codeword is extracted and compared to the codewords of the reference pseudorandom array using a look-up table that was constructed during initialization. For every valid codeword, the disparity is given by the horizontal offset between its respective position in the reference array (i.e. pseudorandom array of the projected pattern) and the reconstructed array (i.e. pseudorandom array of the captured image). The final disparity

map is computed by averaging disparities of overlapping windows.

A problem arises for invalid codewords which emerge due to inaccuracies of the dot segmentation process or due to incomplete windows at object boundaries. Specifically in the context of segmentation errors, codeword correction can help to improve the robustness of the reconstruction procedure. For example, if all codewords are separated by a minimum Hamming distance of three, a single error (i.e. one incorrect bit in the binary codeword string) can be detected and corrected. While the quality of the resulting disparity map is enhanced, the process of code correction increases significantly the overall runtime of the decoding process, as shown by the experimental evaluation in Section 3.

2.4.2 Foreground/background segmentation

Instead of retrieving the actual binary codeword of each individual window, the second approach is based on changes between the reconstructed and reference pseudorandom array to segment the image into different categories (i.e. foreground, background, uncertain and undefined) using binary morphological operators. In the best case scenario - when there is no vertical offset between the positions of the camera and the projector, and the captured image is perfectly aligned in vertical direction with the projected image - the pattern illuminating the scene is only shifted in horizontal direction when hitting an object. Thus, areas that remain unchanged between the reference and reconstructed pseudorandom array are considered as *background*, i.e. areas with zero disparity. These areas can be determined by computing the difference between the two arrays, which causes pixels in the alleged background regions to be blanked out. In principle, this constitutes a simple and computationally fast method to detect steady (i.e. background) areas in the image and also helps to restrict the search space for correspondences of foreground objects.

However, a special case has to be considered that may also yield vacant areas in the difference image, i.e. if similar codewords overlap due to pattern shifting. This problem has to be taken into account in the selection and generation of the used pattern image and can be reduced by selecting a larger Hamming distance between adjacent codewords. As the effected areas are usually smaller than the window size of the corresponding pattern, background regions whose size falls below a given threshold according to the window property are defined as *uncertain*.

Moreover, the reconstructed pseudorandom array might contain vacant regions caused by several reasons: unsuccessful dot segmentation due to low image contrast, strong presence of image noise or poor reflectivity properties of the measured object, shadows caused by a foreground object or missing information at image boundaries due to the aforementioned pattern shift. Since the computation of disparities is not possible in these regions, they are labeled as *undefined*. Finally, the remaining areas of the pseudorandom array which are not marked as background, uncertain or undefined are considered as *foreground*.

The final step of the disparity computation pipeline includes a rescaling to the initial image size, under consideration of the delta offset values and using triangle interpolation of adjacent grid points. Furthermore, image inpainting techniques can be used to fill in the remaining undefined disparity values. A fast and simple inpainting approach is currently implemented to fill the blank regions on a scanline basis by replicating the maximum value of the two disparities corresponding to the pixels located at the left and right hole boundary, i.e. the

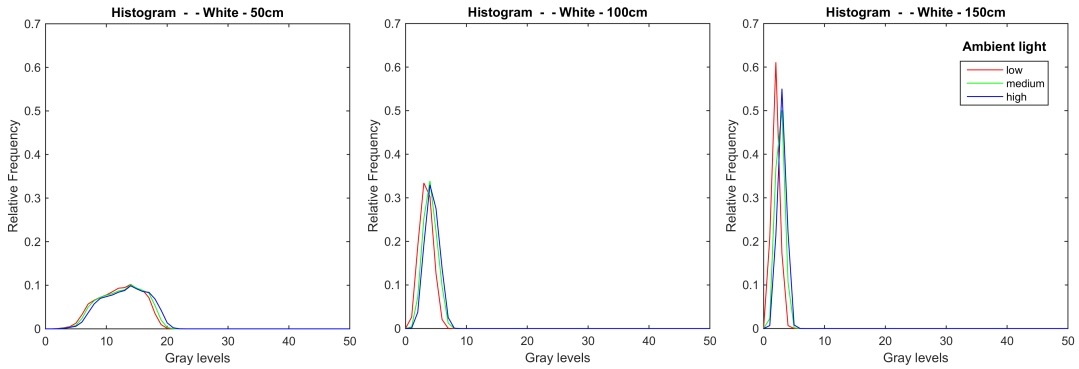

Fig. 3. Histograms of a white frame captured under different lighting conditions and distances.

