

Received July 6, 2020, accepted July 9, 2020, date of publication July 13, 2020, date of current version July 22, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3009025

# A Single-Shot, Pixel Encoded 3D Measurement Technique for Structure Light

AHSAN ELAHI<sup>1</sup>, JUN LU, QI-DAN ZHU, AND LI YONG

College of Automation, Harbin Engineering University, Harbin 150001, China

Corresponding author: Ahsan Elahi (ahsan122@gmail.com)

This work was supported in part by the Natural Science Foundation, Heilongjiang, China, under Grant F201123, and in part by the Fundamental Research Funds for the Central Universities under Grant HEU CFX41304.

**ABSTRACT** The Structure Light System (SLS) is a general concept and it is one of the cheapest methods for the non-contact-based 3D reconstruction. The existing single-shot SLS which is primarily based on the spatial encoding techniques are not optimal in terms of resolution and digitally encoded patterns. Those schemes are not flexible, controllable, and designed up to the level of the pixel. So, to increase the resolution and to implement a flexible controllable pattern we proposed a novel heuristic method based on the spatial neighborhood. In this paper, we propose a multi-resolution SLS which can be implemented with a set of 25 geometrical shaped distinct symbols or alphabets to use in the projection pattern as shape primitive. The size of each symbol is well defined in pixels which enabled us to have access and control up to the full resolution of the projector. The shape descriptive parameters for each symbol or alphabet are also defined and computed. To spread the alphabets in a controllable manner, a method is defined to generate a robust pseudo-random sequence of any required size with a certain number of alphanumeric bases, to be employed in the projection pattern concerning the measured resolution. This arrangement will enable us to design the projection patterns according to the required surface area and the resolution. A new technique is developed for the decoding of the captured image pattern. The decoding process depends upon the classification of symbols which is based on shape descriptive parameters. The searching in the neighborhood of a symbol is carried out through computing the location information, grid distance, and direction information to find the codewords which are used to establish the correspondence.

**INDEX TERMS** Structure light, stereovision, robust pseudo-random sequence, m-arrays, perfect maps, shape descriptors, grid distance.

## I. INTRODUCTION

Fast, real-time, single-shot 3D measurement has become the most challenging task and it has been widely used in industrial manufacturing, the range sensing applications, the inspection and modeling in the automation industry, reverse engineering, and medical imaging applications. In this research, we will define a complete procedure for the development of the fast, single shot, dynamic 3D vision measurement techniques based on the structured light spatial encoding projection.

Many techniques have been evolved during the past two decades for the designing of SLS. Many reviews are

The associate editor coordinating the review of this manuscript and approving it for publication was Kathiravan Srinivasan<sup>1</sup>.

available [1]–[3]. Structure light uses the principle of triangulation for the 3D measurement. In the stereo vision techniques, more than one camera was utilized to solve the correspondence problem between two or more views of the object [4]. In the structure light projection technique, one of the stereo vision cameras is replaced with a light-emitting projector [5]–[7]. Thus, the correspondence between two images is transformed into a perspective looking for corresponding points between the projected pattern and the captured image [8]. The structure light techniques can be divided into two main classes: spatial neighborhood and temporal coding [9]. Geng [10] further differentiates the temporal encoding techniques into the sequential projection techniques which include binary patterns, gray coding, phase shift, photometric, and hybrid techniques. He differentiated the spatial

projection techniques into full-frame spatially varying color patterns, Stripe Indexing (Single Shot), and Grid Indexing: 2D Spatial Grid Patterns.

Fast 3D measurement techniques have become more vital during the past few decades since it was used in the development of real-time applications. The researchers in optical metrology emphasize the techniques based on fringe patterns [11] which were used for the development of real-time 3D applications [12]. The fringe patterns have limitations of lower accuracy and resolution. Rusinkiewicz *et al.* [13], Hall-Holt and Rusinkiewicz [14] developed a real-time 3D shape measurement system based on the stripe boundary code. In their approach the image acquisition and processing time was high and four patterns were required to reconstruct one 3D model. Their technique was not a single-shot, and it was difficult to reach a pixel-level spatial resolution at high speed since the stripe width must be larger enough to cover the whole projector resolution. Consequently, the single-shot 3D measurement techniques remained the hot research area. The researchers from the computer vision proposed spatial neighborhood encoding techniques that are being single shot as well as fast, which can be used for the development of real-time and dynamic applications.

In the spatial codification techniques, the codeword of a specific location is extracted from the surrounding points. The key idea is to ensure the distinctiveness of the codeword at any location in the whole range of the pattern. The examples of spatial neighborhood are the patterns based on De Bruijn sequence [15]–[20], non-formal coding [21], and M-arrays [22]–[30].

The De Bruijn pattern was designed by Boyer and Kak [15], used in a single encoded grid of colored light stripes to measure 3D. Similarly, Hugli and Maitre [16] and Je *et al.* [17] proposed color encoded strip patterns. Likewise, Zhang *et al.* [18] use alternating color strips and a multi-pass dynamic programming algorithm, which aids in the elimination of global smoothness and strict ordering constraints. Pages *et al.* [19] proposed an optimized pattern for single-shot shape acquisition. Vuylsteke and Oosterlinck [20] proposed a single shot binary encoded pattern derived from the pseudorandom noise sequence. This approach was used for extensive feature extraction but at the cost of more time consumption. All of these techniques were not suitable for high-speed measurements, and practical for a colored object or dynamic scenes.

The two-dimensional spatial neighborhood techniques evolved from the grid encoding. Pennington and Will [31] were the first who introduce grid-coding for automatic extraction of the range data. Thus, the grid pattern combines the advantages of both the simple point and the line pattern as sharp discontinuities may indicate abrupt changes at several points on the object surface. But grid coding implies weak constrictions on the physical objects [32], since the labeling of intersecting points of the grid is time-consuming, especially if some parts of lines are occluded [27]. So, each new label is dependent on previously labeled points.

When compared to the simple coding technique such as point and shoot through laser light, as proposed by Strat and Oliveira [33], which is based on a relative labeling approach, but were not able to robustly deal with the occlusion problem. Chen *et al.* [34], [35] proposed a grid-pattern based on uniquely color-encoded codeword. The codeword at any location is defined from the color value at that location and its 4 adjacent neighbors. Since the pattern was designed with multi colors and so it was immune to noise and can be warp through the intrinsic color of the measuring surface. Our technique is employed with all the advantages of grid coding while ignoring the limitation of dependency on the labeling process of previous values since we employ a predefined labeling scheme.

Griffin *et al.* [23] used a multi-valued pseudo-random array, instead of a binary array. In his pattern, each spatial position is represented with a mini-pattern as a special codeword. He used multiple colors to illuminate the projection which has problems in the colored scene and measured surface reflections. These developments lead several researchers to use the theory of perfect maps [36] for employment in the structured-light spatial encoding schemes. The spatial encoding techniques proposed in [24]–[26] are based on the small size pseudo-random sequences which resulted in lower resolution. Petriu *et al.* [24] introduced a pseudo-random four-color encoded grid pattern composed of the rows and columns grid lines applied on a simple cubic surface. The pattern was based on small pseudo-random sequences and only 59% of feature points were detected successfully. Morano *et al.* [27] used an iterative algorithm to generate a pattern based on a pseudo-random array with  $45 \times 45$  features. Instead of symbols, he used a multi-color dot matrix and time multiplexing on/off assignment to illuminate the surface. So the approach was not a single-shot and it was difficult to read labels of each dot hence lesser feature points were deducted. Albitar *et al.* [28] proposed a pattern consist of  $27 \times 29$  features with three symbols applied for the small surface. Many authors try to increase the feature size of the pattern by augmenting M-array. Lu *et al.* [29] used an M-array with three alphabets and a feature size of  $48 \times 52$ . Xiao-Jun *et al.* [30] proposed 10 alphabets based M-array with a feature size of  $79 \times 59$  but he used similarly shaped symbols, so lesser feature points were detected. All these authors try to increase the feature points but their patterns are not completely designed up to the levels of the pixel.

Recently the Microsoft RGB-D cameras are also gaining popularity that mapped depth through either structured light or time of flight calculations [37]. The disadvantages come with the resulting shape that often misses thin geometric structures since due to lesser coarse of resolution with depth map, quantization error, and noise [38]. Most of the RGB-D camera-based SLS has a depth resolution of  $640 \times 480$  pixels. Whereas we proposed a flexible system to design your own choice of pixel resolution and it can also be implemented through RGB-D camera systems. On the contrary, the proposed method is based on its own choice of designing pattern

resolution and size calculation according to the requirements of a covered surface area.

Thus, at present no method is available which may extract more feature points, flexible and controllable enough to design the pattern at own choice of resolution. So this deficiency motivates us to propose this system to deal with the occlusion problems as well as to increase the feature size and designed pattern by utilizing the whole resolution of the projector. Our method is more suitable due to being the utilization of monochromatic light and robust symbols which combine the power of both grid-coding and predefined labeling techniques which were previously implemented with multi colors patterns.

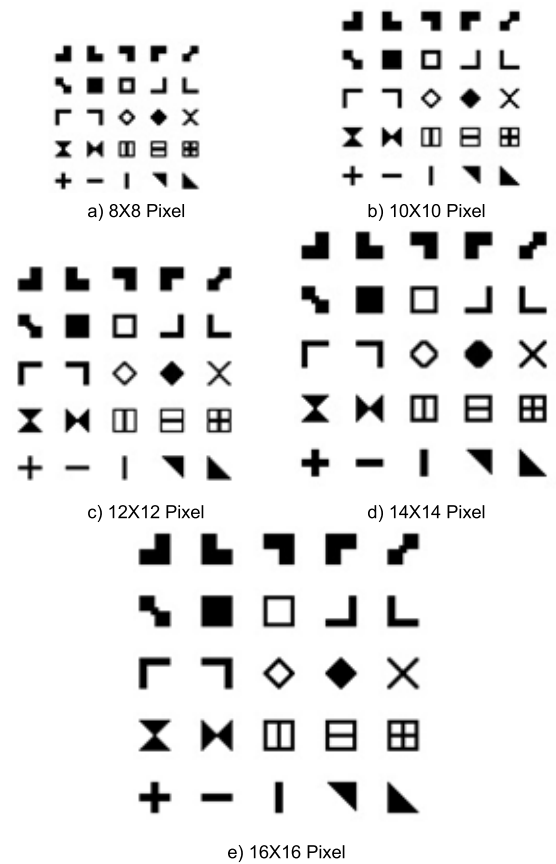
**II. ENCODING PROCESS**

In this section, the whole process of encoding the projection pattern with the proposed method will discuss. The process of encoding includes choosing symbols from the set of proposed symbols or alphabets. The chosen symbols are spread in the projection pattern by using robust pseudo-random sequences. The sizes of robust pseudo-random sequences are generated according to the size of the symbol and the dimensions of the projector, all measured in pixels. The robust pseudo-random sequences are generated through MATLAB and their robustness is ensured through validation of window property. The whole process is explained below.

**A. DESIGN OF SYMBOLS**

In this research, we have proposed a multi-resolution SLS. We have designed and proposed five sets of 25 geometric shaped symbols or alphabets. The basic requirements for the design of alphabet or symbols are their uniqueness i.e. able to be differentiable from other symbols, the properties of being easily decodable and the robustness, their characteristic to carry direction information along with the curvature of the surface and the specific size in term of pixels. So we have designed symbols while keeping in mind the above necessities for the employment as shape primitive in the design of projection pattern. Our proposed symbols are varying in sizes from  $8 \times 8$  to  $16 \times 16$  pixels. Each set of symbols corresponds to a different level of resolution on the measuring surface at a certain value of depth ( $z$ ). From these unique sets of ‘25’ symbols, one can choose few symbols (minimum 2 and maximum up to 8) to design a projection pattern of their own choice of resolution and the corresponding measuring area. Our method of using more symbols in a projection pattern will provide more flexibility and robustness in the design of a projection pattern. The five sets of proposed symbols are shown in figure 1.

To decode these symbols, we use and compute shape description parameter ratios [39], [40]. So, the classification of symbols is based on shape description parameters [40]–[42]. Since shape descriptor parameters based object understanding is more stable against sensor noise and it is more prone to illumination changes and color variation. Most of the shape descriptors are computed through regional



**FIGURE 1.** 5 sets of 25 symbols.

moments [43], [44]. Essential Shape description parameters utilized in the classification of symbols are defined in table 1. The definition of each proposed symbol and the computed values of the shape description parameters are given in table 10 in the appendix. Each symbol has unique geometric property and thus has unique shape description parameter values so they can easily decode with minimum or least chances of errors. The threshold values of these parameters will use in the classification of symbols in the decoding process. The threshold value of each parameter conveys some important information. The use of more shape description parameters will allow us to decode more symbols in the captured image pattern.

