15463 Final Project: 3D Structured Light with XOR, Binary, Gray Codes

RUSLANA FOGLER, Carnegie Mellon University

Three-Dimensional Structured Light is a method used to reconstruct point clouds of a 3D scene by relying on matching correspondences between a camera and a projector. This process is highly advantageous for reconstructing stationary scenes because it is more efficient point matching than conventional stereo vision with two cameras, because of rapid and cost-effective data capture, and because it can be reused to develop more scenes after the calibration is completed. By using binary or gray codes, the number of images required to create point correspondences is reduced to $\lfloor log_2 \text{Number of Column Pixels} \rfloor$, but the results are also susceptible to subsurface light scattering and interreflections. Instead, another approach is using XOR codes, which are more robust to the aforementioned phenomena. In my 15463 final project, I implemented a 3D Structured Light project that supports binary, gray, and XOR codes.

Additional Key Words and Phrases: Camera Calibration, Projector Calibration, Structured Light, Stereo Vision, 3D Reconstruction

ACM Reference Format:
Ruslana Fogler. 2025. 15463 Final Project: 3D Structured Light with XOR, Binary, Gray Codes. *ACM Trans. Graph.* 1, 1 (December 2025), 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

float

1 INTRODUCTION

1.1 3D Structured Lighting Background

Since the late 20th century, 3D Structured Light has gained traction as being an efficient, relatively cheap, and highly accurate way to construct a 3D visualization of a scene. This growth has culminated in efforts such as the Digital Michelangelo Project [1], enormous amounts of research, and applications to industries such as manufacturing, healthcare, cultural preservation, etc. To use the method, a user must set up a light projector and a camera pair to face a scene. The light projector is made to shine a number of certain known patterns on a scene–typically in the form of column stripes. The camera is used in tandem to capture images.

With the acquired image data, the user can extract row/column correspondences from the illuminated scenes. If the camera and projector's intrinsic and extrinsic parameters are known, they can be used to form a stereo pair, and the decoded row/columns serve as epipolar point correspondence. By back-projecting rays and performing triangulation, the depth of a scene can be recovered.

Author's address: Ruslana Fogler, rfogler@andrew.cmu.edu, Carnegie Mellon University.

ACM 0730-0301/2025/12-ART
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1.2 Binary/Gray Lighting Patterns

The most naive pattern to project onto a scene for Structured Light Decoding is illuminating each column. Then, the camera can take a picture for each column that has been illuminated, and column correspondences can be easily picked out with each picture. However, this requires taking as many as 4000 pictures if the user's picture was 4000 pixels wide, which is very impractical.

Instead, a better method is projecting a set of black-and-white binary or gray code stripes onto the scene. With this pattern, each point on the object is illuminated with a different series of stripes per picture, and the unique code can be picked out to identify the column correspondence. These methods reduce the number of images down to $\lfloor log_2 \text{Number of Column Pixels} \rfloor$. While effective, binary and gray code stripe illumination both have weaknesses.

1.3 Global Illumination Problems

Unfortunately, when a scene is globally illuminated by a low-frequency pattern from long ranges, objects within the scene are subject to strong inter-reflections. These additional artifacts will cause certain parts of the scene to appear brighter in captured image, and if they exceed a certain threshold, they can derail the point correspondence decoding afterwards. In these circumstances, higher frequency light is more suitable. However, shining high frequency light on a scene is also to susceptible to effects such as sub-surface scattering, where the projector's incident light is low-pass filtered out by nature of the scene and correspondences become difficult to find.

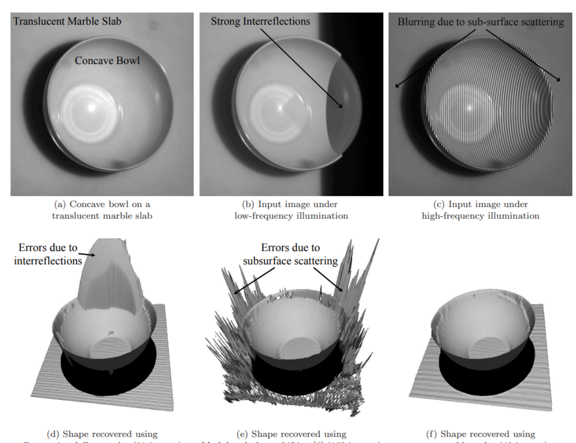

Fig. 1. Errors in reconstruction from interreflections/sub-surface scattering.From figure 7 of Paper "A Practical Approach to 3D Scanning in the Presence of Interreflections, Subsurface Scattering and Defocus"