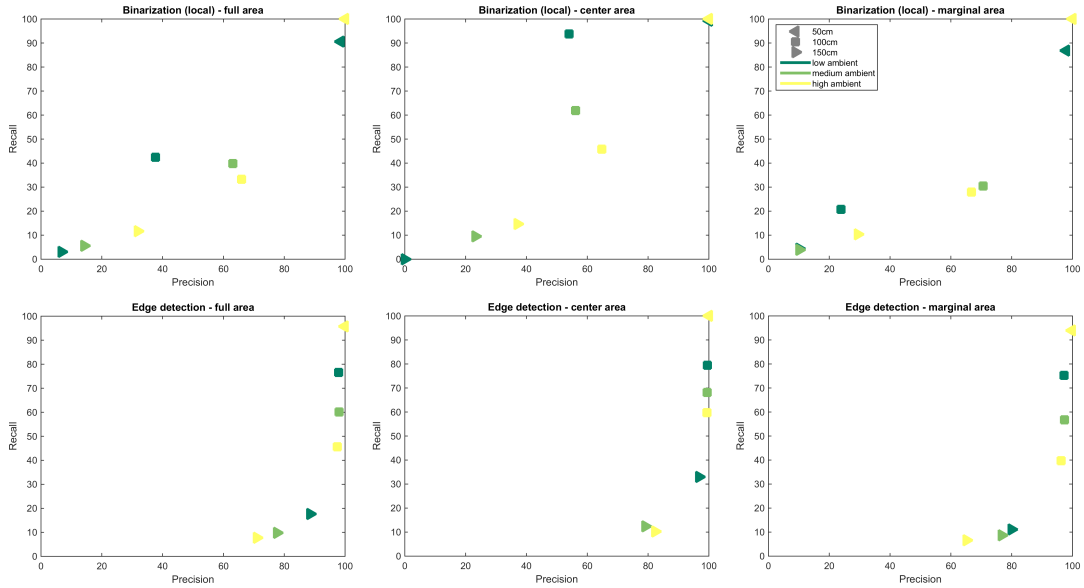

Fig. 4. Precision/Recall plots based on dot retrieval using reference pattern. In the first, second and third column, results are presented regarding the whole image, the center and the marginal area, respectively. Results are generated by using local intensity thresholding and edge detection.

disparity of the object located closer to the camera.

In Figure 2 e, an example of a segmentation result is presented with foreground, background, uncertain and undefined areas colored in green, blue, yellow and red, respectively. Based on the concept of sliding-window stereo matching techniques, each connected component of the foreground segmentation mask is shifted horizontally to calculate the respective disparity values based on the Sum of Absolute Differences (SAD) metric. The resulting disparity map after upscaling and inpainting is shown in Figure 2 f. Darker areas in the disparity map indicate smaller disparity values (i.e., regions in the background), brighter areas correspond to the foreground.

3 Evaluation

In this section, we first evaluate the working range of our SL set-up in terms of captured intensity distribution and successful dot detection. Then we investigate the runtime measured from our implementation of the overall system with different versions of the involved algorithmic components.