The implementation of deep-learning algorithms such as CNN [45] requires three basic elements, 1) Large-scale 3D datasets, 2) Obtainable structure and training, and 3) Graphics processing units (GPU) for the acceleration of the system. All of these three factors are essential elements for the employment of deep-learning methods. Furthermore, the challenges of forming data sets and their training make them slow, and complex processes for the computation. Our method of decoding or classification of symbols or alphabets is based on the shape description parameters, which is a simpler process when implemented since it needs low computational power when compared to the deep-learning. So the complexity of

**TABLE 1.** Shape description parameters.

Symbol	Parameter	Definition
$\epsilon$	Eccentricity	$\frac{\mu_{20} + \mu_{02} - \sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2}}{\mu_{20} + \mu_{02} + \sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2}}$
H	Number of Holes	R - Eu
ED <sub>s</sub>	Equivalent Diameter	$\frac{4A_s}{P_s}$
AR	Aspect ratio	$\frac{\text{Major Axis}}{\text{Minor Axis}}$
S	Solidity	$\frac{A_s}{A_{CH}}$
CAR	Convex to Area Ratio	$\frac{CA_s}{A_s}$
PAR	Perimeter to Area	$\frac{P_s}{A_s}$
Rect	Rectangularity	$\frac{A_s}{A_{bb}}$
Con	Convex Ratio	$\frac{P_{CH}}{P_s}$
CR	Circularity or Compactness	$\frac{A_s}{4\pi A_s}$

Where;  $\mu_{11}$ ,  $\mu_{20}$ ,  $\mu_{02}$  are the second-order moments along the center, x-axis, and y-axis respectively.  $A_s$ , Eu, R,  $P_s$ ,  $CA_s$ ,  $A_{cir}$ , are the area in pixels, Euler's number, number of connected regions, parameter, convex area, and area of the circle with the same parameter.  $A_{CH}$ ,  $P_{CH}$ ,  $A_{bb}$  are the area of the convex hull, the parameter of the convex hull, and the area of the minimum bounding rectangle that enclose the symbol.

the algorithm and the hardware requirements will reduce which will be a significant advantage, and thus the decoding performs well.

## B. GENERATION OF RPRS

Pseudo-random sequences are widely used in many applications [46], [47]. The pseudo-random sequences with more alphabets or symbols or alphanumeric basis make them suitable to use with the large dimensions and provide flexibility to the user to generate a sequence of their desirable dimensions. We employed the Mersenne twister method to generate the pseudo-random sequences [48], [49]. The Mersenne twister has the advantages of being fast, high equal distribution, and a very long period as compared to the shift register method [50], [51]. These properties of Mersenne twister provide principal advantages to the designer to generate very high-quality pseudo-random numbers. Thus larger dimension perfect maps can be formed with a lesser number of alphanumeric bases. The theory of perfect maps such as M-arrays [36] is already employed in the SLS [28]–[30]. We also generated a robust pseudo-random sequence (RPRS) by using the theory of perfect maps. In perfect maps, no codewords will repeat itself and each codeword and its location is unique in the sequence. So the coarse correspondence can be established easily. The RPRS with a specific size is the requirement for the proposed system to spread the symbols in a controllable manner to form the projection patterns. First, we will generate a raw pseudo-random sequence. The raw pseudo-random sequences are converted to robust pseudo-random sequence

by validating their window property. In our method, we utilize the window property of  $3 \times 3$  to check the robustness of the sequence.

The use of a more alphanumeric basis in the pseudo-random sequence provides more robustness since the chances of errors are reduced. The increase in alphabets will increase the robustness and the Hamming distances between the codewords. The repetition of codewords will not occur if the pseudo-random sequence is generated with more symbols or alphabets or alphanumeric bases. So there is a direct relationship between the robustness, the Hamming distances, and the number of alphanumeric bases of the pseudo-random sequence. The robustness of each pseudo-random sequence will ensure through calculating the Hamming distances between the codewords of each independent window of size  $3 \times 3$ . Hamming distances are the ability of the robustness. Since the codewords in each window are based on an alphanumeric basis. So, the difference at any location of two windows while comparing will increase the Hamming distance. Thus, Hamming distances are the measurement of differences in between the codewords of length  $3 \times 3$  of two independent windows. As the alphanumeric bases are increased in the pseudo-random sequences, the range of alphabets used in the codewords will also increase, so the Hamming distances or robustness is improved. The codewords are considered to be robust if the Hamming distances between them are greater than or equal to '3' for the window property of size ' $3 \times 3$ '.

### 1) SIZE CALCULATION OF RPRS

The first step in the generation of a robust pseudo-random sequence is to find the required dimension of the sequence. The size of the pseudo-random sequence is derived from the size of the projector resolution, size of the symbol, and the spacing between two consecutive symbols, which are all defined in pixels. As we proposed a multi-resolution system that depends upon symbol sizes varies from  $8 \times 8$  to  $16 \times 16$  pixels. The smaller symbol size will lead to a larger size of pseudo-random sequences and vice versa as inferred from the equation (1) and (2). The smaller symbol size and spacing will lead to more shape primitive in the same area and thus result in higher measuring resolution. Since more feature points or alphanumeric basis are required to fill the same resolution of the projector. The X and Y dimensions of the desired size of the pseudo-random sequence can be calculated as:

$$X_{RPRS} = \frac{X_{PP}}{S_z + P_s} \quad (1)$$

$$Y_{RPRS} = \frac{Y_{PP}}{S_z + P_s} \quad (2)$$

where;  $X_{PP}$ ,  $Y_{PP}$ ,  $X_{RPRS}$ ,  $Y_{RPRS}$ ,  $S_z$ , and  $P_s$ , are the X & Y dimensions of projector resolution, dimensions of RPRS, Symbol Size, and pixel spacing respectfully.

The calculated X and Y dimensions are rounded up to the next nearest integer and further increase up to the nearest multiple of three to get each dimension divisible by 3. This is done to ensure the validation of the independent window property.

TABLE 2. Dimension calculations for RPRS.

Symbol Size	Spacing	Alphabets	Required Dimensions (m X n)	RPRS Dimensions (m X n)
8X8	1	5	89 X 143	90 X 144
10X10	1	4	73 X 117	75 X 117
12X12	1	4	62 X 99	63 X 99
14X14	2	4	50 X 80	51 X 81
16X16	2	3	45 X 72	45 X 72

Note: we use a projector resolution of 800 X 1280 pixels.

Table 2 summarizes the calculation of ‘X’ and ‘Y’ dimensions of robust pseudo-random sequence (RPRS) for different symbols size, the spacing between consecutive symbols, and the number of alphabets or symbols or alphanumeric bases used.

2) GENERATION OF RPRS

To generate the pseudo-random sequence of required dimensions with a specific number of alphanumeric bases we used the Mersenne twister generator engine available in the MATLAB programming language to implement our algorithm. However, we need to ensure the robustness of each raw pseudo-random sequence by validating the window property. For the purpose, we compared each independent window of size 3 × 3 with any other independent window in raw PRS. If there might not find the same window than pseudo-random sequence (PRS) is verified to be the robust pseudo-random sequence (RPRS). During the process of comparison if there may found a similar window than that raw pseudo-random sequence will discard. The whole process is repeated until a robust pseudo-random sequence (RPRS) will obtain. Once the robust pseudo-random sequence (RPRS) is obtained it has been stored in the memory for later use in the formation of a projection pattern. The flowchart in figure 2 shows the whole process of the generation of robust pseudo-random sequence (RPRS).

3) COMPUTATIONAL REQUIREMENTS FOR RPRS

The total number of independent windows and the total number of comparisons required for the desire robust pseudo-random sequence of size (m X n) using window property of (r X v) can be estimated as:

$$N_w = \frac{\text{Dimensions of PRS}(m \times n)}{\text{window size}(r \times v)} \tag{3}$$

$$\begin{aligned} \text{Total comparison required} \\ = \frac{N_w \times (N_w - 1)}{2} \end{aligned} \tag{4}$$

where; Nw is the total number of independent windows in PRS.

Table 3 summarizes the calculation of independent windows, average Hamming distance, robust codeword, and comparison required for each robust pseudo-random sequence (RPRS). Figure 3 represents the Hamming distance profiles and the percentile of their codewords for each robust

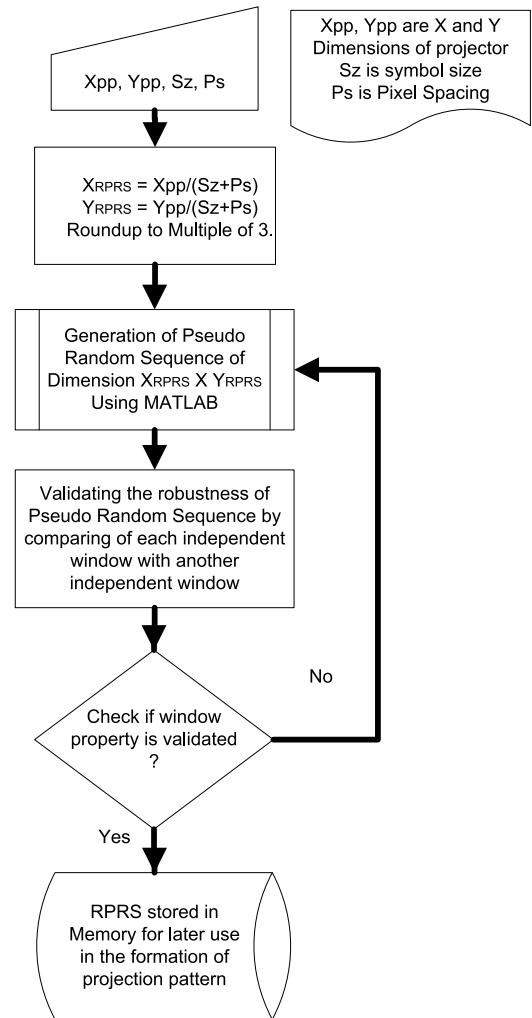


FIGURE 2. Process of generating RPRS.

TABLE 3. Robustness calculation of each RPRS.

RPRS Dimensions (m X n)	Average Hamming Distance	Robust Codeword (%)	Independent windows	Comparisons
90 X 144	7.2013	99.9699	1440	1,036,080
90 X 144	6.7486	99.8645	1440	1,036,080
75 X 117	6.7474	99.8688	975	474,825
63 X 99	6.7493	99.8565	693	239,778
51 X 81	6.7478	99.8649	459	105,111
45 X 72	5.9987	99.1767	360	64,620

pseudo-random sequence (RPRS) generated while using our method.

C. FORMATION OF PROJECTION PATTERN

A considerable number of projection patterns can be formed by using the proposed symbols and the robust pseudo-random sequences generated. The robust pseudo-random sequences consist of an alphanumeric basis which is represented with the symbols or alphabets in the projection pattern. The size

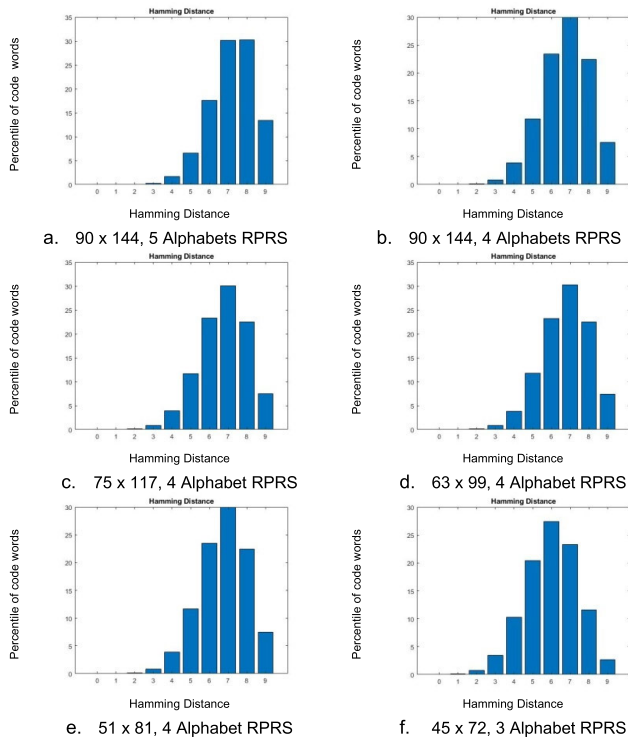


FIGURE 3. Hamming distance distribution for each RPRS.

of each projection pattern will be calculated by rearranging the equation no (1) and (2) as:

$$X_{PP} = X_{RPRS} \cdot (S_z + P_s) \tag{5}$$

$$Y_{PP} = Y_{RPRS} \cdot (S_z + P_s) \tag{6}$$

The size of the projection pattern obtained by using different symbol sizes and the corresponding robust pseudo-random sequence are slightly greater. The sizes of the projection patterns obtained are shown in table 4.

TABLE 4. Computation for the dimension of the projection pattern.