To solve this problem, Gupta et al. [2] proposed obtaining the low-frequency illuminated light result by applying an XOR to two high frequency patterns. This preserves the ability to obtain Structured Light data without inter-reflections by shining high frequency patterns all throughout, and also enables the user to obtain the would-be low frequency direct lighting result through an additional operator. Thus, XOR codes are a more robust way to collect accurate point correspondences for 3D Structured Light. While not entirely free of long/short range or other complex light effects, they outperform binary and gray codes in theory.

2 METHODS

2.1 Implementing Codes and Decoding

After securing equipment (a Sony A7III Camera, tripod, Kodak Luma 350 Projector), the first step was to create a series of binary, gray, and XOR codes. These images were rather straightforward to generate with numpy array functions, and for XOR, XOR-04 for used for the first version (XOR-ing final image as the reference base plane). Much of my algorithms for generating patterns were taken from online procedures [3].

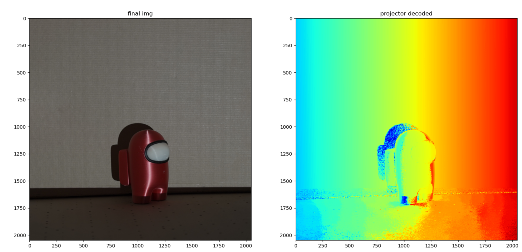

Fig. 2. Gray-Code Decoded Amongus 3D Print, from 10 gray code projections

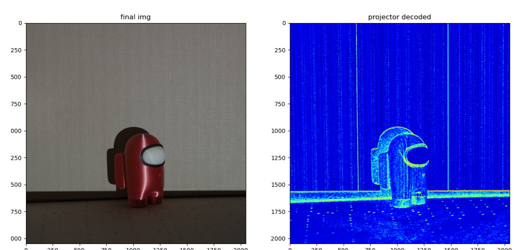

Fig. 3. XOR-Code Decoded Amongus 3D Print, from 10 XOR code projections

Decoding was equally straightforward. The scene was chosen to be of an image 2048 x 2048 pixel size, because increasing the columns any larger would lead to increasingly inaccurate decodings (short-range effects). After capturing 10 images of the scene, the images were stacked and codes were extracted per-pixel. The threshold for determining a corresponding '0' or '1' in each pixel position across the stack of images was simple the per-pixel mean of the stack.

2.2 Camera Calibration

To use stereo triangulation, the camera's intrinsic and extrinsic parameters needed to be found. Finding the camera's intrinsic parameters involved simply capturing images of the 9x7 chessboard from a series of different poses and then using Zhang's method [4] via `cv2.calibrateCamera` to extract the intrinsic and distortion. This process actually took significant amounts of time and re-doing, especially after a belated realization that cropping an image would alter the camera's respective f parameters.