3.1 Working range

The available projector brightness should guarantee that the projected SL pattern can be segmented and extracted from the captured camera image to enable the decoding process. For that purpose, a completely white frame (i.e. all intensities are set to 255 using 8 bit depth) is projected on a plain white wall at three different distances (50 cm, 100 cm and 150 cm) under various indoor lighting conditions. As can be seen in Figure 3, the captured intensity values at a distance of 50 cm deviate by 94.90% from the originally projected intensities. For distances of 100 cm and 150 cm naturally the brightness of the captured camera image further decreases leading to a narrower distribution of pixel intensities. In particular, values only range from 1 to 8 and 0 to 5 reaching their peaks at 4 and 3, respectively. This corresponds to a brightness reduction of almost 99%, and the narrow histogram distributions hinder the setting of suitable threshold values for segmentation. It is worth noting that a plain white wall nearly represents an ideal test case and materials like plastic or fabric additionally diminish the captured pixel brightness, as experiments have indicated. Since our evaluation has been conducted indoors using artificial light sources (i.e. very small amount of infrared radiation), changing ambient light conditions have only small impact on the intensity distributions.

To quantify the quality of the pattern segmentation, a reference pattern - i.e. a pattern where every second grid position is set 'on' - based on an 80x45 random array is projected and the number of retrieved dots is measured. Additionally, to investigate the influence of decreasing projector brightness, the captured image is divided into a 4×4 grid of equally sized rectangles and the detection rates of the 12 rectangles at the marginal area and the remaining 4 rectangles in the center area are analyzed separately. This procedure is motivated by the fact that the brightness of our projector cannot be altered manually and that the currently available brightness of the projector is already very low.

Comparing the ratio between the maximum brightness of the center and the marginal area, experiments have shown that at 50 cm the intensity at border regions reaches between 89.47% and 95.46% compared to the center area for varying lighting conditions. At 100 cm and 150 cm, the ratio falls off to 40% and 30.77%, respectively. Figure 4 shows the precision/recall results of the dot detection based on the segmentation approaches discussed in the previous section. As can be seen, the detection rate drastically decreases with increasing distance. In particular, from 50 cm to 150 cm the recall rate using local image thresholding drops from 90% to 3%, falling already below 45% at 100 cm. Compared to global or local image thresholding, edge detection shows more robust segmentation results, i.e. up to 77% recall at 100 cm while preserving almost 100% precision. Additionally, it can be noticed that there is only a minor difference between precision/recall rates at central and marginal areas when edge detection is used. For example, while for image thresholding the recall score differs by 74% at 100 cm under low ambient light conditions, edge detection results vary only by 5%. In summary, it can be stated that the currently available projector brightness solely allows

for acceptable dot detection rates (95% recall at 100% precision on average) up to a distance of 50 cm in this ideal experimental assembly. We also observed that varying ambient lighting conditions generated by indoor light sources had only minor impact on the detection rate at this distance.

3.2 Computation time

To evaluate the runtime performance, the structured light system was positioned at a distance of 50 cm in front of a white wall. A planar object was then placed at a distance of 7.5 cm, 19 cm and 26.5 cm in front of the wall and illuminated by a dot pattern based on an 80 x 45 random array. In Figure 5, the average runtime results computed over 10 passes are shown, grouped by different combinations of decoding strategies. The three bars per group correspond to the runtime results obtained by capturing the object at the aforementioned three distances. It can be seen that the disparity computations based on codeword look-up and foreground/background (FG/BG) segmentation require almost the same overall runtime, with higher average values obtained when using edge detection rather than image thresholding for pattern segmentation.

