Computer-Aided Design 114 (2019) 91–100

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Fabricating QR codes on 3D objects using self-shadows☆,☆☆

Hao Peng [a], Lin Lu [a,*], Lin Liu [a], Andrei Sharf [b], Baoquan Chen [c]

[a] School of Computer Science and Technology, Shandong University, Qingdao, China
[b] Ben-Gurion University, Israel
[c] Peking University, China

ARTICLE INFO

Keywords:
3D QR code
3D printing
Ambient occlusion
Arbitrary surface

ABSTRACT

QR codes are essentially 2D matrix images that can be easily displayed physically on printed media or digitally on a screen. Their popularity is due to their fast readability and storage capacity. Applications vary widely from product tracking and identification to entertainment, advertising, encryption, and payment. In this work, we explore a novel representation of QR codes as 3D structures that are embedded on arbitrary shapes. Specifically, our method physically prints the 3D QR-code representation on a given region on the shape. Thus, in comparison to paper printed QR-codes, our 3D QR-codes are more resilient and reliable (e.g. harder to break and reproduce). 3D QR-codes are computed from their 2D counterpart as a carefully designed set of carved geometry modules on the surface which encode the black-and-white QR pattern. At the core of our method, we utilize self-shadows which are cast by 3D geometry modules on the surface to generate a correct black–white pattern with adequate contrast which can be easily decoded by standard QR readers. Our technique allows embedding 3D QR-codes on arbitrary objects with minimal modification of the shape and even on thin shell surfaces or highly curved ones. Our technique is robust to lighting environments and scanning angles and allows to easily decode 3D QR-codes with similarity to the 2D version. Finally, we demonstrate our technique feasibility by fabricating a wide range of examples using consumer-level 3D printers and common homogeneous materials. The feasibility and robust readability make 3D QR-codes a good candidate for embedding information on physical objects and open interesting directions in 3D manufacturing such as 3D printed objects ownership and 3D watermarking.

1. Introduction

QR-codes (Quick Response Codes) are machine-readable optical 2D matrix barcodes that are used to store data efficiently. The QR-codes became popular due to their fast readability and storage capacity. Initially, QR-codes were designed for the car industry in Japan to allow high-speed component scanning and track vehicles during manufacturing. QR codes are now used in a much broader context, including both commercial tracking applications and convenience-oriented applications aimed at mobile phone users (termed mobile tagging). Applications include product tracking, item identification, document management, and general marketing (e.g. ticketing, product labeling, mobile payment systems, etc.).

QR-codes are essentially 2D images representing a matrix barcode that are typically displayed using printed media such as billboards and magazines or using digital displays such as mobile and TV screens. This presents some limitations in the usability and generality of QR-codes. Physically printed paper QR-codes are naturally non-weatherproof and easily get damaged by climate conditions or external forces. Additionally, they can be easily forged using standard copy-machines. Traditionally, QR-code recognition operates on planar displays while curved surfaces yield low recognition rates. This becomes a limitation when displaying QR codes on non-planar billboards.

In this paper, we take a new approach and model QR-codes as 3D printable structures embedded in arbitrary shapes. Thus, our technique computes a 3D representation of the QR-code that can be optimally embedded into the 3D surface. Our optimization takes into account 3D printability constraints, QR-code readability and the visual appearance of the 3D shape. In contrast to traditional 2D QR-codes, our 3D QR-codes are not limited to planar surfaces and can be embedded in highly curved shapes.

☆ This paper has been recommended for acceptance by Pierre Alliez, Yong-Jin Liu & Xin Li.

☆☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.cad.2019.05.029.

* Corresponding author.
E-mail addresses: ph1994wh@gmail.com (H. Peng), llu@sdu.edu.cn (L. Lu), lorrain0407@gmail.com (L. Liu), asharf@gmail.com (A. Sharf), baoquan.chen@gmail.com (B. Chen).

https://doi.org/10.1016/j.cad.2019.05.029
0010-4485/

Our optimization allows embedding 3D QR-codes even in thin shells requiring minimal depth. Finally, our 3D QR-codes can be manufactured using commercial-level 3D printers with standard resolution.

QR-codes readability relies on the encoding standard and the contrast between the black and white sections. With the rapid development of 3D printing technology, 3D QR-codes printed with two color materials on flat surfaces have been proposed. Nevertheless, this requires a complex setup as consumer-level 3D printers normally perform using a single material at a time. In this work, we avoid to elaborate 3D printing approaches and perform standard 3D printing using a single homogenous material. Our key idea is to modulate self-shadows that are cast by the 3D QR-code geometry onto the 3D shape to obtain the black and white patterns. Specifically, given a 2D QR-code and 3D shape, we compute carving modules onto the 3D shape that cast shadows onto the surface and generate the black and white pattern. The carving modules are optimized such that the cast QR-code is easily scanned via standard readers and the object is of minimal modification and within 3D printing constraints.

Optimally embedding QR-codes on surfaces is a challenging problem especially in highly curved geometries and thin shell surfaces. To achieve the adequate black–white contrast such that QR-codes are readable, carving depth should be large enough for casting adequate shadows on the surface. Nevertheless carving depth is often limited by shape constraints (thin shells) as well as 3D printability constraints (in terms of support). In this view, it is necessary for our optimizer to reduce carving depth, generating as shallow as possible modules while maintaining adequate contrast for decoding.

Our work makes the following main contributions:

- We propose the novel approach of manufacturing 3D QR-codes embedded in 3D shapes. The 3D QR-codes are printed using homogeneous materials and scanned by standard readers.
- We obtain black–white patterns by utilizing self shadows that are cast on the 3D surface from the 3D QR-code modules. We introduce an optimization scheme on both QR modules and the 3D QR-code geometry.
- We introduce a simulator of self shadows for evaluating the readability of the embedded 3D QR-codes.

2. Related work

In our work, we enhance QR-codes to allow their digital 3D fabrication and utilize this new approach to convey information on 3D printed objects via 3D QR-codes embedding. In the following, we discuss works from both QR-code and digital manufacturing domains in our context.

QR-code enhancement In recent years, researchers have paid much attention to the modification and enhancement of QR-codes with visual features beyond their ordinary appearance [1,2]. Cox [3] proposes an algorithm to encode binary image content as redundant numeric strings appended to the original data. Chu et al. [4] firstly combine halftone images and QR codes. They generate a binary embedding by subdividing each QR module into 3×3 submodules to embed the halftone image and then optimize the binary pattern of each module to achieve both decoding robustness and image quality. Lin et al. [5] facilitate the process of embellishing a QR code by embedding images into a QR code using an error-aware warping technique and stylizing the black and white squares using a binary exemplar. Garateguy et al. [6] embed QR codes into color images and optimize the concentration of pixels and its corresponding luminance to minimize a visual distortion metric. Lin et al. [7] further propose a nearly real-time rendering mechanism to improve the visual quality of QR codes while avoiding affecting their decodability.

QR-code images captured by mobile phones are usually distorted, low quality and may consist of nonuniform illumination, noise, and blur. Researchers have made significant efforts to enhance the robustness of the decoding process [8–10]. Our 3D QR-codes can be scanned and decoded with mobile phone cameras in the same manner as 2D QR-codes are processed. Thus, our method has the same readability as well as camera noise and distortions as the 2D case.