Symbol Size	RPRS Dimensions (m X n)	Alphanumeric Symbols	Spacing	Pattern Dimensions (M X N)
8X8	90 X 144	5	1	812 X 1,298
	90 X 144	4	1	812 X 1,298
10X10	75 X 117	4	1	825 X 1,287
12X12	63 X 99	4	1	819 X 1,287
14X14	51 X 81	4	2	816 X 1,296
16X16	45 X 72	3	2	810 X 1,296

The extra pixels greater than  $800 \times 1280$  will cut from any two sides. Figure 4, shows the texture and portion of six projection patterns obtained while applying with six different RPRS as mentioned in table 4. Each RPRS is generated for different symbol sizes and spacing. So each projection pattern will cover the same area but with a different number of shape primitives and therefore different density. For example, the projection patterns designed with large symbol

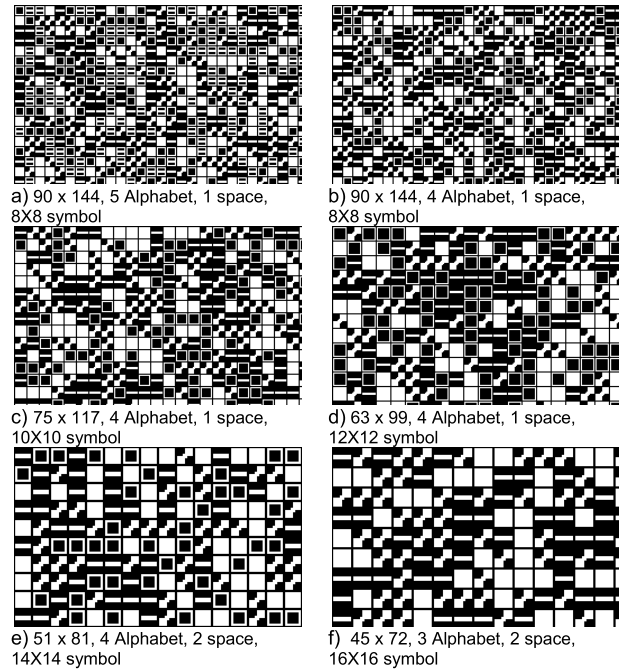


FIGURE 4. Projection patterns for different symbol sizes applying with different RPRS and spacing corresponding to different resolutions.

size ( $16 \times 16$ ) have lesser primitives; while on the other hand, the projection patterns designed with small symbol size ( $8 \times 8$ ) have more primitives. It is important to highlight that the similarly shaped symbols may not use in the same projection pattern. Another very useful property is direction information since every symbol will follow the texture of the surface, so the direction is utilized in the searching of neighborhood symbols for decoding the projection pattern. So the projection pattern must consist of at least one symbol which must carry direction so that the other directionless neighborhood symbols may acquire their direction or orientation from that symbol. If more symbols in the projection pattern may carry direction than it will make the decoding process more efficient and simpler.

#### D. PROJECTOR AND CAMERA CALIBRATION

Calibration is a critical and necessary step for the projector-camera-based SLS. In the calibration process of SLS, the camera-projector devices have to be calibrated to optimize the parameters for the minimization of re-projection errors. The projector can also be seen as the backlight camera path. So, it also requires calibration of intrinsic and extrinsic parameters just like the camera. Our system is first calibrated using the traditional calibration method to get the primary calibration parameters. A reference plane with some precisely printed markers is used for the optimization of primary calibration parameters since the traditional calibration methods rely mostly on the standard reference or the corresponding image model. Thus, before applying the projection patterns over to the measuring surface, the projector and the camera

have to be calibrated with any of the techniques available in [52]–[55].

### III. DECODING PROCESS

In this section, the whole process of decoding will be explained after applying the encoded pattern on the measuring surface. The captured image pattern will first undergo through image contrast enhancement and noise removal, so that image thresholding will carry out to get a binary image. The binary image is used to generate a 3D point cloud of measuring surface. The symbols or the alphabets which are originally spread through a pseudo-random sequence in the binary image are then labeled as specific region number. All the detected regions in the binary image are then classified using the threshold values in table 10 in the appendix. After the classification of each symbol, the location of each symbol is determined by calculating the centroid positions. The direction information and the grid distance of each symbol will also compute. After determining the centroid positions, the direction information, and the grid distances for each symbol, the process of neighborhood searching and decoding of  $3 \times 3$  codewords of the robust pseudo-random sequence (RPRS) will carry out to establish the correspondence between projected and captured image patterns. Finally, the principle of triangulation is used to reconstruct the 3D of the measuring surface. The whole process of decoding is summarized in the flow chart shown in figure 5.

#### A. PREPROCESSING

The first step of decoding is image preprocessing i.e. to prepare the captured image for decoding. In this step, the captured pattern of an image which is obtained from the measuring surface first undergoes the image contrast enhancement and noise removal through filtering, and so the image segmentation is performed to obtain a binary image. The segmentation has a significant impact on the decoding process. As if the segmentation process is weaker than the neighborhood symbols will merge and if it is stronger than so many binary regions may be deleted which can be decoded as symbols. So balance is required in the segmentation process. The best results can be obtained through optimum global thresholding using Otsu's method which is used to perform clustering-based image thresholding [56].

#### B. LABELING

The binary regions obtained through segmentation are labeled with specific region numbers through the employment of the algorithm specified by Haralick and Shapiro [57].

#### C. SHAPE DESCRIPTOR PARAMETERS

After labeling each binary region in the captured image, the shape description parameters as described in table 2 are calculated for each binary region to classify these regions into the specific symbol or alphabet.

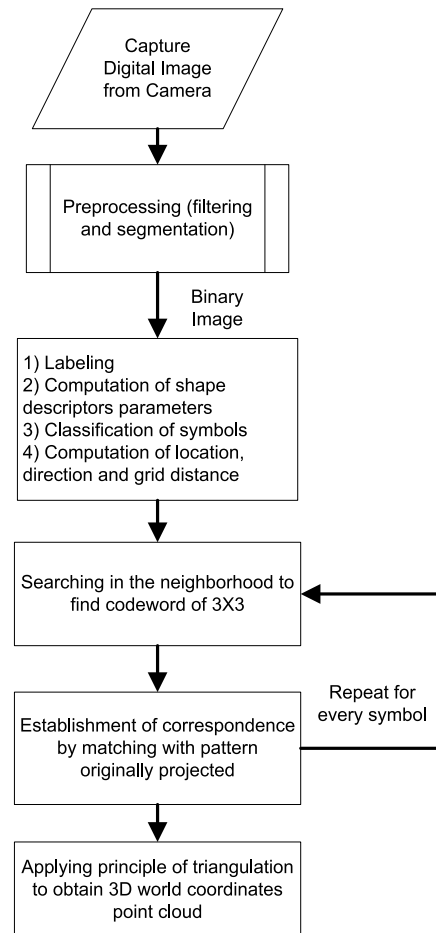


FIGURE 5. Decoding process.

#### D. CLASSIFICATION OF SYMBOLS

After the labeling and the computation of the shape descriptor parameters of each binary region in the captured image pattern, the classification of symbols or alphabets is carried based on the shape description parameters or ratios, by comparing with their earlier computed threshold values in table 10. The utilization of more shape descriptor parameters will reduce the chances of errors in the process of classification. Robust classification is obtained by utilization of more number of shape description parameters than alphabets or symbols.

#### E. COMPUTATION OF LOCATION, DIRECTION & GRID DISTANCE

In this research, we employed a new technique to search in the neighborhood of a symbol. The technique is based on the computation of grid distance, centroid position, and orientation. The neighborhood searching is carried out after the classification of each symbol and the identification of the centroid positions. The neighborhood searching is carried out to determine the location of each window of RPRS to establish the correspondence between the projected pattern and

captured image. So, before establishing the correspondence these parameters will be determined.

1) CENTROID

After the classification of each symbol, the location is determined through the computation of the centroid or center of gravity. So, the location of each symbol is determined and the corresponding label number has been assigned. This location information i.e. centroid position is utilized in the searching of the neighborhood symbols. The centroid position or center of gravity will be determined through the following formulation:

$$\text{Centroid } (\bar{x}, \bar{y}) = \frac{1}{N} \left( \sum_1^N x_i, \sum_1^N y_i \right) \quad (7)$$

or

$$\text{Centroid } (\bar{x}, \bar{y}) = \left( \frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \quad (8)$$

where; N, i, m<sub>00</sub>, m<sub>10</sub>, and m<sub>01</sub> are the total number of pixels, ith position of a pixel in a symbol, zero-order moments along the center, x-axis, and y-axis respectively.

2) DIRECTION OR ORIENTATION

The direction information is necessary to find the neighborhood symbols or alphabet in the projection pattern. When a symbol or alphabet falls on the measuring surface it will follow the curvature. So each neighborhood symbol is determined inline or in the direction of the previous symbol. If the alphabets or symbol carry the direction as their inheritance property then it will make the process of computation simpler. The direction of each symbol can be computed from the following equation:

$$\text{Orientation } (\theta) = \frac{1}{2} \arctan\left(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}}\right) \quad (9)$$

where; μ<sub>11</sub>, μ<sub>20</sub>, μ<sub>02</sub> are the second-order moments along the center, x-axis, and y-axis respectively.

So it is necessary to compute the direction for all those alphabets or symbols which do not possess it initially as inherited property. The simplest way is to acquire from the neighborhood. So, it is attained from the symbol which is present at the close range. The distance between two neighborhood symbols ‘d<sub>min</sub>’ can be computed using Euclidean distance formula between the centroid positions of two neighborhood symbols. This distance must be within close range or proximity, ‘R<sub>C</sub>’. The formulation is shown below:

$$d_{\min} = \sqrt{(x_o - x_n)^2 + (y_o - y_n)^2} \quad (10)$$

$$d_{\min} < R_C$$

where; (x<sub>o</sub>, y<sub>o</sub>) is the centroid of a symbol that does not possess direction initially. (x<sub>n</sub>, y<sub>n</sub>) is the centroid of the neighborhood with inherit direction property. Note: The value of the close range, ‘R<sub>C</sub>’ is selected in such a way that the acquired direction is from the closest alphabet and it is usually 2.5 times of grid distance.

3) GRID DISTANCE

The computation of grid distance follows the computation of direction for each symbol. The grid distance can be defined as the smallest distance between the centroid positions of the two consecutive neighborhood symbols. The grid distance is varied with the surface orientation. The grid distance can be computed through either direct computation from the equivalent diameter of a symbol or acquiring from the nearest square symbol as they utilize the maximum area in the symbol space, so they have a maximum equivalent diameter. To find the nearest square object of an alphabet the procedure of searching in the close range is utilized. Hence grid distance can be defined as equivalent diameter cumulative with pixel space between two consecutive symbols. Therefore it can mathematically express as:

$$\text{grid} = \frac{ED_s}{\sqrt{FAR}} + S_{\text{pix}} \quad (11)$$

$$\text{or grid} = ED_{ns} + S_{\text{pix}} \quad (12)$$

where; the grid is the grid distance between two neighborhood symbols. ED<sub>s</sub> is the equivalent diameter of a symbol. ED<sub>ns</sub> is the equivalent diameter of the nearest Square symbol. S<sub>pix</sub> is pixel space. FAR is the filled area ratio.

The estimated grid distances for different sizes alphabets and pixel spacing are shown in table 5.

TABLE 5. Grid distances calculation.

Symbol Size	Spacing	Grid distance
8X8	1	10.027
10X10	1	12.2838
12X12	1	14.5406
14X14	2	16.7973
16X16	2	20.0541

F. SEARCHING IN THE NEIGHBORHOOD

The searching in the neighborhood means to find the next neighborhood and consecutive of symbol and their next alphabet in both vertical and horizontal direction, to find out the codewords having windows size of 3 × 3. This will utilize in the establishment of correspondence between the projected pattern and the captured image. The necessary parameters required for searching in the neighborhood are the centroid positions, direction, and grid distances which were computed earlier. The next neighborhood and consecutive alphabet are determined by the estimation of the location (centroid position) of that symbol. The estimated centroid positions of neighborhood symbols are calculated by using the current centroid positions, direction, and grid distances. Simple trigonometric rules can be applied to calculate the expected centroid positions of right and downside neighborhood symbols. The geometrical concept for the calculation is explained in figure 6. The whole process is mathematically described as:

$$x_{\text{mb}} = x_{\text{alp}} + \text{grid} * \cos \theta \quad (13)$$



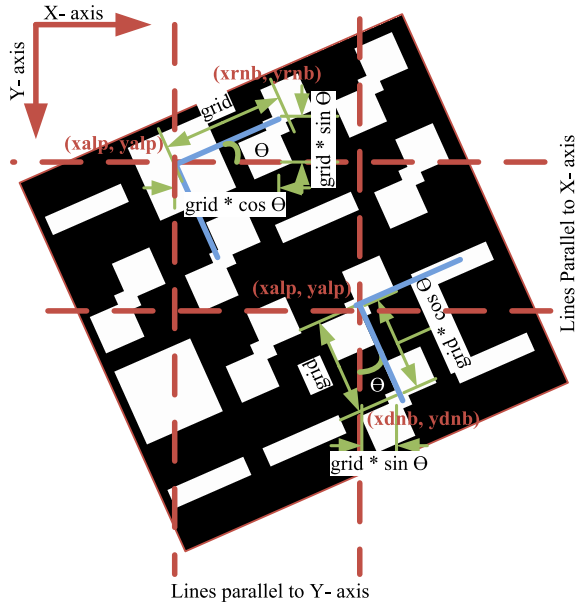


FIGURE 6. Searching in the neighborhood.

$$y_{rn} = y_{alp} - \text{grid} * \sin \theta \tag{14}$$

$$x_{dn} = x_{alp} + \text{grid} * \sin \theta \tag{15}$$

$$y_{dn} = y_{alp} + \text{grid} * \cos \theta \tag{16}$$

where;  $(x_{alp}, y_{alp})$ ,  $(x_{rn}, y_{rn})$  and  $(x_{dn}, y_{dn})$  is the centroid position of the current alphabet, right and downside neighborhood respectively.  $\text{grid}$  is estimated grid distance between two neighborhood alphabets. ‘ $\theta$ ’ is the direction of the current alphabet whose neighborhood is to be determined.