The overall runtime can be split up into three major components: Initialization, threshold computation (only used by image thresholding) and disparity computation. Initialization sets up the internal parameters according to the properties of the specified pattern image and builds the corresponding codeword look-up table. This takes 12 milliseconds (ms) on average. For image thresholding, the computation of the binary thresholds requires about 54 ms for a distance of 7.5 cm. When the object is placed farther away from the wall (i.e., closer to the camera), the related runtime increases to 62 ms at 19 cm and 131 ms at 26.5 cm. This is due to the fact that the algorithm tries to find the optimal thresholds according to the dot size. However, as the object moves towards the camera, there are some unilluminated regions in the captured frame since the pattern image partially shifts outside the viewing area. A similar behavior can be observed for the disparity computation including codeword correction. While the disparity computation (which also includes the dot segmentation) takes on average 86 ms for the image thresholding approach and 228 ms using edge detection, the runtime rises at the aforementioned distances to 204, 496, 1007 ms and 341, 642, 989 ms, respectively, when codeword correction is applied. In this context, the growing number of undefined pixels should also be taken into account. As illustrated in Figure 6, the amount of invalid pixels exceeds 18% at 19 cm and 45% at 26.5 cm, which can be attributed to the same reasons as for threshold computation. The corresponding percentages of corrected codewords are about 15% and 6%, respectively. It should be noted that the total number of codewords is smaller than the number of pixels according to the window property.

4 Conclusion

In this paper, we have investigated the properties of a depth sensing system based on near-infrared SL illumination. Based on the projection of a so-called 'single-shot' pseudorandom binary pattern, techniques for projector/camera image rectification with a suitable calibration pattern and dot segmentation under low-light projector illumination were explored. Experiments have shown that currently a limited working range of 50 cm and a pattern image based on an 80×45 pseudorandom array are feasible.

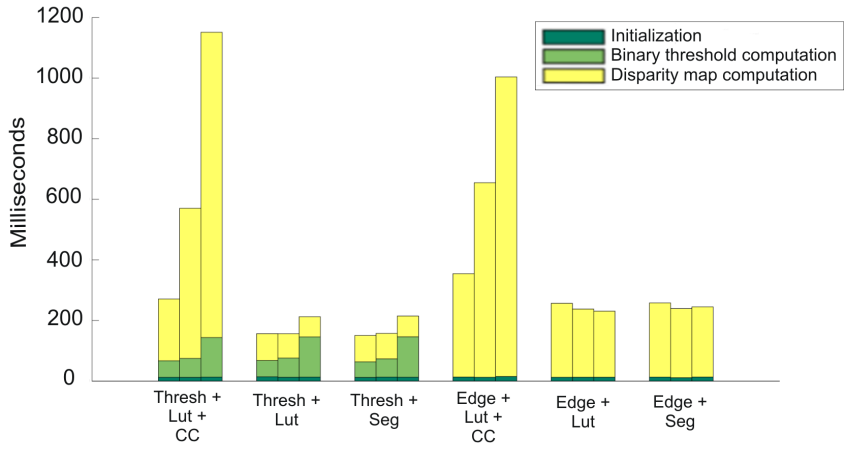

Fig. 5. Runtime results grouped by different combinations of decoding strategies. The three bars per group correspond to the runtime results obtained by capturing a planar object at 7.5 cm (left), 19 cm (middle), and 26.5 cm (right) in front of a 50 cm distant wall. Abbreviations: Thresh = image thresholding, Edge = edge detection, Lut = codeword look-up table, Seg = FG/BG segmentation, CC = codeword correction.