Similar to us, efforts were made to enhance the decoding capabilities and robustness of non-planar QR codes. Li et al. [11] manage to extract the finder patterns and code boundary on a cylindrical surface under the constraints that the two vertical boundaries are parallel to each other and parallel to the generatrix of the cylinder. Lay et al. [12] further lift the constraints and rectify the distortion for QR images posted on cylinders, such that the QR code can be recognized. Our work takes a large step in this direction, allowing to embed QR-codes in any 3D shape not limited to a specific curvature or geometry structure.

Light and shadow from 3D printed shapes. With the advent of commodity 3D printing, we observe the emergence of a branch of works focusing on 3D shape modeling and printing for the creation of various optical phenomena.

Mitra and Pauly [13] optimize 3D shapes to cast varying shadow images when lit from different directions. Their geometric optimization computes a 3D volume whose cast shadows best approximate the provided input images. Our work drives inspiration from their approach in designing 3D structures to control the cast shadows. Nevertheless, our goals are significantly different as self shadows by the 3D QR-code should be highly accurate and recognizable by standard QR readers. Additionally, our 3D embedding should account for minimal modification and printability of the overall 3D shape.

Alexa and Matusik [14] construct relief surfaces whose diffuse reflection approximates given images under known directional illumination, however without considering self shadows or other shading effects. In a follow-up, they propose a method for generating images from surfaces with self-occlusions. Their technique computes a distribution of small tubes with varying length on the surface which create the desired shading effect by controlling the amount of trapped light in the tube [15].

ShadowPix is introduced [16] to compute surfaces that can both cast and receive self-shadows forming different prescribed images depending on the light direction. Similar to us, they modulate a grid of 3D elements in the surface to create light and shadow effects for a given light direction. Their technique aims at reconstructing gray-scale images on planar surfaces while ours aims at producing accurate white–black QR patterns embedded on arbitrary, possibly highly curved shapes.

Wang et al. [17] introduce a general discrete version of congruences based on piecewise-linear correspondences between triangle meshes into lighting and shape analysis. A particular application of such structures is freeform shading and lighting systems for architecture. Their method places emphasis on the light interaction between the model and the environment, while our method focuses on the self-occlusion of the model.

Schüller et al. [18] introduce appearance-mimicking surfaces which are a generalization of bas-reliefs and reproduce the look and details of a 3D object on a thin surface when observed from designated viewpoints. They focus on computing height fields and normals to resemble the 3D model without accounting for self-shadow effects.

In general, bas relief generation on arbitrary surfaces is a well-researched topic [19–22]. These techniques involve gradient domain processing and generate 3D like structures on planar

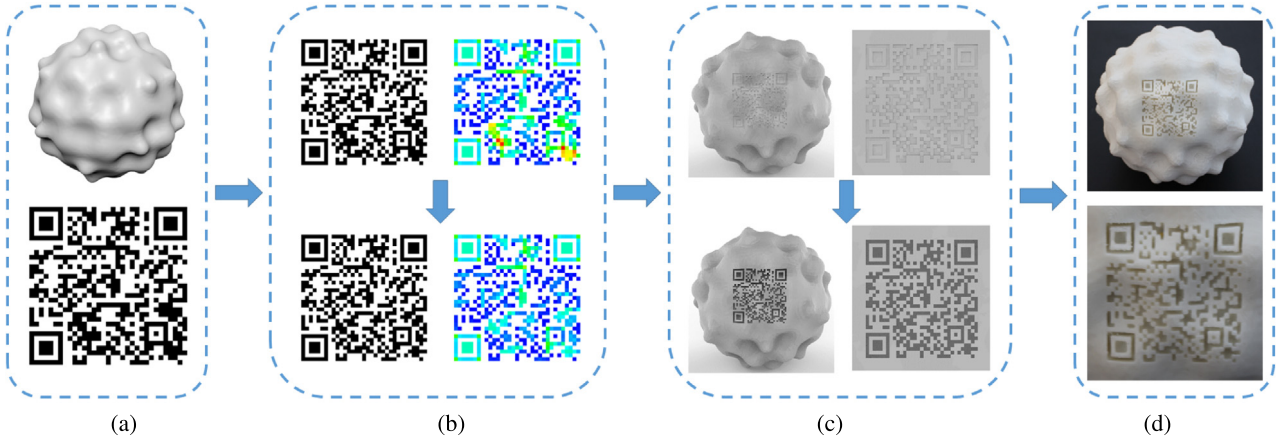

Fig. 1. Given a QR code and a 3D shape (a), our method optimizes the QR code to suppress black modules continuity (b), and iteratively carves the 3D QR (c) and achieves an optimized 3D printed QR code that is readable by a standard decoder (d).

surfaces. Nevertheless, these techniques have little in common to us as they do not consider self-shadows and are limited to shallow 3D geometry embedded onto relatively simple surfaces.

3D printed perforated lampshades [23] is yet another example of *play-of-light*. They investigate light projected footprints to achieve continuous grayscale images via the optimization of tiny tubes on the surface shell.

Li et al. [24] introduce AirCode, a technique that allows embedding QR-codes inside physically fabricated objects without changing their appearance. In their work, air pockets are produced beneath the surface during 3D fabrication, affecting the scattering light and creating black–white patterns on the surface. Similarly, QR codes are embedded inside 3D objects for identification purposes [25]. By taking advantage of additive layer-by-layer manufacturing, codes are segmented and embedded in numerous object layers without interfering with the surface. Since their pattern lies beneath the surface it suffers from low contrast and is not easily decoded without proper lighting or when printed with challenging matte materials. In contrast, our goal is to embed QR codes with robust readability and minimal geometry distortion.

On the same path, Wei et al. [26] embed QR codes for anti-counterfeiting utilizing SLM-based 3D printing. The QR is fabricated with a specific "tagging" material such that it can be recognized via X-ray imaging.

Recently Kikuchi et al. [27] suggested an embedding of QR codes onto CAD models represented by B-spline surfaces. In contrast, our method focuses on general meshes and targets for optimized modulation. With the valid simulation, our optimization is performed on both QR module distribution and the carving depth, and thus allows generating the optimized QR codes which have better characteristics w.r.t. their carving distribution and depth. I.e., our results offer the robust decodability with fewer modulations on the shape.

3. Overview

Given a 2D QR code image Q and an arbitrary 3D shape S, our aim is to carve the 2D QR code onto the surface, such that it can be easily read and decoded by common decoders (such as mobile phones). Thus, our 3D QR-code carving objectives are:

- robustness: 3D-QR-code should be easily decoded in presence of various lighting environments and from a large range of angles and positions.
- generality: 3D QR-codes should permit their embedding on arbitrary surfaces possibly with high curvature, thin shells, etc.
- printability: 3D QR-codes should account for printability constraints such as physical resolutions.
- appearance: 3D QR-codes should have minimal influence on the appearance of the 3D shape.