Since the grid distance between the neighborhood symbols varies with the surface orientation and the curvature. So the estimated centroid positions may not be the exact centroid positions. The actual grid distance will determine after reading the label value at the position of estimated centroid positions. The label value will identify that symbol and then actual centroid will retrieve from the memory. After knowing the actual centroid position of neighborhood symbol than actual grid distance between two neighborhood symbols will be calculated. The Euclidean distance formula is used to calculate the actual grid distance.

The initial centroid position was known so the actual grid distance for the right and downside neighborhood is determined as follows:

$$\text{grid}_{rn} = \sqrt{(x_{rn} - x_{alp})^2 + (y_{rn} - y_{alp})^2} \tag{17}$$

$$\text{grid}_{dn} = \sqrt{(x_{dn} - x_{alp})^2 + (y_{dn} - y_{alp})^2} \tag{18}$$

where;  $(x_{rn}, y_{rn})$ ,  $(x_{dn}, y_{dn})$  are the actual centroid position of the alphabet when searching towards the right and downside of the neighborhood respectively.

If the label value found in the right or downside neighborhood appears to be zero due to any reason, then searching is made around the estimated centroid position along the

diagonal at 45 degrees angle along the upper and lower sides for a distance started from a pixel up to half grid distance. When an alphabet is detected within this range along the diagonal than the process of searching is completed. The calculation for searching along the diagonal from the estimated neighborhood is expressed as:

$$x_{ud} = x_{nb} + d_i \cos \left( \frac{\pi}{4} \right) \tag{19}$$

$$y_{ud} = y_{nb} - d_i \sin \left( \frac{\pi}{4} \right) \tag{20}$$

$$x_{ld} = x_{nb} - d_i \cos \left( \frac{\pi}{4} \right) \tag{21}$$

$$y_{ld} = y_{nb} + d_i \sin \left( \frac{\pi}{4} \right) \tag{22}$$

where;  $(x_{nb}, y_{nb})$  is an estimated centroid position for neighborhood symbol.  $(x_{ud}, y_{ud})$  and  $(x_{ld}, y_{ld})$  are upper and lower side searching positions calculated along diagonal at 45 degrees from the estimated centroid position.  $d_i$  is the iterative distance measured in pixels varies from 1, 2, . . . . to  $\frac{\text{grid}}{2}$  (half grid distance).

G. ESTABLISHMENT OF CORRESPONDENCE

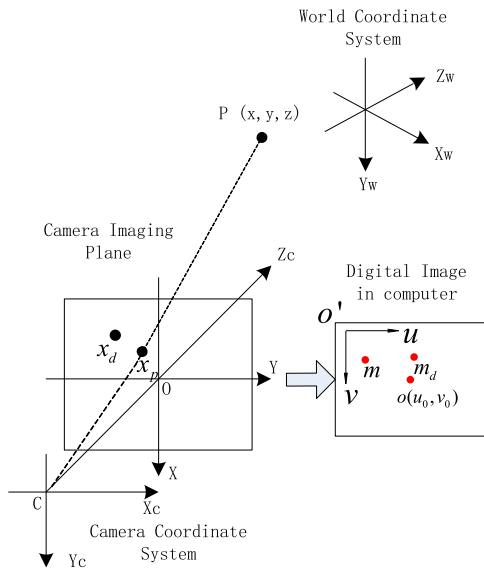
Once the neighborhood of an alphabet or symbol has been identified the procedure for searching in the neighborhood is repeated for every next symbol to find it’s right and downside neighborhood to obtain a codeword of a window having size  $3 \times 3$ . This window is used to establish the correspondence between the pattern originally projected and the pattern obtained from the captured image. When a matching window of a codeword with size  $3 \times 3$  alphabets or symbols is found in the captured image pattern it is used to establish correspondence with the pattern originally projected, by finding a similar window. Since the location of each symbol is known in the captured image and projected patterns. So this location information of each symbol with its deviation from the original pattern is utilized in the measurement of 3D.

H. RECONSTRUCTION OF 3D

The purpose of decoding is to establish the correspondence between projected and captured image patterns from the measuring surface. The need of establishing correspondence is to find a grid of matching points in the projection pattern and texture of points obtained from the measuring surface so that principle of triangulation may be applied to reconstruct or obtain the 3D shape. In this sub-section, we will define the whole procedure adopted for the reconstruction of 3D, from the corresponding points obtained by matching the related points in between the projected and the captured image patterns on measuring surface through window-based operation as described in the previous section.

1) LINEAR CAMERA MODEL [58], [59]

The ideal linear camera model is shown in figure 7. Here, any point ‘P’ in the world coordinate space is observed on the imaging plane from the optical center ‘C’ of a camera through the intersection at a point, ‘ $x_p$ ’ whereas; ‘m’



**FIGURE 7.** Ideal linear camera optical model and corresponding digital image.

is the corresponding computer screen image. As evident in figure 7, the following coordinate system can be observed and described here:

*a: WORLD COORDINATE SYSTEMS*

The spatial coordinates of any point P in the world coordinate system can be written as:  $(X_w, Y_w, Z_w, 1)^T$

*b: CAMERA COORDINATE SYSTEM*

The camera is used to describe the position of any object in the space and its surrounding environment. The Camera coordinate system can be expressed as:  $(X_c, Y_c, Z_c, C)$ .

Where; the point ‘C’ being the optical center of the camera coordinate system,  $Z_c$  is the optical axis, XOY is the camera imaging plane which is parallel to the  $X_c Y_c$  plane.

*c: PIXEL COORDINATES*

The camera image plane is transformed into a digital image, ‘m’ on the computer. In figure 7;  $u_0 v_0$  is the computer screen coordinates axes formed from the camera imaging plane. The origin is  $o'$  which is at the upper left corner of the screen. The points in the digital images are represented with pixels.

**2) COORDINATE TRANSFORMATION**

The point ‘P’ in the world coordinates system lies in space have a coordinate position of  $(X_w, Y_w, Z_w, 1)^T$ , is converted to two-dimensional digital image ‘m’, with the pixel coordinates of  $(u, v, 1)^T$ , through coordinate transformations. First, the world coordinate system is transformed to the camera coordinate system by the following relationship;

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = M \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (23)$$

where; R is  $3 \times 3$  rotational matrix, T is  $3 \times 1$  translational matrix, M is outside camera parameter matrix. R & T are outside camera calibration parameters.

The perspective camera view is then normalized into the digital image by using simple geometry and the pinhole camera model. So, the following equation can be obtained:

$$\begin{bmatrix} u_n \\ v_n \end{bmatrix} = \begin{bmatrix} X_c \\ Z_c \\ Y_c \\ Z_c \end{bmatrix} \quad (24)$$

where  $(u_n, v_n)$  are X - Y coordinates formed by image-capturing plane of the camera,  $(X_c, Y_c, Z_c)$  are the camera coordinates system.

The camera lens distortion can be accommodated by including radial distortion and tangential distortion, which can be expressed as:

$$m_d = \begin{bmatrix} u_d \\ v_d \end{bmatrix} = \begin{bmatrix} 1 + k_1 r^2 + k_2 r^4 \end{bmatrix} \begin{bmatrix} u_n \\ v_n \end{bmatrix} + dx \quad (25)$$

where;  $(u_d, v_d)$  are the coordinates of the image plane in the camera after accommodating the effects of lens distortion.

The first term represents radial distortion; here in the first term  $k_1, k_2$  are the radial distortion parameters. The second term represents tangential distortion; here in second term dx is the tangential distortion and it is defined as:

$$dx = \begin{bmatrix} 2p_1 u_n v_n + p_2 (r^2 + 2u_n^2) \\ p_1 (r^2 + 2v_n^2) + 2p_2 u_n v_n \end{bmatrix} \quad (26)$$

where; the parameters,  $p_1, p_2$  are tangential distortion. While ‘r’ can be defined as:

$$r^2 = u_n^2 + v_n^2 \quad (27)$$

Finally, the transformation from digital image coordinates stored in the computer to the coordinates of an image capturing plane in the camera after accommodating the effects of lens distortion is given by:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & \gamma\beta & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_d \\ v_d \\ 1 \end{bmatrix} = K \begin{bmatrix} u_d \\ v_d \\ 1 \end{bmatrix} \quad (28)$$

where;  $(u_0, v_0)$  are the exact Position coordinates of center point ‘o’ in computer image,  $\alpha, \beta$  represent scale factor for u-Axis and v-Axis in the digital image. K is the parameter matrix inside the camera unit.

**3) 3D MODEL**

The establishment of correspondence may lead to the grid of matching points in between the captured image from the measuring surface and the projected pattern. So we know the grid points of the captured image pixel coordinates, can be represented with  $(u_1, v_1, 1)^T$ , and similarly, the projected image pixel coordinates can be represented with  $(u_2, v_2, 1)^T$ . So with the corresponding relationship between these two coordinates, we can reconstruct the pixel coordinates of the

world coordinate system  $(X_W, Y_W, Z_W, 1)$ . Equation (24) can be rewritten in the form of matrix multiplication as follows:

$$Z_{C1} \begin{bmatrix} u_{n1} \\ v_{n1} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_{C1} \\ Y_{C1} \\ Z_{C1} \\ 1 \end{bmatrix} \quad (29)$$

Usually, we do not need to consider the tangential lens distortion since now a day the camera lens distortion effects are overcome and accommodate during camera manufacturing; therefore, the equation (25) and (26) will be simplified as follows:

$$\begin{bmatrix} u_{d1} \\ v_{d1} \\ 1 \end{bmatrix} = \left[ 1 + k_1 r^2 + k_2 r^4 \right] \begin{bmatrix} u_{n1} \\ v_{n1} \\ 1 \end{bmatrix} \quad (30)$$

Finally, the equations (23), (28), (29) and (30) defines the relationship between the coordinate points of the world coordinate in space and the pixel coordinates of captured image:

$$Z_{C1} \begin{bmatrix} u_{n1} \\ v_{n1} \\ 1 \end{bmatrix} = \left[ 1 + k_1 r^2 + k_2 r^4 \right] K \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{bmatrix} \quad (31)$$

Eventually, it can be written as follows:

$$Z_{C1} \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = M \begin{bmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{bmatrix} \quad (32)$$

Similarly, we can find the relationship between the world coordinate system and the coordinates system of the projected pattern with a point to point transformation as follows:

$$Z_{C2} \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix} = N \begin{bmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{bmatrix} = \begin{bmatrix} n_{11} & n_{12} & n_{13} & n_{14} \\ n_{21} & n_{22} & n_{23} & n_{24} \\ n_{31} & n_{32} & n_{33} & n_{34} \end{bmatrix} \begin{bmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{bmatrix} \quad (33)$$

where;  $Z_{C1}$  and  $Z_{C2}$  are the coordinates of point 'P' in the camera coordinates system and an optical axis of the projector coordinate system respectively. ' $m_{ij}$ ' is the  $i$ th row and the  $j$ th column of the matrix 'M', while ' $n_{ij}$ ' is the  $i$ th row and the  $j$ th column of matrix 'N' respectively.