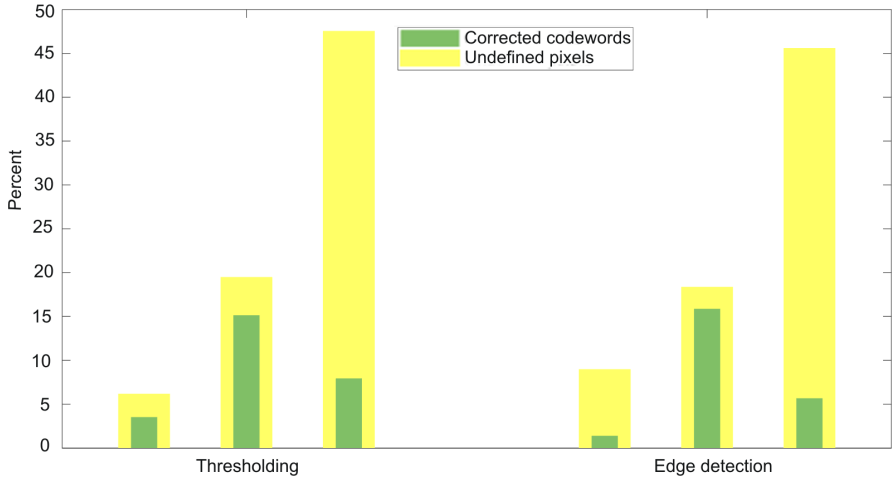

Fig. 6. Percentage of undefined pixels and amount of corrected codewords grouped by the two dot segmentation strategies: image thresholding and edge detection. The three bars per group correspond to results obtained at distances of 7.5 cm, 19 cm, and 26.5 cm.

When investigating the temporal behavior of different algorithmic components, we found that the more robust pattern segmentation based on edge detection, as opposed to local image thresholding, was accompanied by a certain increase of the overall runtime (e.g., from about 150 ms to about 250 ms). The additional incorporation of codeword correction strategies based on the Hamming distance led to a further increase in runtime, which became particularly

pronounced for test scenes with large disparity differences.

In future work, the implemented strategies for disparity computation between the original and captured pattern image, which currently rely on a codeword look-up table or foreground/background segmentation, could be expanded to include active stereo matching techniques. Moreover, enhancement of the projector brightness would allow to use modulations of gray values instead of strictly binary patterns to enrich the underlying codeword alphabet and enable the application of smaller window sizes and denser pattern images.

Acknowledgements

This work has been supported by the Austrian Research Promotion Agency (FFG) under project Robust Depth 3D (project no. 848163).

References

1. L. De-Maeztu, S. Mattoccia, A. Villanueva, and R. Cabeza. Linear stereo matching. In *IEEE International Conference on Computer Vision*, pages 1708 – 1715, 2011.
2. J. Geng. Structured-light 3d surface imaging: a tutorial. *Advances in Optics and Photonics*, 3(2):128–160, June 2011.
3. A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz. Fast cost volume filtering for visual correspondence and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(2):504–511, 2013.
4. T. Jia, Z. Zhou, and H. Gao. Depth measurement based on infrared coded structured light. *Journal of Sensors*, 2014:1–8, 2014.
5. H. Kawasaki, R. Furukawa, R. Sagawa, and Y. Yagi. Dynamic scene shape reconstruction using a single structured light pattern. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, June 2008.
6. Y. Lei, K. Bengtson, L. Li, and J. Allebach. Design and decoding of an m-array pattern for low-cost structured light 3d reconstruction systems. In *20th IEEE International Conference on Image Processing (ICIP)*, pages 2168–2172, Sept. 2013.
7. R. A. Morano, C. Ozturk, R. Conn, S. Dubin, S. Zietz, and J. Nissanov. Structured light using pseudorandom codes. *IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI)*, 20(3):322–327, Mar. 1998.
8. D. Moreno and G. Taubin. Simple, accurate, and robust projector-camera calibration. In *Proceedings of the Second International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, pages 464–471, Oct. 2012.
9. N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, Jan 1979.
10. J. Salvi, S. Fernandez, T. Pribanic, and X. Llado. A state of the art in structured light patterns for surface profilometry. *Pattern Recognition*, 43(8):2666 – 2680, 2010.
11. J. Salvi, J. Pages, and J. Batlle. Pattern codification strategies in structured light systems. *Pattern Recognition*, 37(4):827 – 849, 2004.
12. D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1/2/3):7 – 42, 2002.
13. D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages I:195–I:202, June 2003.
14. Y. Taguchi, A. Agrawal, and O. Tuzel. Motion-aware structured light using spatio-temporal decodable patterns. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part V (ECCV)*, pages 832–845, 2012.
15. L. Zhang, B. Curless, and S. M. Seitz. Rapid shape acquisition using color structured light and

multi-pass dynamic programming. In *Proceedings of the 1st IEEE International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT)*, pages 24–36, 2002.

16. Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22(11):1330–1334, Nov. 2000.