To accomplish these goals, we optimize the QR-code data and its 3D geometry embedding. Since it is infeasible to compute the exact lighting phenomena in a physical scene, we use a simulation approximation and calibration of the 3D QR code. Thus, we optimize the 3D QR code embedding based on our environmental simulation and finally 3D print it.

A standard QR code [28] is constructed of square modules, set in a regular square array and consists of function patterns and encoding regions. We do not use the function patterns for data encoding and thus focus on the encoding regions involving data and error correction codewords. To successfully decode a 3D QR code on a general surface requires the captured code to be of good alignment, sufficient resolution and high contrast.

Thus, the 3D QR-code generation consists of carving the shape 3D geometry to provide the correct black–white pattern on the surface with proper contrast. Black areas, representing black modules, are occluded regions which are shadow cast by the surface. Similarly, white modules are obtained in unoccluded regions on the surface which cast direct light.

Our optimization scheme includes two parts, QR optimization and 3D carving optimization (see Fig. 1). First, we optimize the black and white module distribution to enhance the QR contrast. Intuitively, black modules should have a small area in order to be easily occluded. If black modules have a large area, occluding them is hard requiring large occluders. We re-distribute the black modules to optimize their size, suppressing largely connected black regions. This yields good occlusion of the black modules and a good contrast in the QR image.

Second, to account for appearance and printability, we minimize the carving depth for less support during printing process and limit it not to exceed the maximum thickness of the 3D surface. We assume that the QR codes are scanned in an ordinary environment, from the top view with no additional light. We also assume that the target 3D-QR region is given, either a non-salient or a salient area depending on the applications.

3D carving optimization starts by projecting the QR code on the surface. Starting with a uniform carving depth for all black modules, we simulate the resulting QR image in the 3D scene. While the simulated QR image has not met the overall contrast threshold, we carve the vertices on the surface belonging to a black module. We repeat carving until the contrast exceeds the decoding threshold value.

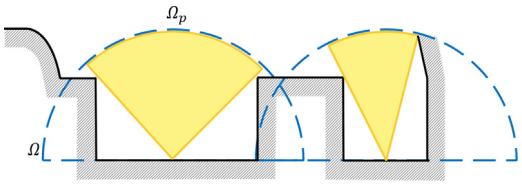

Fig. 2. Illustration of the visibility computation.

4. QR simulation

To evaluate our 3D QR code structure in terms of readability, we simulate the self shadows and generate the QR code image from its 3D structure projection. The self shadows are generated by both carved modules and the surface geometry.

4.1. Light model

We assume that the 3D carved shape S is a Lambertian surface and that the dominant illumination in the environment is ambient lighting. The illumination for each vertex point in the QR region is determined by lights and self-occlusions casting shadows on the surface. Essentially, the carved black modules are occluded by nearby white module structures.

For each 3D vertex point p on S with normal $\vec{n}$, we compute its illumination value following the ambient occlusion [29] method. The ambient occlusion AO for p can be computed by integrating the visibility function over the hemisphere Ω with respect to projected solid angle:

$$AO(p) = \frac{1}{\pi}\int_{\Omega} V_{p,w}(\vec{n}\cdot\omega)\,\mathrm{d}\,\omega = \frac{1}{\pi}\int_{\Omega_p}(\vec{n}\cdot\omega)\,\mathrm{d}\,\omega, \tag{1}$$

where $V_{p,w}$ is the visibility function at p, defined to be 0 if p is occluded in the direction ω and 1 otherwise, and $d\omega$ is the infinitesimal solid angle step of the integration variable ω. For each vertex point p, we can restrict the integral domain on the hemisphere to its visible area, i.e., the integral is computed in $\Omega_p \subset \Omega$, where $\Omega_p = \{\omega | V_{p,\omega} = 1, \omega \in \Omega\}$. (See Fig. 2)

We denote the intensity of ambient illumination as l_a. The measured illumination for p is computed as follows:

$$I_p = \rho\pi l_a AO(p), \tag{2}$$

where ρ is the albedo at p.

In practice, the computation is performed as a discrete manner. Thus, we subsample each module uniformly using 5×5 vertex points and compute their local illumination, i.e., the visible area Ω_p, $AO(p)$ and I_p. This yields a sufficient approximation of the illumination per module and consequently their black and white contrast.

4.2. Illumination calibration

Once illumination is computed we can construct a simulated QR code image. Still a calibration mapping is required from the simulated image to the actual image. To map the illumination values computed in Eq. (2) to actual gray scale values, a gamma correction is needed. Next, we carry out physical experiments to obtain the correct gamma correction parameter.

We design a reference model and fabricate it with white PLA materials by an FDM 3D printer. On the plate, we uniformly arrange carved modules as square holes varying in size and depth (see Fig. 3a). The size increases from left to right, and the depth increase from top to bottom. The albedo value of white PLA material is $\rho = 0.87$.

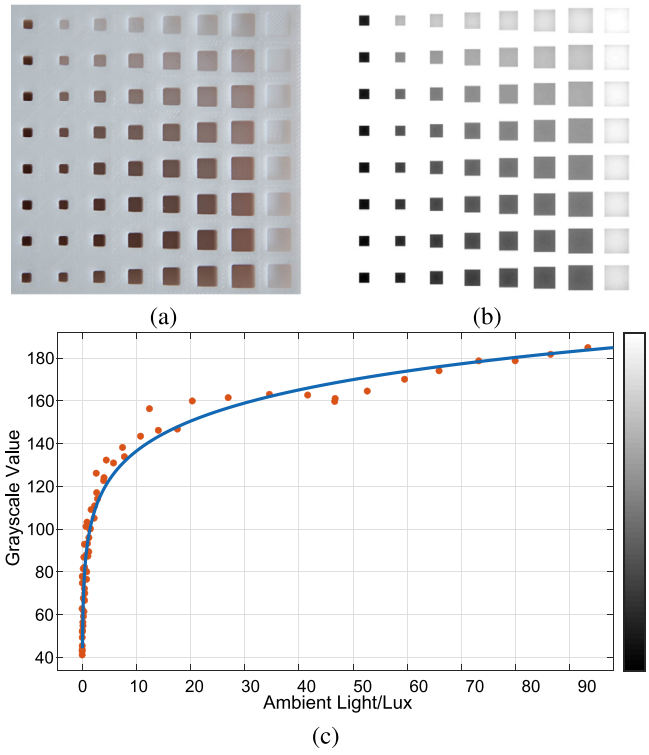

Fig. 3. From the physical reference model (a), we calculate its irradiance (b), and fit a curve to reflect the relationship between irradiance and grayscale value (c).

We first photograph the physical model with ambient light only and obtain the gray scale image (Fig. 3a). The ambient illumination is $l_a = 112.2$ lux based on our test. For each point p in the carved region, we calculate its illumination value l_p following Eq. (2), and visualize the values in Fig. 3b. Then we can explore the relationships between the gray value and its according illumination value, as shown in Fig. 3c.

5. QR optimization

To optimize the 3D QR code embedding with the objectives for the minimal modulations on the general surfaces, we carefully design a two-stage scheme, modules optimization and depth optimization.