From the above two equations (32) and (33) with the elimination of ' $Z_{C1}$ ' and ' $Z_{C2}$ ' we can obtain world coordinate i.e.  $[X_W, Y_W, Z_W]^T$ . The system of equations with three elements of world coordinates is as shown in matrix form:

$$\begin{bmatrix} (u_1 m_{31} - m_{11}) & (u_1 m_{32} - m_{12}) & (u_1 m_{33} - m_{13}) \\ (v_1 m_{31} - m_{21}) & (v_1 m_{32} - m_{22}) & (v_1 m_{33} - m_{23}) \\ (u_1 n_{31} - n_{11}) & (u_1 n_{32} - n_{12}) & (u_1 n_{33} - n_{13}) \\ (v_1 n_{31} - n_{21}) & (v_1 n_{32} - n_{22}) & (v_1 n_{33} - n_{23}) \end{bmatrix} \times \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} = \begin{bmatrix} m_{14} - u_1 m_{34} \\ m_{24} - v_1 m_{34} \\ n_{14} - u_1 n_{34} \\ n_{24} - v_1 n_{34} \end{bmatrix} \quad (34)$$

In algebraic form it can be represented as (35), as shown at the bottom of the page. Thus, the above system of equations (34) or (35) is constrained of a linear system, which is composed of four equations with three unknowns. A theoretical unique solution can be obtained directly. However, practically the extracted data applications may contain noise. Therefore, using the least square method to solve for the world coordinates of point P is equivalent to the sum of squares of the minimum distance for two rays emitted by the camera. So, the equation number (34) can be rewritten as follows:

$$AP = b \quad (36)$$

where;

$$A = \begin{bmatrix} (u_1 m_{31} - m_{11}) & (u_1 m_{32} - m_{12}) & (u_1 m_{33} - m_{13}) \\ (v_1 m_{31} - m_{21}) & (v_1 m_{32} - m_{22}) & (v_1 m_{33} - m_{23}) \\ (u_1 n_{31} - n_{11}) & (u_1 n_{32} - n_{12}) & (u_1 n_{33} - n_{13}) \\ (v_1 n_{31} - n_{21}) & (v_1 n_{32} - n_{22}) & (v_1 n_{33} - n_{23}) \end{bmatrix}$$

$$P = \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix}$$

$$b = \begin{bmatrix} m_{14} - u_1 m_{34} \\ m_{24} - v_1 m_{34} \\ n_{14} - u_1 n_{34} \\ n_{24} - v_1 n_{34} \end{bmatrix}$$

Then the Point 'P' in the world coordinate system can be determined as:

$$P = (A^T A)^{-1} A^T b \quad (37)$$

#### IV. RESULTS AND EXPERIMENT

The experiment is carried out to validate our method and to evaluate the performance of our system. Our experimental system consists of a digital camera (DH-HV2051UC) with  $1600 \times 1200$  pixels and a projector (DELL M110) with  $800 \times 1280$  pixels. The projector is placed at a distance of 196 cm

$$\left\{ \begin{array}{l} (u_1 m_{31} - m_{11}) X_W + (u_1 m_{32} - m_{12}) Y_W + (u_1 m_{33} - m_{13}) Z_W \\ (v_1 m_{31} - m_{21}) X_W + (v_1 m_{32} - m_{22}) Y_W + (v_1 m_{33} - m_{23}) Z_W \\ (u_1 n_{31} - n_{11}) X_W + (u_1 n_{32} - n_{12}) Y_W + (u_1 n_{33} - n_{13}) Z_W \\ (v_1 n_{31} - n_{21}) X_W + (v_1 n_{32} - n_{22}) Y_W + (v_1 n_{33} - n_{23}) Z_W \end{array} \right\} = \left\{ \begin{array}{l} m_{14} - u_1 m_{34} \\ m_{24} - v_1 m_{34} \\ n_{14} - u_1 n_{34} \\ n_{24} - v_1 n_{34} \end{array} \right\} \quad (35)$$

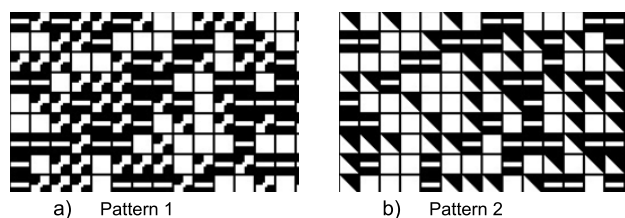


FIGURE 8. Texture of patterns used in the experiment.

TABLE 6. Shape description parameter and their threshold values.

Shape Description Primitive	Diagonally arranged squares	Filled square	Horizontal stripe or bar	Right triangle
Eccentricity ( $\epsilon$ )	0.9238	0	0.9682	0.8150
No of Hole (H)	0	0	0	0
Equivalent Diameter ( $ED_s$ )	13.1590	18.0541	9.0270	13.1590
Aspect Ratio (AR)	2.6110	1	4	1.7257
Solidity (S)	0.7391	1	1	1
Convex to Area Ratio (CAR)	1.3529	1	1	1
Perimeter to Area (PAR)	0.4054	0.2283	0.5456	0.3692
Rectangularity (Rect)	0.5313	1	1	0.5313
Circularity or Compactness (CR)	0.4475	0.7497	0.5250	0.5393
Convex Ratio (Convex)	0.6530	1.0952	1.1456	1.2745

whereas the camera is placed at a distance of 200 cm from the measuring surface, while the camera and projector are 18 cm apart. We perform our experiment on three surfaces: 1) A simple plane surface which is approximately 800 mm wide and 600 mm long, 2) The cylindrical surface which has 150 mm radius and 406 mm of height 3) The sculpture we used has the size of 223 mm of height, 190 mm of width and 227 mm of depth. The standard deviation of the cylindrical surface is equal to its radius which is 150 mm. The estimated standard deviation of the textured surface, i.e. sculpture is approximately between 200 to 225 mm.

For experimentation and proving our methodology, we form two projection patterns that are implemented by using three symbols of size  $16 \times 16$  pixels. So our measured resolution will be about 18 mm (refer to table 8). The two patterns are formed using diagonally arranged squares (symbol 5), filled square (symbol 7), horizontal bar (symbol 22), and right isosceles triangle (symbol 25). The only difference between the two projection patterns is that in the second pattern we replaced the symbol of the diagonally arranged square with an isosceles right triangle. To spread these symbols we use an RPRS of three alphanumeric bases with a size of  $45 \times 72$  primitives and window property of  $3 \times 3$ . The spaces between two consecutive symbols are of 2 pixels. The textures of these two patterns are shown in figure 8.

TABLE 7. Detected and decoded primitive.

Surface Type	Pattern 1 (Primitives)			Pattern 2 (Primitives)		
	Detected	Decoded	%	Detected	Decoded	%
Original Pattern	3240	3240	100	3240	3240	100
Simple Plane	1647	1614	98	1650	1617	98
Curve Cylinder	1160	1132	97.6	1161	1128	97.1
Textured Sculpture	689	585	84.9	685	578	84.4

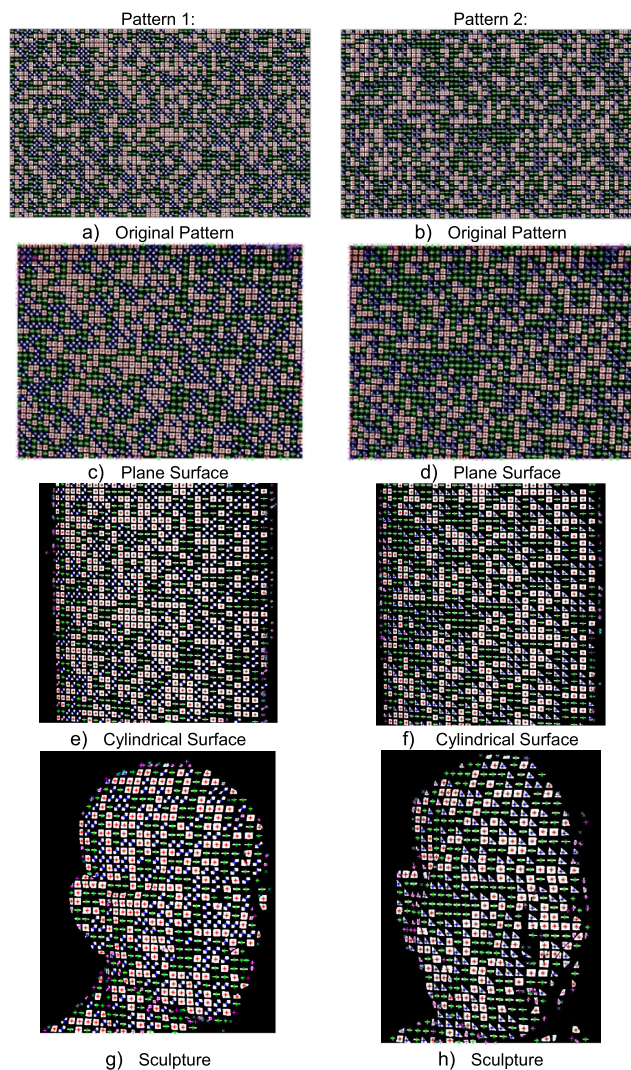


FIGURE 9. Classification or decoding of symbols.

To decode these patterns on the measuring surface we apply the method discuss in the previous section. The four symbols used in these patterns are classified using shape description parameters. The relevant shape description parameters used to classify four symbols in these patterns and their corresponding threshold values used for the classification are shown in table 6 which is a subset of table 10 at the appendix. Since each pattern consists of three symbols, therefore three shape description parameters may be

**TABLE 8. Comparison of resolution and covered area with other methods.**

Symb of Size	Depth (z) cm	Our Method		Albiter (2007)		Chen (2008)		Wijenayake (2012)	
		Area (cm <sup>2</sup> )	Resol ution (mm)	Area (cm <sup>2</sup> )	Resol ution (mm)	Area (cm <sup>2</sup> )	Resol ution (mm)	Area (cm <sup>2</sup> )	Resol ution (mm)
8X8	250	103.8 X 166	11.8	103.8 X 111.5	37.5	103.8 X 166	32.5	99.3 X 99.3	23
10X10			14.3						
12X12			16.8						
14X14			20.8						
16X16			23.3						
8X8	200	83 X 132.8	9.4	83 X 89.2	30	83 X 132.8	26	79.4 X 79.4	18.4
10X10			11.4						
12X12			13.4						
14X14			16.6						
16X16			18.6						
8X8	150	62.3 X 99.6	7.1	62.3 X 66.9	22.5	62.3 X 99.6	19.5	59.6 X 59.6	13.8
10X10			8.6						
12X12			10.1						
14X14			12.5						
16X16			14						
8X8	120	49.8 X 79.7	5.6	49.8 X 53.5	18	49.8 X 79.7	15.6	47.6 X 47.6	11
10X10			6.8						
12X12			8						
14X14			10						
16X16			11.1						
8X8	100	41.5 X 66.4	4.7	41.5 X 44.6	15	41.5 X 66.4	13	39.7 X 39.7	9.2
10X10			5.7						
12X12			6.7						
14X14			8.3						
16X16			9.3						
8X8	80	33.2 X 53.1	3.8	33.2 X 35.7	12	33.2 X 53.1	10.4	31.8 X 31.8	7.4
10X10			4.6						
12X12			5.4						
14X14			6.6						
16X16			7.4						
8X8	60	24.9 X 39.8	2.8	24.9 X 26.8	9	24.9 X 39.8	7.8	23.8 X 23.8	5.5
10X10			3.4						
12X12			4						
14X14			5						
16X16			5.6						
8X8	40	16.6 X 26.6	1.9	16.6 X 17.8	6	16.6 X 26.6	5.2	15.9 X 15.9	3.7
10X10			2.3						
12X12			2.7						
14X14			3.3						
16X16			3.7						

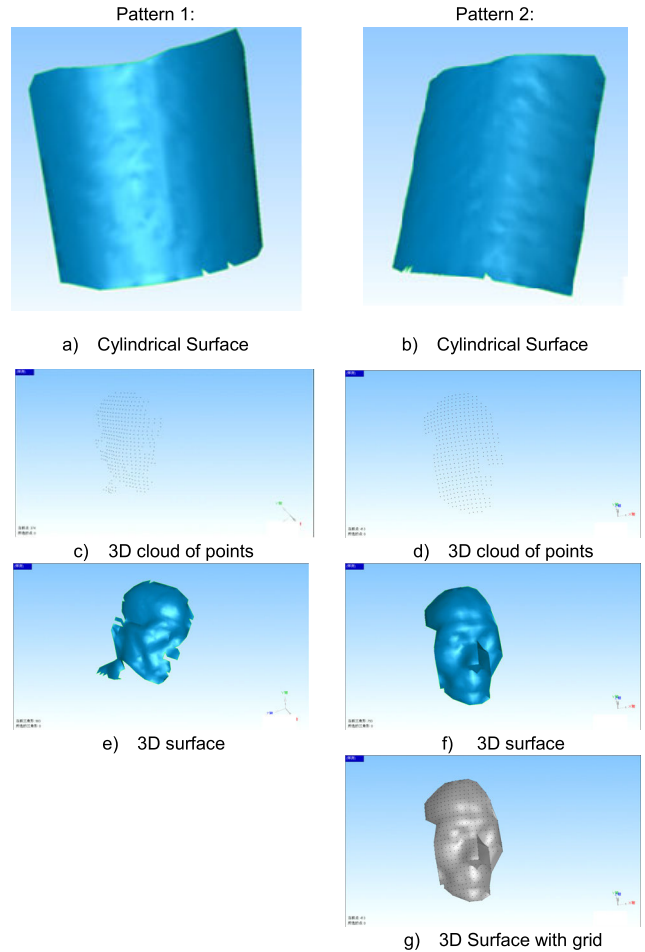
**TABLE 9. Time calculation for different processes of decoding.**

	Original Pattern	Plane Surface	Cylinder	Sculpture
Preprocessing (Filtering + Thresholding)	566	611	649	644
Labeling	42	53	41.5	38
Parameter calculation	587	365.6	361	271
Classification	3.3	2.2	2.7	2.7
Correspondence	485	480	331.1	318
Rate of Matching	0.19	0.3	0.29	0.5

Note: Time is measured in milliseconds

enough to classify symbols. The shape description parameters like rectangularity, eccentricity, aspect ratio, convex to area ratio, parameter to area ratio, area function, and direction or orientation can be enough and used to classify the symbols in both the patterns used in experimentation. If more shape description parameters will use to classify symbols than chances of errors due to the wrong classification will remove.