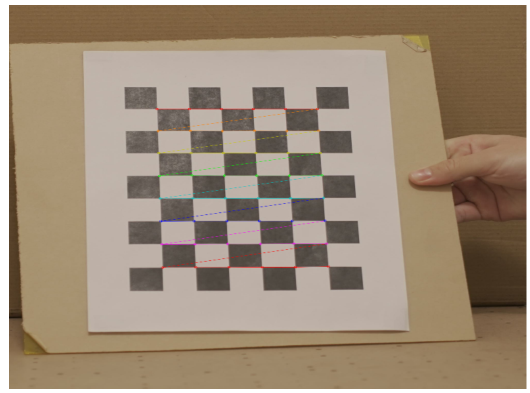

Fig. 4. Camera Calibration Image 1

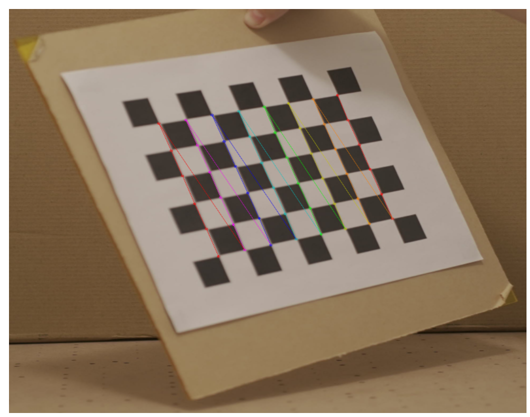

Fig. 5. Camera Calibration Image 2

2.3 Projector Calibration

Another requirement that was needed for stereo triangulation was the projector's intrinsic and extrinsic parameters. Ultimatley, the method used was from Daniel Moreno and Gabriel Taubin's projector calibation proposal [4]. Their paper proposed the following method: after calculating the camera's intrinsic parameters, projector calibration can be done by projecting all gray code patterns onto a chessboard at various poses (in this case, five were used). Then, to approximate the projector's intrinsic matrix, the row/col of each chessboard corner that were captured by the camera could be mapped by homography to the gray-decoded image. Instead of a global homography, however, the process was made more robust by capturing individual homographies in a patch neighborhood for each corner. If a homography was found successfully, it could be

ALGORITHM 1: Projector Calibration

Input: Chessboard Images $I_{chessboards}$, decoded gray-code image $I_{decoded}$, Camera's Intrinsic Matrix and Distortion Parameters K_c, d_c
Output: Projector's Intrinsic Matrix, Projector's Distortion Parameters K_p, d_p
;
for *each image* I *in* $I_{chessboards}$ **do**
  $Corners_c$ = findChessboardCorners(I, K_c, d_c)
  for *each* $corner_c$ *in* $Corners$ **do**
    $patch$ = get_neighborhood($corner$)
    for p_x, p_y *in* $patch$ **do**
      if $decoded[p_y] - p_x < tolerance$ **then**
        pp_x = decoded[p_x] $pp_y = y$
      end
    end
    H = getHomography(all pp_x, all pp_y)
    if H *exists* **then**
      $corner_p$ = H × $corner_c$
      $Corners_p$.append($corner_p$)
    end
  end
end
K_p, d_p = calibrateCamera($Corners_c, Corners_p, I_{chessboards}$)

used to map the camera's captured chessbord corner to the projector's decoded version to form a new point, which then could be fed into `cv2.calibrateCamera` to find the projector's intrinsic matrix and distortion parameters.

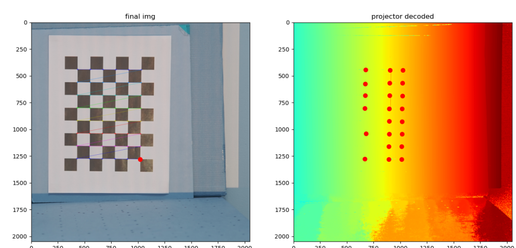

Fig. 6. Checkerboard Pose 1, Projector Calibration Corner Mapping via patch neighborhood homographies

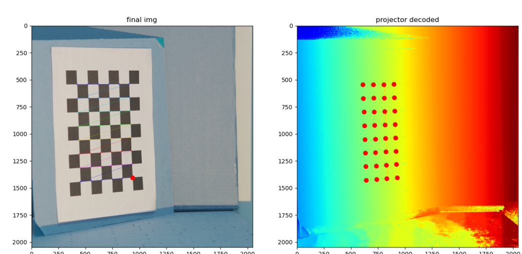

Fig. 7. Checkerboard Pose 2, Projector Calibration Corner Mapping

Then, after finding intrinsic/distortion parameters for both the camera and projector, the extrinsic Rotation and Translation matrices from one to the other were found with a stereo calibration function call (`cv2.stereoCalibrate`).

This process was easily the most arduous of the final project, and had to be redone many times due to numerous situational lighting problems or algorithm issues.

2.4 Triangulation

After calculating the camera's Intrinsic Matrix and distortion K_c, d_c, the projector's Intrinsic Matrix and distortion K_p, d_p, and the Camera to Projector stereo rotation and translation vectors R, T, solving the triangulation problem becomes extraordinarily simple. First, we back-project the camera point with its perspective matrix $K_c[I|0]$ and distortion parameters, where I is a 3x3 identity matrix. Then, we similarly back-project the same point with the projector perspective matrix $K_p[R|t]$ and its own distortion parameters, and measure the intersection. The z value found at that intersection corresponds to the depth from the stereo pair. By performing this operation with each detected point correspondence found in the Structured Light decoded image, a point cloud can be formed.