5.1. 3D QR carving

Given a 2D QR code image and a 3D shape with target QR position, we project the 2D QR code image onto the 3D surface generating a QR region on the surface, as shown in Fig. 4. We assume that the carving region is continuous, with no self occlusions and that the projection direction is in the normal direction to the surface as QR codes are typically scanned from top-view. The projected QR image on the surface defines a partition of the surface vertices inside the QR region into black and white modules. Note that we remesh the QR region on the surface into a dense triangulation. Particularly, we carve the vertex of the QR image along the projection direction $\boldsymbol{d}$ to its new position $p(t) = p_0 + t\cdot\boldsymbol{d}$, where p_0 is the intersection of projection ray and the object's surface and $t = 0$ for the vertices of white modules.

5.2. Module optimization

In our 3D QR-code, black modules are obtained in the 3D scene due to self occlusions on the object's surface which cast shadows.

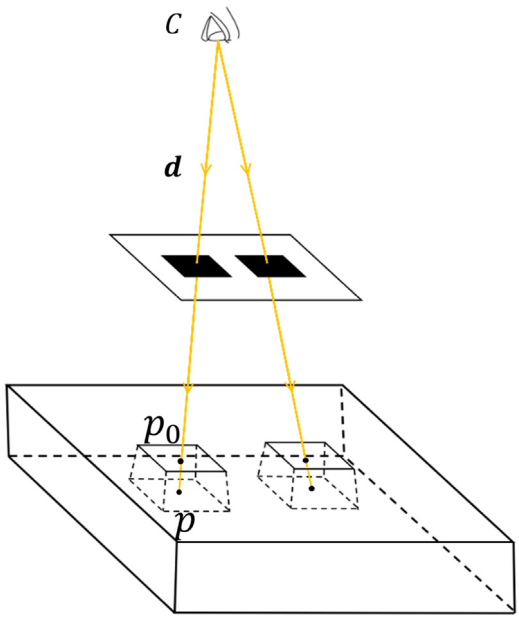

Fig. 4. Initial generation of 3D QR code with the most shallow depth.

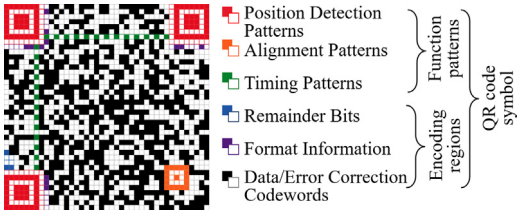

Fig. 5. Structure of a QR code symbol.

Thus, black modules are generated by carving them on the shape and occluding them from the light.

We perform the optimization on the modules in the *data/error correction codewords* of the *encoding regions* (Fig. 5). We first compute the intensity for each black module, via a weighted average illumination of its sample points:

$$I(m) = \sum_{p_j \in m} \omega_{p_j} I_{p_j}, \tag{3}$$

where p_j represents the sample point in the module m, ω_{p_j} is the weight value of p_j using Gaussian distribution in all points of m.

From the simulation, we can see that the shadow intensity varies in the modules, depending on their distributions and the embedding 3D shape. Naturally, the black module area determines the size of its occluder, i.e. the larger the black area is, the bigger occluder we need in order to cast enough shadow on it. In other words, with the same carving depth, the larger the black area is, the less contrast it provides (see Fig. 6).

Therefore, we optimize the modules distribution to increase the contrast of carved black modules. The objective is to minimize the continuous area size for black modules by suppressing their continuity without hampering QR correctness. Let M denote the variable set formed by all the black modules in data/error correction codewords. We formulate the objective function E_m as follows,

$$E_m(M) = \frac{(1-\lambda)}{N} \sum_m \frac{I(m)}{I(m_0)} + \lambda L(Q), \tag{4}$$

where $I(m_0)$ represents the original illumination for module m, and N is the number of black modules in M. The first item is the total illumination for the black modules, and the second item is the loss of redundancy of the QR code. The coefficient λ indicates the weight on the robustness of the optimized QR. We use $\lambda = 0.1$ in the experiments.

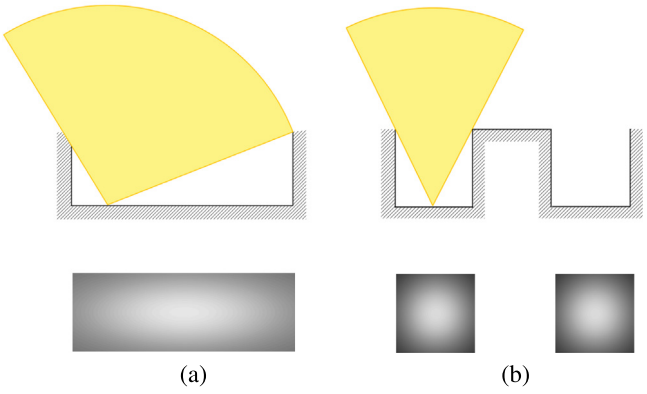

Fig. 6. Continuity suppression 1D illustration. By suppressing the black modules continuity (a), we obtain larger contrast (b). The bottom images indicate the self-shadows.

Loss function $L(Q)$. The rationale for the optimization on modules is the redundancy in the QR codes that allows the modules to be partially changed as long as the change does not violate the readability of the QR-codes. The term $L(Q)$ is to evaluate the loss of redundancy from the original code Q_0 to the optimized one Q.

Based on the standard [28], the data/error correction codewords of a QR-code Q generally form into *blocks* $\{B_i\}_{i=1}^{n_B}$. Embedded data is evenly distributed in each block, and then the block exploits Reed Solomon algorithm to generate a series of error correction codewords placed after the data codeword sequence. Suppose that there are n_C codewords $\{C_j\}_{j=1}^{n_C}$ in a block. Each *codeword* is formed by eight neighboring modules $\{m_k\}_{k=1}^{8}$, $m = 1$ for the white module and $m = 0$ for the black one.

To guarantee the readability of the QR code, the maximum number of erroneous codewords must be less than half of the number of error correction codewords for each block. We denote the maximum number of erroneous codewords by ϵ.

We define ΔC as an indicator function to indicate whether a codeword C is erroneous or not:

$$\Delta C = \begin{cases} 1, & \sum_{k=1}^{8} \Delta m_k^2 > 0 \\ 0, & \sum_{k=1}^{8} \Delta m_k^2 = 0 \end{cases} \tag{5}$$

To measure the loss caused by the erroneous codewords, we formulate the loss function of QR-codes $L(Q)$ as

$$L(Q) = \begin{cases} \frac{1}{n_B} \sum_{i=1}^{n_B} (\frac{1}{\epsilon} \sum_{j=1}^{n_C} \Delta C_{ij})^2, & \forall B_i \subset Q, \sum_{j=1}^{n_C} \Delta C_{ij} \le \epsilon \\ \infty, & \exists B_i \subset Q, \sum_{j=1}^{n_C} \Delta C_{ij} > \epsilon \end{cases} \tag{6}$$

Here $L(Q)$ is in $[0, 1]$ if Q is readable and the value of $L(Q)$ indicates loss on the robustness of Q. The larger $L(Q)$ is, the less redundancy Q possesses, i.e., less robustness for Q. If $L(Q)$ goes to infinity, which means that the number of erroneous codewords exceeds ϵ in some block, then Q cannot be decoded.