In the first step, rectangularity measure is used to differentiate diagonally arranged squares or isosceles right triangle from the two other alphabets or symbols which are filled squares and horizontal stripe. The square and the horizontal stripe have a higher value of rectangularity ratio almost equal to '1' when compared to diagonally arranged squares or isosceles right triangle which is only 0.53



**FIGURE 10. Reconstruction of 3D.**

(see table 6). After differentiating symbols using rectangularity further classification is made by using eccentricity, and the area function. Squares are separated from stripes using eccentricity. The value of eccentricity is utilized for classification among square or stripe. The eccentricity ratio for horizontal stripe symbol is the highest, i.e. equal to '1', and for the solid square symbol it is the lowest i.e. equal to '0'. Further classification is based on area function since the solid square symbol has the largest area while the horizontal stripe symbol has the lowest value of area function so they can be easily differentiated.

Our method of classification of symbols or alphabets in the projection pattern is validated through applying our algorithm on the original pattern and it has observed that 100% of symbols are decoded, and then we apply our method by projecting pattern on the plane surface, curved or cylindrical surface, and surfaces with more texture such as sculpture.

Table 7 shows the number of symbols or primitives decoded or classified for both the pattern used in experimentation when applying to different measuring surfaces. It is evident from table 7 that our decoding algorithm worked well when compared to other methods such as

Petriu *et al.* [24], Albiter *et al.* [28], Chen *et al.* [34], [35] and Wijenayake *et al.* [26]. Reference [24] was able to decode only 59% of primitive while [28] able to decode 95% of primitives on the cylindrical surface on the other hand we have decoded 97% of primitives. Due to the utilization of a monochromatic pattern having only two intensities of colors, white and black; our decoding algorithm will extract more feature points when compared to [26] and [34]. Since more shape description parameters are used to detect the symbols, therefore, no false detection or wrong classification of symbols has been observed. Our result shows that 100% of symbols have been decoded when applied to the original projection pattern which shows the method is the most reliable one.

Figure 9 shows detected and decoded or classified primitives or symbols for two patterns used in our experimentation purpose. It shows decoded primitives for 1) original (projected patterns), 2) patterns from a plane surface, 3) curved cylindrical surface, and 4) the textured surface i.e. sculpture. To differentiate symbols from one and another centroid positions of each symbol are represented with different colors. So, the centroid positions of diagonally arranged small squares (in pattern 1) or isosceles right triangle (in pattern 2) are represented with a blue star (\*). Similarly, the centroid positions of solid filled squares are marked with a red star (\*) whereas the centroid positions of horizontal stripes are shown with a green star (\*). The centroid positions of the symbols which are unable to decode or classify are represented by pink plus (+) sign.

The direction or orientation of the measuring surface is calculated from two symbols in each pattern i.e. from diagonally oriented square symbol (in pattern 1) or isosceles right triangle (in pattern 2) and the horizontal stripe symbol (in both patterns). The direction information of solid filled square will be calculated either from two of these symbols in the neighborhood as described in the previous section. The grid distance is calculated as described earlier in the previous section.

Our system can perform well from 40 to 250 cm of depth range. Table 8 shows the resolution obtained and area covered on a plane surface, in between the depth of 40 to 250 cm by using our system, and also compared with the systems proposed by other researchers such as Albiter *et al.* [28], Chen *et al.* [34], [35] and Wijenayake *et al.* [26]. These resolutions are achieved by using our experimental setup and projector. The resolution or accuracy achieved with our method is significantly higher than the previous method. We can achieve a resolution of 1.9 mm at the depth distance of 40 cm while the previous methods such as [28] has an accuracy of 6 mm, [35] has 5.2 mm and [26] has 3.7 mm.

The time durations were measured on the average core i5 computer for different processes. The time calculations are based on average times for each process, while each event is

optimized and processed many times, so the optimum results are being shown here. All-time calculations are measured in milliseconds. Table 9 shows the time durations for different processes involved during decoding. Each process has a specific time duration. The preprocessing time increases with the complexity of the surface while it decreases with the decrease of detected primitives. The number of decoded primitives are higher in the original pattern and simple surface, when compared to the complex or textured surface such as sculpture or cylinder. A similar phenomenon is observed in the processes such as; labeling, computation of shape description parameters, classification of symbols, and the establishment of correspondence. The matching rate increases with the complexity and texture of the surface. The rate of matching is less on the simple surface and high on the texture and complex surface.

After applying the techniques as described earlier the 3D of simple and complex surfaces, such as cylindrical objects and sculpture are measured at the accuracy level of 18 mm and are shown in figure 10.

## V. CONCLUSION

In this paper, a single shot novel method is introduced to generate a pixel-level design of projection pattern for structure light system (SLS) based on spatial encoding technique. Unlike the previous methods, our encoding technique is more flexible and designed and controllable up to the pixel level. We have proposed 25 geometrical shaped symbols, well-controlled in size. We have computed '10' shape description parameters for each symbol or alphabet, which will enable us to use and decode up to '8 to 9' symbols in a single projection pattern. So more symbols can be used and decoded in the future. Instead of M-arrays, we use robust pseudo-random sequences to spread symbols in a projection pattern. We present a comparatively easy and flexible technique to generate robust pseudo-random sequences of any required size and also ensuring their robustness through the window property. The use of more symbols in a projection pattern will enhance the robustness of RPRS. With our method, more flexible projection patterns that are controllable up to the pixel level of the projector are implemented and the projection patterns can be designed according to the required size of surface area. With our technique, a large surface area can be covered in a single shot arrangement. Due to control at the pixel level, the resolution of the measured 3D surface is also improved. The comparison with the previous methods shows that the accuracy level or resolution of our system is significantly better. We have implemented a new method for decoding based on grid distance between consecutive symbols.

## APPENDIX

See Table 10.

**TABLE 10.** Proposed symbols and their shape description parameters.


























Symbol No	Shape pattern	Symbol Shape (16 X16)	Symbol Size	$\epsilon^*$	H	ED <sub>s</sub>	AR	Solidity	$\theta$	CAR	PAR	Rect	CR	Convex	
1.	Thick reverse L Shaped		8X8	0.7303	0	7.8176	1.4639	0.8889	$\frac{\pi}{4}$	1.1250	0.5469	0.75	0.6967	0.9905	
			10X10			9.7721		0.8824		1.1333	0.4545		0.6454	0.9387	
			12X12			11.7265		0.8780		1.1389	0.3882		0.6143	0.9063	
			14X14			13.6809		0.8750		1.1429	0.3386		0.5935	0.8841	
16X16	15.6353	0.8727	1.1458	0.3000	0.5785	0.8679									
2.	Thick L Shaped		8X8	0.7303	0	7.8176	1.4639	0.8889	$-\frac{\pi}{4}$	1.1250	0.5469	0.75	0.6967	0.9905	
			10X10			9.7721		0.8824		1.1333	0.4545		0.6454	0.9387	
			12X12			11.7265		0.8780		1.1389	0.3882		0.6143	0.9063	
			14X14			13.6809		0.8750		1.1429	0.3386		0.5935	0.8841	
16X16	15.6353	0.8727	1.1458	0.3000	0.5785	0.8679									
3.	Inverted Thick L Shaped		8X8	0.7303	0	7.8176	1.4639	0.8889	$-\frac{\pi}{4}$	1.1250	0.5469	0.75	0.6967	0.9905	
			10X10			9.7721		0.8824		1.1333	0.4545		0.6454	0.9387	
			12X12			11.7265		0.8780		1.1389	0.3882		0.6143	0.9063	
			14X14			13.6809		0.8750		1.1429	0.3386		0.5935	0.8841	
16X16	15.6353	0.8727	1.1458	0.3000	0.5785	0.8679									
4.	Inverted thick reverse L Shaped		8X8	0.7303	0	7.8176	1.4639	0.8889	$\frac{\pi}{4}$	1.1250	0.5469	0.75	0.6967	0.9905	
			10X10			9.7721		0.8824		1.1333	0.4545		0.6454	0.9387	
			12X12			11.7265		0.8780		1.1389	0.3882		0.6143	0.9063	
			14X14			13.6809		0.8750		1.1429	0.3386		0.5935	0.8841	
16X16	15.6353	0.8727	1.1458	0.3000	0.5785	0.8679									
5.	Diagonally arranged small squares		8X8	0.9238	0	6.5795	2.6110	0.7727	$\frac{\pi}{4}$	1.2941	0.7151	0.5313	0.5751	0.8226	
			10X10	0.9250		8.1369	2.6314	0.7429		1.3462	0.6183	0.5200	0.5030	0.7464	
			12X12	0.9226		9.9656	2.5916	0.7647		1.3077	0.4985	0.5417	0.5158	0.7201	
			14X14	0.9223		11.6174	2.5874	0.7571		1.3208	0.4461	0.5408	0.4740	0.6767	
16X16	0.9238	13.1590	2.6110	0.7391	1.3529	0.4054	0.5313	0.4475	0.6530						
6.	Diagonally arranged small squares		8X8	0.9238	0	6.5795	2.6110	0.7727	$-\frac{\pi}{4}$	1.2941	0.7151	0.5313	0.5751	0.8226	
			10X10	0.9250		8.1369	2.6314	0.7429		1.3462	0.6183	0.5200	0.5030	0.7464	
			12X12	0.9226		9.9656	2.5916	0.7647		1.3077	0.4985	0.5417	0.5158	0.7201	
			14X14	0.9223		11.6174	2.5874	0.7571		1.3208	0.4461	0.5408	0.4740	0.6767	
16X16	0.9238	13.1590	2.6110	0.7391	1.3529	0.4054	0.5313	0.4475	0.6530						
7.	Filled square		8X8	0	0	9.0270	1.0000	1.0000	0	1.0000	0.4231	1.0000	0.8730	1.1819	
			10X10			11.2838					0.3492		0.8203	1.1456	
			12X12			13.5406					0.2969		0.7877	1.1226	
			14X14			15.7973					0.2581		0.7656	1.1068	
16X16	18.0541	0.2283	0.7497	1.0952											
8.	Unfilled square		8X8	0	1	7.8176	1.0000	0.7500	0	1.3333	0.5641	0.7500	0.6547	1.1819	
			10X10			9.0270		0.6400		1.5625	0.5456	0.6400	0.5250	1.1456	
			12X12			10.0925		0.5556		1.8000	0.5344	0.5556	0.4376	1.1226	
			14X14			11.0558		0.4898		2.0417	0.5270	0.4898	0.3750	1.1068	
16X16	11.9416	0.4375	2.2857	0.5218	0.4375	0.3280	1.0952								
9.	Thin reverse L Shaped		8X8	0.8398	0	5.9708	1.8421	0.6512	$\frac{\pi}{4}$	1.5357	0.9375	0.4375	0.4064	0.8381	
			10X10	0.8501		6.7703	1.8989	0.5625		1.7778	0.9469	0.3600	0.3098	0.7627	
			12X12	0.8398		8.9562	1.8421	0.6364		1.5714	0.6655	0.4375	0.3584	0.7632	
			14X14	0.8475		9.7721	1.8838	0.5769		1.7333	0.6636	0.3827	0.3028	0.7233	
16X16	0.8398	11.9416	1.8421	0.6292	1.5893	0.5144	0.4375	0.3375	0.7291						
10.	Thin L Shaped		8X8	0.8398	0	5.9708	1.8421	0.6512	$-\frac{\pi}{4}$	1.5357	0.9375	0.4375	0.4064	0.8381	
			10X10	0.8501		6.7703	1.8989	0.5625		1.7778	0.9469	0.3600	0.3098	0.7627	
			12X12	0.8398		8.9562	1.8421	0.6364		1.5714	0.6655	0.4375	0.3584	0.7632	
			14X14	0.8475		9.7721	1.8838	0.5769		1.7333	0.6636	0.3827	0.3028	0.7233	
16X16	0.8398	11.9416	1.8421	0.6292	1.5893	0.5144	0.4375	0.3375	0.7291						
11.	Inverted thin reverse L Shaped		8X8	0.8398	0	5.9708	1.8421	0.6512	$\frac{\pi}{4}$	1.5357	0.9375	0.4375	0.4064	0.8381	
			10X10	0.8501		6.7703	1.8989	0.5625		1.7778	0.9469	0.3600	0.3098	0.7627	
			12X12	0.8398		8.9562	1.8421	0.6364		1.5714	0.6655	0.4375	0.3584	0.7632	
			14X14	0.8475		9.7721	1.8838	0.5769		1.7333	0.6636	0.3827	0.3028	0.7233	
16X16	0.8398	11.9416	1.8421	0.6292	1.5893	0.5144	0.4375	0.3375	0.7291						
12.	Inverted thin L Shaped		8X8	0.8398	0	5.9708	1.8421	0.6512	$-\frac{\pi}{4}$	1.5357	0.9375	0.4375	0.4064	0.8381	
			10X10	0.8501		6.7703	1.8989	0.5625		1.7778	0.9469	0.3600	0.3098	0.7627	
			12X12	0.8398		8.9562	1.8421	0.6364		1.5714	0.6655	0.4375	0.3584	0.7632	
			14X14	0.8475		9.7721	1.8838	0.5769		1.7333	0.6636	0.3827	0.3028	0.7233	
16X16	0.8398	11.9416	1.8421	0.6292	1.5893	0.5144	0.4375	0.3375	0.7291						
13.	Unfilled Rhombus		8X8	0	1	5.9708	1.0000	0.7000	0	1.4286	0.7166	0.4375	0.6955	1.5949	
			10X10			6.7703		0.6000		1.6667	0.7136	0.3600	0.5456	1.5571	
			12X12			7.4848		0.5238		1.9091	0.7116	0.3056	0.4488	1.5330	
			14X14			9.8370		0.5588		1.7895	0.5152	0.3878	0.4958	1.4303	
16X16	10.3418	0.5833	1.7143	0.5218	0.3281	0.4637	1.5038								
14.	Filled Rhombus		8X8	0	0	7.1365	1.0000	1.0000	0	1.0000	0.5016	0.6250	0.9936	1.5949	
			10X10			8.7404					0.4281	0.6000	0.9093	1.5571	
			12X12			10.3418					0.5833	0.3728	0.5833	0.8568	1.5330
			14X14			13.1590					0.2879	0.6939	0.8872	1.4303	
16X16	13.5406	0.2956	0.5625	0.7950	1.5038										
15.	Thick X Alphabet		8X8	0.5324	0	5.9708	1.0000	0.4375	$\frac{\pi}{2}$	2.2857	1.2323	0.4375	0.2352	0.3478	
			10X10	0.4909		6.7703		0.3600		2.7778	1.2709	0.3600	0.1720	0.2623	
			12X12	0.4570		7.4848		0.3056		3.2727	1.2955	0.3056	0.1354	0.2105	
			14X14	0		9.8370		0.3878		2.5789	0.8708	0.3878	0.1735	0.2418	
16X16	0	10.5851	0.3438	2.9091	0.8799	0.3438	0.1468	0.2066							