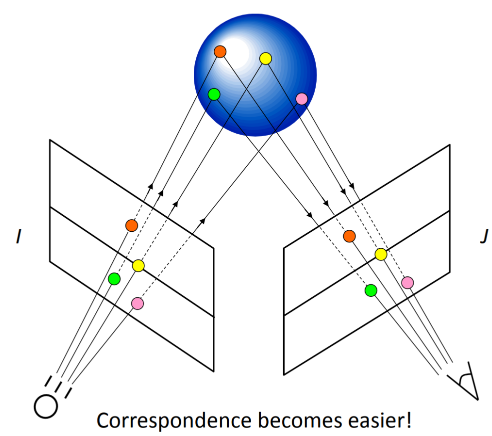

Fig. 8. Setup Picture 1

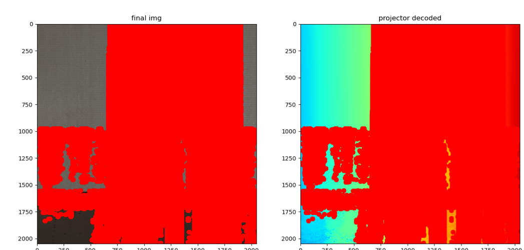

Fig. 9. Decoded Correspondence Points found in Gray code patterns on Among Us

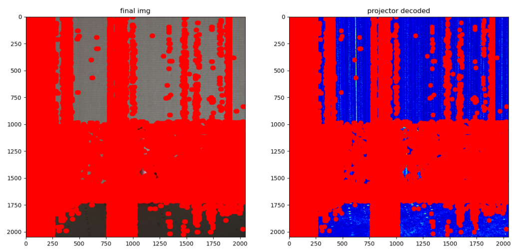

Fig. 10. Decoded Correspondence Points found in XOR patterns on Among Us

3 EXPERIMENTAL EVALUATION

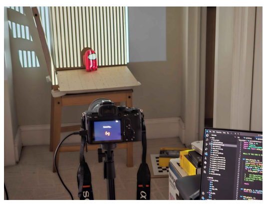

Fig. 11. Setup Picture 1

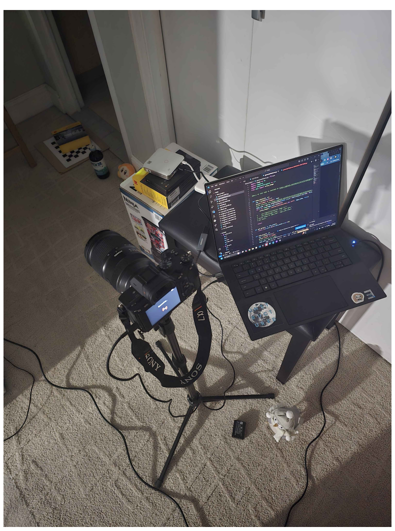

Fig. 12. Setup Picture 2

The procedure above was done with gray codes, binary codes, and XOR codes. While XOR codes were expected to produce the best result, due to projector resolution difficulties, they performed relatively weaker compared to the gray code and binary code results. However, what *can* be seen of their shape is more promising in regards to removing inter-reflection artifacts, as expected. Unfortunately, the time before the final project deadline was too tight to implement the truly robust method in Gupta et al's paper, where a combination of low-frequency gray codes of two types and XOR-2, XOR-4 codes could be used to cross-verify and produce the best result.

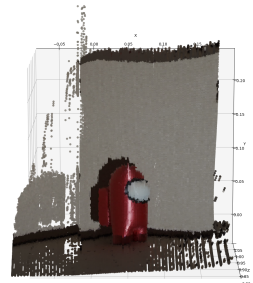

Fig. 13. Among us, projected with Binary Codes

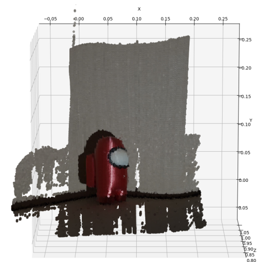

Fig. 14. Among us, projected with Gray Codes

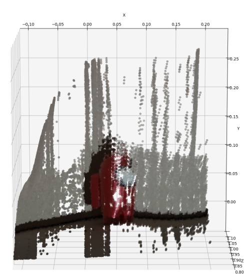

Fig. 15. Among us, projected with XOR Codes

Here, clearly, the XOR code did not overwhelmingly outperform the gray and binary codes as anticipated. However, if one examines closely, I think it handles the shiny 3D-printed material specularity better and captures the shape with less distortion in various portions. The same goes for the Waddle Dee and Kombucha Bottle below.