To minimize the module energy E_m in Eq. (4), we adopt a gradient descent approach. For each iteration, we traverse all the black modules to compute ∇E_m. Then, we flip the black module

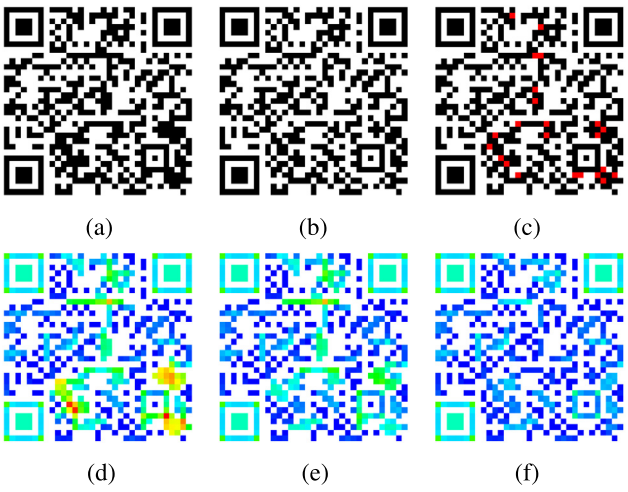

Fig. 7. An example for QR data optimization. The input QR code (a) is optimized into (b), and the flipped black modules are highlighted in red (c). Subfigures (d,e,f) show the input QR code, the QR code after 7 iterations, and the result after optimization (totally 23 iterations), respectively. The color for each black module is coded by its intensity value. Red indicates large intensity, i.e., less shadows . (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

to white according to ∇E_m, and update E_m, until $\|\nabla E_m\| < 10^{-4}$. Fig. 7 gives an example of the module optimization.

5.3. Depth optimization

Given the optimized QR-code Q, we further optimize the carving depth of the 3D QR-code, since shallower depth implies better printability, less modulations and more feasibility to the given 3D shapes. We define the target optimized QR image as U, that is robust enough for decoding. The objective of depth optimization is to achieve the target QR image U for $G(t)$, with the minimal carving depth t for each black module.

Fixing the QR modules, the condition for successful decoding is good contrast quality, which is evaluated by "symbol contrast (SC)" [28]. SC is defined as the difference of two arithmetic means of the 10% darkest pixels $\alpha = \overline{min_{0.1}(U_B)}$ and 10% lightest pixels $\beta = \overline{max_{0.1}(U_w)}$, respectively. We use γ to denote SC, i.e., $\gamma = \beta - \alpha$.

The threshold $\eta = (\alpha + \beta)/2$ is used to distinguish black modules U_B and white modules U_W. To ensure the readability, the minimal SC value is 0.2. Thus, we construct the feasible set that satisfies the above conditions:

$$U(\gamma) \subset \left\{\forall u_b \in U_B, u_w \in U_W | \gamma \geq 0.2 \cap u_b \langle \eta \cap u_w \rangle \eta\right\}. \tag{7}$$

To achieve the minimal depth, we set $\gamma = 0.2$. We extract white modules U_W from the original simulation image G_0 and fix the white modules in depth optimization. We use a simple rule to construct U_B, i.e., $U_B := \{\forall u_b \in U_B | u_b = \beta - \gamma\}$.

We represent the depth energy by E_t to measure the difference between the simulation result G and the target image U and formulate the depth optimization as:

$$\begin{aligned} &\min \ E_t(t) = (\|G(t) - U\|_F)^2 \\ &s.t. \quad 0 \leq t < \hat{t} \end{aligned} \tag{8}$$

where $\|\cdot\|_F$ refers to the Frobenius norm of the difference and $\hat{t}$ is the maximal carving depth, determined by the thickness of the input shape or user specifications.

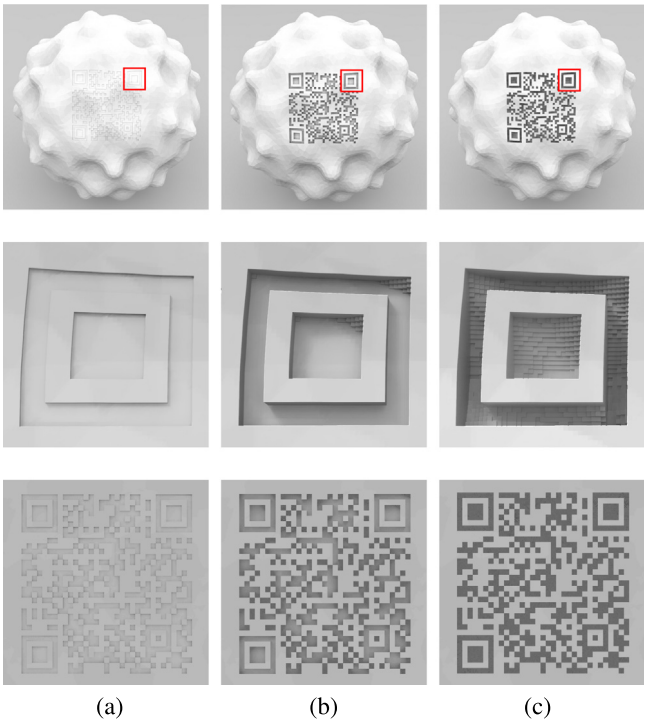

Fig. 8. The depth optimization starts from uniform carving with the minimal depth, iteratively increases the depth for each sample vertex according to the simulated results. From left to right, we show the initial state, an intermediate result and the final optimized 3D QR-code. The second row shows the zoom in of the 3D QR geometry for the region highlighted in red . (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

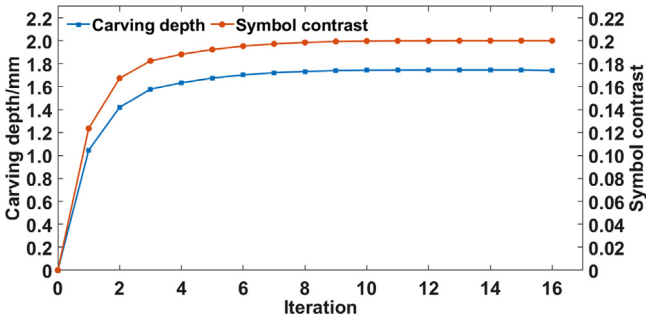

Fig. 9. Both average carving depth and symbol contrast (SC) increase along with the iterations, until the SC value reaches the readability threshold (0.2).

We employ Newton's method for optimization and use the secant method to approximate the first and second derivatives of $E_t(t)$. For each depth iteration, we update the carving depth as:

$$t^{(k+1)} = t^{(k)} - \frac{E_t'(t^{(k)})}{E_t''(t^{(k)})}. \tag{9}$$

We repeat the iteration until it satisfies the condition of convergence $\|\nabla E_t\| < 10^{-4}$, or the threshold $\hat{t}$ is reached. If the given shape is too thin to embed the QR code, the optimization returns false.