TABLE 10. (Continued.) Proposed symbols and their shape description parameters.

16.	Vertical isosceles triangle connected at the middle		8X8	0.7385		7.1365	1.4832	0.6250		1.6000	0.7956	0.6250	0.3950	0.6285
			10X10	0.7559		8.7404	1.5275	0.6000		1.6667	0.6895	0.6000	0.3506	0.5802
			12X12	0.7670	0	10.3418	1.5584	0.5833		1.7143	0.6061	0.5833	0.3241	0.5500
			14X14	0.7746		11.9416	1.5811	0.5714		1.7500	0.5398	0.5714	0.3064	0.5293
			16X16	0.7802		13.5406	1.5986	0.5625		1.7778	0.4861	0.5625	0.2939	0.5143
17.	Horizontal isosceles triangles connected at the middle		8X8	0.7385		7.1365	1.4832	0.6250		1.6000	0.7956	0.6250	0.3950	0.6285
			10X10	0.7559		8.7404	1.5275	0.6000		1.6667	0.6895	0.6000	0.3506	0.5802
			12X12	0.7670	0	10.3418	1.5584	0.5833		1.7143	0.6061	0.5833	0.3241	0.5500
			14X14	0.7746		11.9416	1.5811	0.5714		1.7500	0.5398	0.5714	0.3064	0.5293
			16X16	0.7802		13.5406	1.5986	0.5625		1.7778	0.4861	0.5625	0.2939	0.5143
18.	A vertical bar inside square		8X8	0.3447		7.1365	1.0653	0.6250		1.6000	0.6769	0.6250	0.5456	1.1819
			10X10	0.3722		8.1369	1.0774	0.5200		1.9231	0.6715	0.5200	0.4265	1.1456
			12X12	0.3883	2	9.0270	1.0851	0.4444		2.2500	0.6681	0.4444	0.3501	1.1226
			14X14	0.2512		12.1530	1.0331	0.5918		1.6897	0.4362	0.5918	0.4531	1.1068
			16X16	0.2639		13.1590	1.0367	0.5313		1.8824	0.4297	0.5313	0.3983	1.0952
19.	A horizontal bar inside square		8X8	0.3447		7.1365	0.6250		1.6000	0.6769	0.6250	0.5456	1.1819	
			10X10	0.3722		8.1369	1.2000	0.5200		1.9231	0.6715	0.5200	0.4265	1.1456
			12X12	0.3883	2	9.0270	0.4444	0.4444	0	2.2500	0.6681	0.4444	0.3501	1.1226
			14X14	0.2512		12.1530	0.5918	0.5918		1.6897	0.4362	0.5918	0.4531	1.1068
			16X16	0.2639		13.1590	0.5313	0.5313		1.8824	0.4297	0.5313	0.3983	1.0952
20.	+ inside unfilled square		8X8			7.8176	0.7500		1.3333	0.5641	0.7500	0.6547	1.1819	
			10X10			9.0270	0.6400		1.5625	0.5456	0.6400	0.5250	1.1456	
			12X12	0	4	10.0925	1.0000	0.5556	0	1.8000	0.5344	0.5556	0.4376	1.1226
			14X14			12.9641		0.6735		1.4848	0.3833	0.6735	0.5156	1.1068
			16X16			14.0935		0.6094		1.6410	0.3746	0.6094	0.4568	1.0952
21.	Plus '+' sign		8X8			5.9708	0.7000		1.4286	0.8489	0.4375	0.4956	0.6732	
			10X10			6.7703	0.6000		1.6667	0.8780	0.3600	0.3603	0.5062	
			12X12	0	0	7.4848	1.0000	0.5238	0	1.9091	0.8965	0.3056	0.2828	0.4056
			14X14			11.0558		0.7059		1.4167	0.4926	0.4898	0.4293	0.5075
			16X16			11.9416		0.6512		1.5357	0.4922	0.4375	0.3685	0.4354
22.	Horizontal stripe or bar		8X8	0.9682		4.5135	4.0000			0.9572		0.6821	1.3058	
			10X10	0.9798		5.0463	5.0000			0.9618		0.5405	1.2477	
			12X12	0.9860	0	5.5279	6.0000	1.0000	0	1.0000	0.9648	1.0000	0.4476	1.2092
			14X14	0.9583		8.4440	3.5000				0.5535		0.5829	1.1614
			16X16	0.9682		9.0270	4.0000				0.5456		0.5250	1.1456
23.	Vertical stripe or bar		8X8	0.9682		4.5135	4.0000			0.9572		0.6821	1.3058	
			10X10	0.9798		5.0463	5.0000			0.9618		0.5405	1.2477	
			12X12	0.9860	0	5.5279	6.0000	1.0000	$\frac{\pi}{2}$	1.0000	0.9648	1.0000	0.4476	1.2092
			14X14	0.9583		8.4440	3.5000				0.5535		0.5829	1.1614
			16X16	0.9682		9.0270	4.0000				0.5456		0.5250	1.1456
24.	Right triangle		8X8	0.8090		5.9708	1.7014			0.7115	0.4375	0.7054	1.4054	
			10X10	0.8119		7.5694	1.7129			0.5923	0.4500	0.6334	1.3506	
			12X12	0.8134	0	9.1670	1.7190	1.0000	$-\frac{\pi}{4}$	1.0000	0.5059	0.4583	0.5921	1.3179
			14X14	0.8142		10.7641	1.7226				0.4409	0.4643	0.5654	1.2961
			16X16	0.8148		12.3608	1.7248				0.4054	0.4688	0.5467	1.2807
25.	Right triangle		8X8	0.8107		6.7703	1.7082			0.6469	0.5625	0.6637	1.3740	
			10X10	0.8127		8.3683	1.7164			0.5458	0.5500	0.6103	1.3324	
			12X12	0.8139	0	9.9656	1.7210	1.0000	$-\frac{\pi}{4}$	1.0000	0.4712	0.5417	0.5774	1.3060
			14X14	0.8145		11.5624	1.7238				0.4141	0.5357	0.5553	1.2878
			16X16	0.8150		13.1590	1.7257				0.3692	0.5313	0.5393	1.2745

\* The alphabet no 1 to 6, 9 to 12, 16, 17, 22 to 25 are carrying the direction information since their eccentricity values are greater than 0.7.

REFERENCES

[1] J. G. Webster, T. Bell, B. Li, and S. Zhang, "Structured light techniques and applications," *Wiley Encycl. Electr. Electron. Eng.*, no. 8, pp. 1–24, 2016, doi: 10.1002/047134608x.w8298.

[2] S. Zhang, "High-speed 3D shape measurement with structured light methods: A review," *Opt. Lasers Eng.*, vol. 106, pp. 119–131, Jul. 2018, doi: 10.1016/j.optlaseng.2018.02.017.

[3] J. Salvi, S. Fernandez, T. Pribanic, and X. Llado, "A state of the art in structured light patterns for surface profilometry," *Pattern Recognit.*, vol. 43, no. 8, pp. 2666–2680, Aug. 2010, doi: 10.1016/j.patcog.2010.03.004.

[4] T. Moons, "3D reconstruction from multiple images part 1: Principles," *Found. Trends Comput. Graph. Vis.*, vol. 4, no. 4, pp. 287–404, 2008, doi: 10.1561/0600000007.

[5] R. J. Valkenburg and A. M. Mc Ivor, "Accurate 3D measurement using a structured light system," *Image Vis. Comput.*, vol. 2909, pp. 68–80, Jan. 1997.

[6] Z. Wei, F. Zhou, and G. Zhang, "3D coordinates measurement based on structured light sensor," *Sens. Actuators A, Phys.*, vol. 120, no. 2, pp. 527–535, May 2005, doi: 10.1016/j.sna.2004.12.007.

[7] G. Zhang, J. He, and X. Li, "3D vision inspection for internal surface based on circle structured light," *Sens. Actuators A, Phys.*, vol. 122, no. 1, pp. 68–75, Jul. 2005, doi: 10.1016/j.sna.2005.04.012.

[8] J. Batlle, E. Mouaddib, and J. Salvi, "Recent progress in coded structured light as a technique to solve the correspondence problem: A survey," *Pattern Recognit.*, vol. 31, no. 7, pp. 963–982, 1998, doi: 10.1016/S0031-3203(97)00074-5.

[9] J. Salvi, J. Pagès, and J. Batlle, "Pattern codification strategies in structured light systems," *Pattern Recognit.*, vol. 37, no. 4, pp. 827–849, Apr. 2004, doi: 10.1016/j.patcog.2003.10.002.

[10] J. Geng, "Structured-light 3D surface imaging: A tutorial," *Adv. Opt. Photon.*, vol. 3, no. 2, p. 128, Jun. 2011, doi: 10.1364/aop.3.000128.

[11] C. Zuo, S. Feng, L. Huang, T. Tao, W. Yin, and Q. Chen, "Phase shifting algorithms for fringe projection profilometry: A review," *Opt. Lasers Eng.*, vol. 109, pp. 23–59, Oct. 2018, doi: 10.1016/j.optlaseng.2018.04.019.

[12] S. Zhang, "Recent progresses on real-time 3D shape measurement using digital fringe projection techniques," *Opt. Lasers Eng.*, vol. 48, no. 2, pp. 149–158, Feb. 2010, doi: 10.1016/j.optlaseng.2009.03.008.

[13] S. Rusinkiewicz, O. Hall-Holt, and M. Levoy, "Real-time 3D model acquisition," in *Proc. 29th Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH)*, 2002, pp. 438–446, doi: 10.1145/566570.566600.