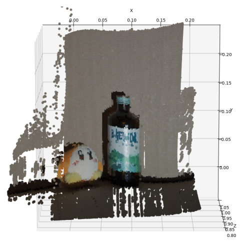

Fig. 16. Waddle Dee and Kombucha Bottle, projected with Gray Codes

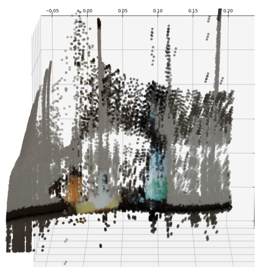

Fig. 17. Waddle Dee and Kombucha Bottle, projected with XOR Codes

4 POTENTIAL DIRECTIONS

Because camera and projector calibration took an extensive portion of time, it was difficult to attempt more ambitious Structured Light code patterns. Other codes include Parsa et al's proposed À La Carte Structured Light Patterns [5], where image data can be fed int a ZNCC (zero noramlized cross correlation) and a maximum likelihood objective function can be formed to find optimal structured light patterns for a given noisy image. Like what was mentioned above, another avenue that would have been promising to explore would be to have implemented the minimum gray stripe-width codes and XOR-2 codes, and create a self-verifying pipeline resistant to low-range and high-range light effects.

5 CONCLUSIONS

Structured light offers an efficient, relatively cheap, and accurate method for 3D scene reconstruction. This project explored various approaches to structured light, including binary codes, gray codes, and XOR codes. While XOR codes were expected to outperform others due to their robustness to global illumination issues, projector resolution limitations hindered their performance in this experiment.

While the implementations of algorithms for camera and projector calibration, along with triangulation to create a 3D point cloud were successful, the camera and projector calibration proved to be a time-consuming bottleneck. After having learned first-hand how to calibrate, I'm sure that I could go further with the other pursuits mentioned above if I had more time. Examining À La Carte structured light patterns or implementing minimum gray stripe-width codes with XOR codes could lead to a more robust system resistant to various light effects. Additionally, more complex structured light code patterns or real time would have been also wonderful to implement.

6 REFERENCES

Digital Michelangelo Project [Levoy 1999]

A Practical Approach to 3D Scanning in the Presence of Interreflections, Subsurface Scattering and Defocus (where XOR projection patterns was proposed) [Gupta et al. 2013]

A La Carte Structured Light [Mirdehghan et al. 2018]

Simple, Accurate, and Robust Projector-Camera Calibration [Moreno and Taubin 2012]

Helpful Repository on Obtaining Generating Pattern Code https://github.com/elerac/structuredlight

Taubin G., Moreno D., Lanman D. 2014 Course on 3D Structured Light Scanning [PowerPoint presentation], Brown University

Srinivasa et al., Nayar et al, 2016 16823 Structured Light 3D Scanning Lecture [PowerPoint presentation], Carnegie Mellon Univ

Gkioulekas, I. 2017 15463 Computational Photography Course Lecture 27 Structured Light [PowerPoint presentation], Carnegie Mellon Univ

Procam Repository, which implements the Projector Calibration source mentioned in paper: Simple, Accurate, and Robust Projector-Camera Calibration https://github.com/kamino410/procamcalibration

REFERENCES

Mohit Gupta, Amit Agrawal, Ashok Veeraraghavan, and Srinivasa G. Narasimhan. 2013. A Practical Approach to 3D Scanning in the Presence of Interreflections, Subsurface Scattering and Defocus. *Int. J. Comput. Vision* 102, 1–3 (March 2013), 33–55. https://doi.org/10.1007/s11263-012-0554-3

M. Levoy. 1999. The digital Michelangelo project. In *Second International Conference on 3-D Digital Imaging and Modeling (Cat. No.PR00062)*. 2–11. https://doi.org/10.1109/IM.1999.805329

Parsa Mirdehghan, Wenzheng Chen, and Kiriakos N. Kutulakos. 2018. Optimal Structured Light a la Carte. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 6248–6257. https://doi.org/10.1109/CVPR.2018.00654

Daniel Moreno and Gabriel Taubin. 2012. Simple, Accurate, and Robust Projector-Camera Calibration. In *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization Transmission*. 464–471. https://doi.org/10.1109/3DIMPVT.2012.77