As shown in Fig. 8, the overall contrast increases along with the depth optimization iterations. We also plot the change of average carving depth and the contrast, i.e. SC value, along with iterations in Fig. 9.

Fig. 10. A 3D QR code carved on a bumpy sphere (a), its photograph image (b) and simulated image (c).

Table 1
Statistics of the results.

Model	Ver.	QRT (s)	GeoT (s)	#Iter.	TotalT (s)
Cuboid	4	37.5	117.8	14	155.3
Bumpy	4	37.6	153.3	16	190.9
Bunny	4	39.7	234.5	23	274.2
David's head	4	41.5	129.2	14	170.7
Stub	4	39.8	167.9	16	207.7
Vase_header	4	39.4	106.9	12	146.3
Vase_footer	4	39.4	118.2	13	157.6
Cup	6	/	237.8	24	237.8

6. Results

We implemented our proposed approach in C++ on an Intel® Core™ i7-6700K CPU @ 4.0 GHz and 16 GB RAM. The geometry processing library libigl [30] is used for mesh manipulation. The running time and number of iterations are listed in Table 1. 3D QR optimization performance including simulation is fast and in all our experiments stayed below 10 min. We observe that the most time-consuming step is the simulation and it directly relates to the QR resolution.

We use a Ver. 4 QR code which contains 33 × 33 modules and can embed 36–80 codewords. Each module is uniformly sampled by 5 × 5 vertex points in both simulation and carving process.

We use an FDM-based 3D printer (HORI Z500) with PLA materials to fabricate the models. The nozzle diameter is 0.4 mm. Fig. 17 shows the 3D printed QR codes on different surface geometries. Note that even for the shell shapes like Vase, bumpy surface like David's head and highly curved geometries like Bunny and Stub, our method successfully introduces 3D QR codes that are robustly decoded.

Simulation. We evaluate our simulation by comparing it with the image of the physical 3D QR code in a real environment and lighting (see Fig. 10). Specifically, for a carved 3D QR code on a bumpy sphere shape, we compute its simulated QR image and also photograph the printed 3D QR code. We compute their SC values which are 0.201 for simulation and 0.204 for the photo. This shows a high similarity between the two sources, demonstrating the sufficient accuracy of our simulation. The real photo has a slightly smaller SC value due to the noise in photographing.

Evaluation. To validate the readability of the fabricated 3D QR codes, we have tested the models in various situations. We regard a successful decoding if the QR code is decoded within 3 s using a standard decoder.

To evaluate the robustness of our 3D QR codes in various aspects, we tested the decoding success rate on several standard QR readers, in different lighting environments, from a range of scanning angles and listed the statistics in Fig. 11. The success rate (vertical axis) is recorded over 50 decoding trials. These experiments and interactions are also in the accompanying video.

Results show that the 3D QR codes have good robustness when decoded by mainstream decoders and mobile phones (Fig. 11a).

Regarding different lighting environments, the success rate is high and gets lower on the noon because the lighting is too strong, and the decoder needs more time to focus on the subject (Fig. 11b).

The chart in Fig. 11c demonstrates that if we tilt the decoder with certain angles from the original orthogonal scanning direction, the 3D QR codes possess robustness within 30 degrees, i.e., the range of the scanning angle is 60 degrees. When the scanning angle is tilted too much, say 50 degrees, the success rate decreases for both 2D and 3D QR codes. Our 3D QR codes are less robust because the sidewall of the modules introduces noises to the scanning image. We tested on the uniform carved 3D QR codes as well, which show the same robustness on the scanning angles with our optimized 3D QR, except that the success rate decreases a bit slower when the angle is larger than 30 degrees. However, the depth in the uniform carving is much larger than that of our optimized 3D QR codes.

Furthermore, we examined the relationships between the physical sizes and decoding success rates (Fig. 11d), in which the horizontal axis indicates the size for each module. The printed QR codes in different sizes are shown in Fig. 12, in which the module size is 1.8 mm, 1.5 mm, 1.2 mm, 0.9 mm, and 0.6 mm, from left to right, respectively. The success rate decreases if the module size gets small. E.g., 0.9 mm-module brings noises and increases the decoding time. If the module size is 0.6 mm, the features cannot be fully realized by our consumer-level 3D printer and thus can hardly be recognized. In the following experiments, we set the module size as 1.2 mm.

The QR codes printed in different colors and materials like plastics (FDM), powders (SLS) and resins (SLA) are shown in Fig. 13. Our structural optimization is not influenced by these

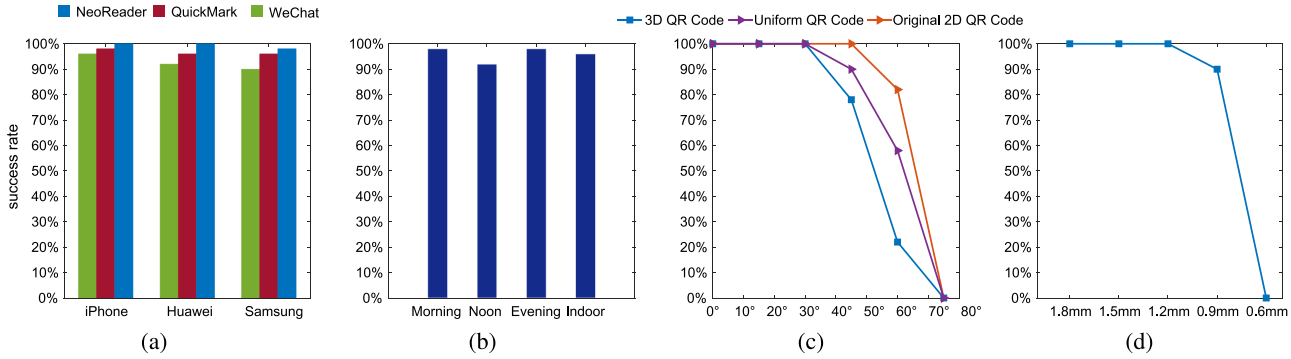

Fig. 11. Statistics on the robustness of the 3D QR codes in terms of different decoders and mobile phones (a), different lighting environments (b), different scanning angles (c), different physical sizes (d).

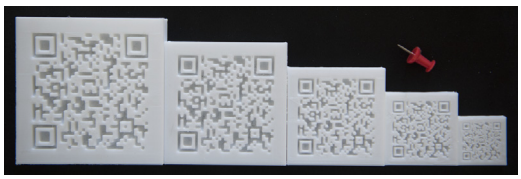

Fig. 12. A 3D QR code printed in different sizes. The length is 73.8 mm, 61.5 mm, 49.2 mm, 36.9 mm, 24.6 mm, respectively, from left to right.