[14] O. Hall-Holt and S. Rusinkiewicz, "Stripe boundary codes for real-time structured-light range scanning of moving objects," in *Proc. 8th IEEE Int. Conf. Comput. Vis. (ICCV)*, 2001, pp. 359–366, doi: 10.1109/ICCV.2001.937648.



- [15] K. L. Boyer and A. C. Kak, "Color-encoded structured light for rapid active ranging," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-9, no. 1, pp. 14–28, Jan. 1987.
- [16] H. Hugli and G. Maitre, "Generation and use of color pseudo random sequences for coding structured light in active ranging," *Ind. Insp.*, vol. 1010, p. 75, Feb. 1989, doi: [10.1117/12.949215](https://doi.org/10.1117/12.949215).
- [17] C. Je, S. W. Lee, and R. H. Park, "High-contrast color-stripe pattern for rapid structured-light range imaging," in *Proc. 8th Eur. Conf. Comput. Vis. (ECCV)*, in Lecture Notes in Computer Science, vol. 3021, 2004, pp. 95–107, doi: [10.1007/978-3-540-24670-1\\_8](https://doi.org/10.1007/978-3-540-24670-1_8).
- [18] L. Zhang, B. Curless, and S. M. Seitz, "Rapid shape acquisition using color structured light and multi-pass dynamic programming," in *Proc. 1st Int. Symp. 3D Data Process. Visualizat. Transmiss.*, 2002, pp. 24–36, doi: [10.1109/TDPVT.2002.1024035](https://doi.org/10.1109/TDPVT.2002.1024035).
- [19] J. Pagés, J. Salvi, C. Collewet, and J. Forest, "Optimised de Bruijn patterns for one-shot shape acquisition," *Image Vis. Comput.*, vol. 23, no. 8, pp. 707–720, Aug. 2005, doi: [10.1016/j.imavis.2005.05.007](https://doi.org/10.1016/j.imavis.2005.05.007).
- [20] P. Vuylsteke and A. Oosterlinck, "Range image acquisition with a single binary-encoded light pattern," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 2, pp. 148–164, 1990, doi: [10.1109/34.44402](https://doi.org/10.1109/34.44402).
- [21] M. Maruyama and S. Abe, "Range sensing by projecting multiple slits with random cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 6, pp. 647–651, Jun. 1993.
- [22] H. Morita, K. Yajima, and S. Sakata, "Reconstruction of surfaces of 3-D objects by M-array pattern projection method," in *Proc. 2nd Int. Conf. Comput. Vis.*, Dec. 1988, pp. 468–473, doi: [10.1109/ccv.1988.590025](https://doi.org/10.1109/ccv.1988.590025).
- [23] P. M. Griffin, L. S. Narasimhan, and S. R. Yee, "Generation of uniquely encoded light patterns for range data acquisition," *Pattern Recognit.*, vol. 25, no. 6, pp. 609–616, 1992, doi: [10.1016/0031-3203\(92\)90078-W](https://doi.org/10.1016/0031-3203(92)90078-W).
- [24] E. M. Petriu, Z. Sakr, H. J. W. Spoelder, and A. Moica, "Object recognition using pseudo-random color encoded structured light," in *Proc. 17th IEEE Instrum. Meas. Technol. Conf.*, May 2002, pp. 1237–1241, doi: [10.1109/imtc.2000.848675](https://doi.org/10.1109/imtc.2000.848675).
- [25] E. M. Petriu, T. Bieseman, N. Trif, W. S. Mcmath, and S. K. Yeung, "Visual object recognition using pseudo-random grid encoding," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Jul. 1992, pp. 1617–1624.
- [26] U. Wijenayake, S.-I. Choi, and S.-Y. Park, "Combination of color and binary pattern codification for an error correcting M-array technique," in *Proc. 9th Conf. Comput. Robot Vis.*, May 2012, pp. 139–146, doi: [10.1109/CRV.2012.26](https://doi.org/10.1109/CRV.2012.26).
- [27] R. A. Morano, C. Ozturk, R. Conn, S. Dubin, S. Zietz, and J. Nissano, "Structured light using pseudorandom codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 3, pp. 322–327, Mar. 1998.
- [28] C. Albitar, P. Graebing, and C. Doignon, "Robust structured light coding for 3D reconstruction," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Oct. 2007, pp. 7–12, doi: [10.1109/ICCV.2007.4408982](https://doi.org/10.1109/ICCV.2007.4408982).
- [29] J. Lu, J. Han, E. Ahsan, G. Xia, and Q. Xu, "A structured light vision measurement with large size M-array for dynamic scenes," in *Proc. 35th Chin. Control Conf. (CCC)*, Jul. 2016, pp. 3834–3839, doi: [10.1109/ChiCC.2016.7553951](https://doi.org/10.1109/ChiCC.2016.7553951).
- [30] X.-J. Jia, Z.-J. Zhang, and Q.-C. Yu, "Construction for M-arrays and application in structured light," *J. Shanghai Univ.*, vol. 15, pp. 63–68, Feb. 2011.
- [31] P. M. Will and K. S. Pennington, "Grid coding: A novel technique for image processing," *Proc. IEEE*, vol. 60, no. 6, pp. 669–680, Jun. 1972.
- [32] G. Hu and G. Stockman, "3-D surface solution using structured light and constraint propagation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 4, pp. 390–402, Apr. 1989.
- [33] A. V. Strat and M. M. Oliveira, *A Point-and-Shoot Color 3D Camera SUNY at Stony Brook*. IEEE Computer Society Press, 2003.
- [34] S. Y. Chen, Y. F. Li, and J. Zhang, "Realtime structured light vision with the principle of unique color codes," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 429–434, doi: [10.1109/ROBOT.2007.363824](https://doi.org/10.1109/ROBOT.2007.363824).
- [35] S. Y. Chen, Y. F. Li, and J. Zhang, "Vision processing for realtime 3-D data acquisition based on coded structured light," *IEEE Trans. Image Process.*, vol. 17, no. 2, pp. 167–176, Feb. 2008.
- [36] T. Etzion, "Constructions for perfect maps and pseudorandom arrays," *IEEE Trans. Inf. Theory*, vol. 34, no. 5, pp. 1308–1316, Sep. 1988, doi: [10.1109/18.21260](https://doi.org/10.1109/18.21260).
- [37] V. Villena-Martínez, A. Fuster-Guilló, J. Azorín-López, M. Saval-Calvo, J. Mora-Pascual, J. Garcia-Rodríguez, and A. Garcia-Garcia, "A quantitative comparison of calibration methods for RGB-D sensors using different technologies," *Sensors*, vol. 17, no. 2, p. 243, Jan. 2017, doi: [10.3390/s17020243](https://doi.org/10.3390/s17020243).
- [38] B. Haefner, S. Peng, A. Verma, Y. Queau, and D. Cremers, "Photometric depth super-resolution," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Jun. 18, 2019, doi: [10.1109/tpami.2019.2923621](https://doi.org/10.1109/tpami.2019.2923621).
- [39] L. Kurnianggoro, S. Kom, and K.-H. Jo, "A survey of 2D shape representation: Methods, evaluations, and future research directions," *Neurocomputing*, vol. 300, pp. 1–16, Jul. 2018, doi: [10.1016/j.neucom.2018.02.093](https://doi.org/10.1016/j.neucom.2018.02.093).
- [40] M. Yang, K. Kpalma, and J. Ronsin, "A survey of shape feature extraction techniques," in *Pattern Recognition*, P.-Y. Yin, Ed. Rijeka, Croatia: InTech, 2008, pp. 43–90.
- [41] D. Zhang and G. Lu, "Review of shape representation and description techniques," *Pattern Recognit.*, vol. 37, no. 1, pp. 1–19, Jan. 2004, doi: [10.1016/j.patcog.2003.07.008](https://doi.org/10.1016/j.patcog.2003.07.008).
- [42] M. Peura and J. Iivarinen, "Efficiency of simple shape descriptors," in *Proc. 3rd Int. Work. Vis. Form*, 1997, pp. 443–451.
- [43] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, London, U.K.: Pearson, 2009, pp. 861–877, doi: [10.1117/1.3115362](https://doi.org/10.1117/1.3115362).
- [44] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing: Analysis and Machine Vision*, 3rd ed. Toronto, ON, Canada: Thomson, 2008.
- [45] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, "Deep learning for visual understanding: A review," *Neurocomputing*, vol. 187, pp. 27–48, Apr. 2016, doi: [10.1016/j.neucom.2015.09.116](https://doi.org/10.1016/j.neucom.2015.09.116).
- [46] J. Rivat and A. Sarkozy, "On pseudorandom sequences and their application," *Electron. Notes Discrete Math.*, vol. 21, pp. 369–370, Aug. 2005, doi: [10.1016/j.endm.2005.07.060](https://doi.org/10.1016/j.endm.2005.07.060).
- [47] J. Rivat and A. Sárkozy, "On pseudorandom sequences and their application," in *General Theory of Information Transfer and Combinatorics (Lecture Notes in Computer Science)*, vol. 4123. Berlin, Germany: Springer-Verlag, 2006, pp. 343–361, doi: [10.1007/11889342\\_19](https://doi.org/10.1007/11889342_19).
- [48] M. Matsumoto and T. Nishimura, "Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator," *ACM Trans. Model. Comput. Simul.*, vol. 8, no. 1, pp. 3–30, Jan. 1998.
- [49] *Mersenne Twister Home Page—A Very Fast Random Number Generator*. Accessed: Oct. 27, 1997. [Online]. Available: <http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/emt.html>
- [50] M. Matsumoto, M. Saito, H. Haramoto, and T. Nishimura, "Pseudorandom number generation: Impossibility and compromise," *J. Univ. Comput. Sci.*, vol. 12, no. 6, pp. 672–690, 2006, doi: [10.3217/jucs-012-06-0672](https://doi.org/10.3217/jucs-012-06-0672).
- [51] A. Jagannatham, "Mersenne twister—A pseudo random number generator and its variants," *ACM Trans. Model. Comput. Simul.*, vol. 8, no. 1, pp. 3–30, 2008.
- [52] L. Nie, Y. Ye, and Z. Song, "Method for calibration accuracy improvement of projector-camera-based structured light system," *Opt. Eng.*, vol. 56, no. 7, Jul. 2017, Art. no. 074101, doi: [10.1117/1.oe.56.7.074101](https://doi.org/10.1117/1.oe.56.7.074101).
- [53] J. Salvi, X. Armangué, and J. Battle, "A comparative review of camera calibrating methods with accuracy evaluation," *Pattern Recognit.*, vol. 35, no. 7, pp. 1617–1635, Jul. 2002.
- [54] B. Huang, S. Ozdemir, Y. Tang, C. Liao, and H. Ling, "A single-shot-per-pose camera-projector calibration system for imperfect planar targets," in *Proc. IEEE Int. Symp. Mixed Augmented Reality Adjunct (ISMAR-Adjunct)*, Oct. 2018, pp. 15–20, doi: [10.1109/ISMAR-Adjunct.2018.00023](https://doi.org/10.1109/ISMAR-Adjunct.2018.00023).
- [55] D. Moreno and G. Taubin, "Simple, accurate, and robust projector-camera calibration," in *Proc. 2nd Int. Conf. 3D Imag., Model., Process., Visualizat. Transmiss.*, Oct. 2012, pp. 464–471, doi: [10.1109/3DIMPVT.2012.77](https://doi.org/10.1109/3DIMPVT.2012.77).
- [56] M. Sezgin and B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation," *J. Electron. Imag.*, vol. 13, no. 1, pp. 146–165, 2004, doi: [10.1117/1.1631315](https://doi.org/10.1117/1.1631315).
- [57] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, vol. 1. Reading, MA, USA: Addison-Wesley, 1992.
- [58] K. Hata and S. Savarese, *CS231A Course Notes 1: Camera Models*. Stanford, CA, USA: Stanford Univ., 2015, p. 16.
- [59] S. Savarese, *Lecture 2: Camera Models*. Stanford, CA, USA: Stanford Univ., 2015, p. 18.



**AHSAN ELAHI** received the master's degree in control science and engineering from Harbin Engineering University (HEU), in 2015, where he is currently pursuing the Ph.D. degree. His current research interests include computer vision, control systems, and smart boat design.



**QI-DAN ZHU** received the Ph.D. degree. He is currently a Professor and a Ph.D. Tutor with Harbin Engineering University. He holds six invention patents. He has published more than 100 high-quality academic papers in national core journals and academic conferences at home and abroad.



**JUN LU** received the Ph.D. degree. He is currently a Professor and a Ph.D. Tutor with Harbin Engineering University. He has written four books. He has published more than 50 academic articles in national and international journals.



**LI YONG** received the master's degree in detective technology and automatic device from Heilongjiang University, in 2014. He is currently pursuing the Ph.D. degree with Harbin Engineering University (HEU). His current research interests include computer vision and deep learning.

...