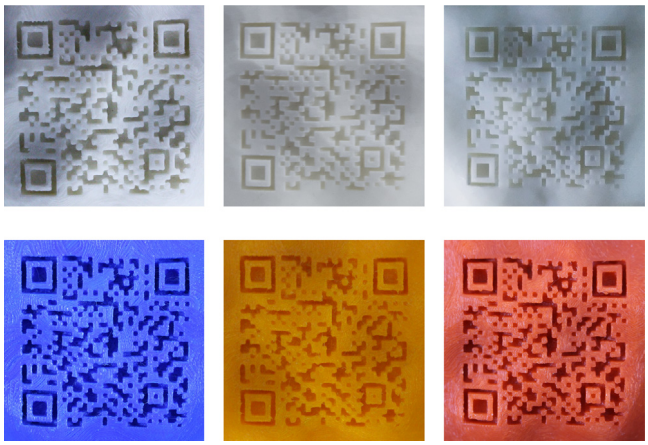

Fig. 13. A 3D QR code printed on different materials and colors. The first row lists printouts in PLA, nylon powders, and resins, from left to right, respectively. Models in the second row are printed with PLA in different colors.

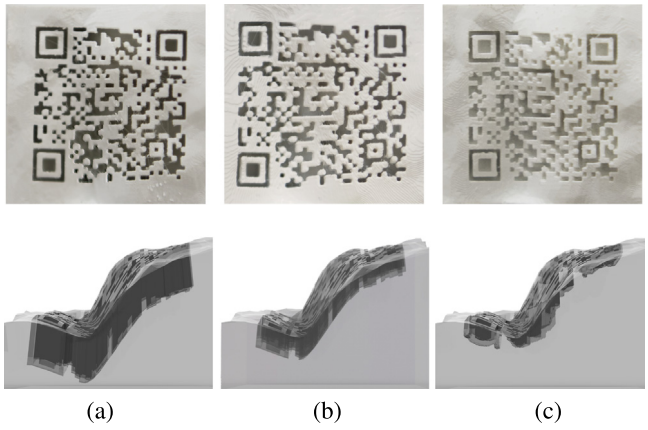

Fig. 14. Uniform carving on the initial QR image (a), uniform carving on the QR image after module optimization (b), and the optimized carved QR code (c). The average depth is 8.16 mm, 5.44 mm, and 1.86 mm for (a–c), respectively. The upper row shows photographs of the printed QR codes on the bunny model, and the lower row demonstrates the carving depth via translucent rendering from the side view.

different settings in the scene and we do not have to recompute structural optimization.

We also note that the 3D-QR-code possesses properties of the 3D printed models. It is normally stable and hard to edit, but cannot resist the damaging operations like removing the bars (white modules) or filling in some soft materials (black modules).

Comparison. To demonstrate the effectiveness of our depth optimization, we compare our optimized 3D QR structure with uniform carving. As shown in Fig. 14, our result need much less modulation on the shape than uniform carving to provide

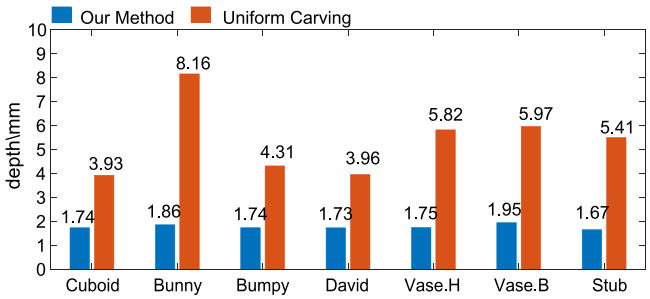

Fig. 15. Our method optimizes the carving depth, thus saving more modulations than uniform depth carving.

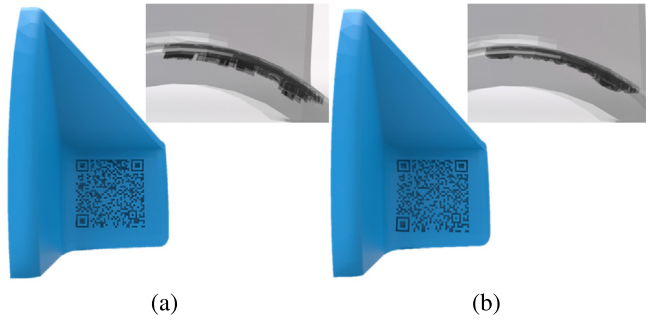

Fig. 16. Given the same QR code and the flange model, our method optimizes the module distribution and carves less on the shape (b), comparing with the method in [27] (a). The top-right translucent rendering from side view in each subfigure shows the carving depth.

enough decoding contrast. The average depth for uniform carving without module optimization is 8.16 mm while applying module optimization decreases the carving depth to 5.44 mm. After depth optimization, we can obtain the readable 3D QR code only with average carving depth 1.86 mm. The average carving depth for both methods is listed in Fig. 15.

Comparing with the method in [27], we optimize on both the module distribution and carving depth based on the guidance of the simulator and measurement on the readability robustness, and thus produce less modulation on the shape for offering robust decoding. As shown in Fig. 16, given the same QR code and the flange model, the average depth in our method is 1.52 mm, which saves 30% shape carving than the result from [27] with the average depth 2.18 mm. (The module size is 1.2 mm for both models.) The main reason is that there is no valid simulation in [27], which makes it very hard to fine optimize on the shape modulation.

7. Conclusions and futures

In this paper, we proposed a method for generating 3D QR code to make it readable and fabricable when it is printed in any curved surface using homogeneous material. There are two major components in our approach. One is the simulation based on improved ambient occlusion and physical experiments, and the other is a module optimization followed by a geometric optimization processing which enables the readability of 3D QR codes. Finally, we tested our approach by manually scanning to verify the effectiveness of our 3D QR codes on a series of models.

Limitations. Even though we propose an improved ambient occlusion algorithm to speed up each iteration, it still takes a few minutes to generate a final result. Our approach also limited by physical environment since it is sensitive to directional light.

Fig. 17. The printed 3D-QR-codes on arbitrary shapes, and the zoom-ins of the QR codes.

The support is not completely avoided is another limitation faced during fabrication using a 3D printer.

Future Work. The approach we have presented in this paper is a preliminary attempt to generate 3D QR code. Hence, there are many potential aspects that can be improved. Our current method only considers ambient light, which works well in the scenario without apparent directional light. Furthermore, a more general environment can be studied as future work. In addition, it is worthwhile to consider the effectiveness of different printed materials of different colors on the relationship between ambient occlusion and real grayscale image. Finally, our work can be potentially improved by achieving support-free and realizing real-time.

Acknowledgments

We thank all the anonymous reviewers for their valuable suggestions. This work is supported by grants from National 973 Program, China (2015CB352501), NSFC (61572291), Young Scholars Program of Shandong University (YSPSDU), China, and the Open Project Program of State Key Laboratory of Virtual Reality Technology and Systems, China, Beihang University (VRLAB2019A01). Sharf's work was partially supported by his Israel Science Foundation (ISF) and German-Israeli Foundation for Scientific Research and Development (GIF) grants.

References

[1] Visualead: Visual qr code generator, http://www.visualead.com/.
[2] FLICKR, Qr code art, http://www.flickr.com/groups/qr-art/, 2009.
[3] Cox R. Qart codes, http://research.swtch.com/qart, 2012.
[4] Chu H-K, Chang C-S, Lee R-R, Mitra NJ. Halftone QR codes. ACM Trans Graph 2013;32(6). http://dx.doi.org/10.1145/2508363.2508408, 217:1–217:8.
[5] Lin Y-S, Luo S-J, Chen B-Y. Artistic Qr code embellishment. Comput Graph Forum 2013;32(7):137–46. http://dx.doi.org/10.1111/cgf.12221, URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12221 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12221.
[6] Garateguy GJ, Arce GR, Lau DL, Villarreal OP. QR images: Optimized image embedding in qr codes. IEEE Trans Image Process 2014;23(7):2842–53. http://dx.doi.org/10.1109/TIP.2014.2321501.
[7] Lin SS, Hu MC, Lee CH, Lee TY. Efficient qr code beautification with high quality visual content. IEEE Trans Multimed 2015;17(9):1515–24. http://dx.doi.org/10.1109/TMM.2015.2437711.
[8] Chu C-H, Yang D-N, Chen M-S. Image stablization for 2d barcode in handheld devices. In: Proceedings of the 15th ACM international conference on multimedia. MM '07, New York, NY, USA: ACM; 2007, p. 697–706. http://dx.doi.org/10.1145/1291233.1291394, URL http://doi.acm.org/10.1145/1291233.1291394.

[9] Yang H, Kot AC, Jiang X. Binarization of low-quality barcode images Captured by mobile phones using local window of adaptive location and size. IEEE Trans Image Process 2012;21(1):418–25. http://dx.doi.org/10.1109/TIP.2011.2155074.

[10] van Gennip Y, Athavale P, Gilles J, Choksi R. A regularization approach to blind deblurring and denoising of qr barcodes. IEEE Trans Image Process 2015;24(9):2864–73. http://dx.doi.org/10.1109/TIP.2015.2432675.

[11] Li X, Shi Z, Guo D, He S. Reconstruct argorithm of 2d barcode for reading the QR code on cylindrical surface. In: 2013 international conference on anti-counterfeiting, security and identification (ASID). 2013, p. 1–5. http://dx.doi.org/10.1109/ICASID.2013.6825309.

[12] Lay KT, Wang LJ, Han PL, Lin YS. Rectification of images of qr codes posted on cylinders by conic segmentation. In: 2015 IEEE international conference on signal and image processing applications (ICSIPA). 2015, p. 389–93. http://dx.doi.org/10.1109/ICSIPA.2015.7412222.

[13] Mitra NJ, Pauly M. Shadow art. ACM Trans Graph 2009;28(5). http://dx.doi.org/10.1145/1618452.1618502, 156:1–156:7. URL http://doi.acm.org/10.1145/1618452.1618502.

[14] Alexa M, Matusik W. Reliefs as images. ACM Trans Graph 2010;29(4). http://dx.doi.org/10.1145/1778765.1778797, 60:1–60:7. URL http://doi.acm.org/10.1145/1778765.1778797.

[15] Alexa M, Matusik W. Images from self-occlusion. In: Proceedings of the international symposium on computational aesthetics in graphics, visualization, and imaging. CAe'11, New York, NY, USA: ACM; 2011, p. 17–24. http://dx.doi.org/10.1145/2030441.2030445, URL http://doi.acm.org/10.1145/2030441.2030445.

[16] Bermano A, Baran I, Alexa M, Matusk W. Shadowpix: Multiple images from self shadowing. Comput Graph Forum 2012;31(2pt3):593–602. http://dx.doi.org/10.1111/j.1467-8659.2012.03038.x.

[17] Wang J, Jiang C, Bompas P, Wallner J, Pottmann H. Discrete line congruences for shading and lighting. In: Proceedings of the eleventh eurographics/ACMSIGGRAPH symposium on geometry processing. Eurographics Association; 2013, p. 53–62.

[18] Schüller C, Panozzo D, Sorkine-Hornung O. Appearance-mimicking surfaces. ACM Trans Graph 2014;33(6). http://dx.doi.org/10.1145/2661229.2661267, 216:1–216:10. URL http://doi.acm.org/10.1145/2661229.2661267.

[19] Weyrich T, Deng J, Barnes C, Rusinkiewicz S, Finkelstein A. Digital bas-relief from 3d scenes. ACM Trans Graph 2007;26(3). URL http://doi.acm.org/10.1145/1276377.1276417.

[20] Sun X, Rosin PL, Martin RR, Langbein FC. Bas-relief generation using adaptive histogram equalization. IEEE Trans Vis Comput Graphics 2009;15(4):642–53. http://dx.doi.org/10.1109/TVCG.2009.21.

[21] Zhang YW, Zhou YQ, Li XL, Liu H, Zhang LL. Bas-relief generation and shape editing through gradient-based mesh deformation. IEEE Trans Vis Comput Graphics 2015;21(3):328–38. http://dx.doi.org/10.1109/TVCG.2014.2377773.

[22] Zhang Y-W, Zhang C, Wang W, Chen Y. Adaptive bas-relief generation from 3d object under illumination. Comput Graph Forum 2016;35(7):311–21. http://dx.doi.org/10.1111/cgf.13028.

[23] Zhao H, Lu L, Wei Y, Lischinski D, Sharf A, Cohen-Or D, et al. Printed perforated lampshades for continuous projective images. ACM Trans Graph 2016;35(5). http://dx.doi.org/10.1145/2907049, 154:1–154:11. URL http://doi.acm.org/10.1145/2907049.

[24] Li D, Nair AS, Nayar SK, Zheng C. Aircode: Unobtrusive physical tags for digital fabrication. In: Proceedings of the 30th annual ACM symposium on user interface software and technology. UIST'17, 2017, p. 449–60.

[25] Chen F, Luo Y, Tsoutsos NG, Maniatakos M, Shahin K, et al. Embedding tracking codes in additive manufactured parts for product authentication. Adv Energy Mater 2018;0(0):1800495. http://dx.doi.org/10.1002/adem.201800495, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/adem.201800495 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/adem.201800495.

[26] Wei C, Sun Z, Huang Y, Li L. Embedding anti-counterfeiting features in metallic components via multiple material additive manufacturing. Addit Manuf 2018;24:1–12. http://dx.doi.org/10.1016/j.addma.2018.09.003, URL http://www.sciencedirect.com/science/article/pii/S2214860418305189.

[27] Kikuchi R, Yoshikawa S, Jayaraman PK, Zheng J, Maekawa T. Embedding QR codes onto b-spline surfaces for 3d printing. Comput Aided Des 2018;102:215–23. http://dx.doi.org/10.1016/j.cad.2018.04.025, URL http://www.sciencedirect.com/science/article/pii/S0010448518302537, Special Issue on SPM 2018.

[28] ISO, ISO/IEC 18004:2015(E). Information technology - Automatic identification and data capture techniques - QR Code bar code symbology specification.

[29] Landis H. Production-ready global illumination. SIGGRAPH Course Notes 2002.

[30] Jacobson A, Panozzo D. Libigl: Prototyping geometry processing research in c++. In: SIGGRAPH Asia 2017 courses. SA'17, 2017, http://dx.doi.org/10.1145/3134472.3134497, 11:1–11:172. URL http://doi.acm.org/10.1145/3134472.3134497.