

Novel Method to Reconstruct a Surface Grid Using Linear Regression Modelling



Si Chenglei, Mao Yu Di, Pang Eng Meng Wyzley and Chen Shunfa

Abstract The main focus of this report is to explore methods to laser scan objects by mapping the locations of the scanned pixels into real-world coordinates through an algorithm with the use of a projected laser plane. This report presents two different approaches: A simplified method to laser-assisted 3D scanning using Geometrical Relations and a proposed Linear Regression Models method that was based on the simplified one and compares the differences between the two. The general form of the regression models was chosen after observation of the main mathematical constructs used in the stack model and plane model of the line-based laser triangulation method. These geometrical concepts, utilizing mainly the equations and intersections of lines and planes were inspiration for the regression model. The process follows that a laser plane is generated using a laser diode and cylindrical lens and cast onto an object. After taking images of the object at different angles, image analysis is carried out using Python 2 and OpenCV2. Then, the Linear Regression Models are trained with pre-existing data before making predictions while the Geometric Relations method uses the measurements of the mechanism in the equipment to map the red pixel location detected in the images to the point cloud of the object.

Keywords 3D scanning · Linear regression models · Generating a point cloud

1 Introduction

Both physical, as well as digital 3D models are useful for a wide variety of applications such as industrial design, prototyping, and even in the production of movies and video games. [1] Our purpose is to develop a precise simple set of procedures with algorithms that when coupled with a cheap set-up can allow us to seamlessly scan objects, find the coordinates of red laser points in an image, and generate a 3D digital model of the object, which is the point cloud. For the sake of comparison,

S. Chenglei (✉) · M. Y. Di · P. E. M. Wyzley
River Valley High School, Mathematics Leaders Academy, Singapore, Singapore
e-mail: sichenglei1125@gmail.com

C. Shunfa
CRADLE, Science Centre Singapore, Singapore, Singapore

© Springer Nature Singapore Pte Ltd. 2019
H. Guo et al. (eds.), *IRC-SET 2018*,
https://doi.org/10.1007/978-981-32-9828-6_11

an overall projected cost of a possible setup we can build is about USD\$85 (Raspberry Pi 3 model B—USD\$45, Picamera—USD\$30, stepper motor—USD\$5, acrylic cutouts—USD\$5) [2] which is less than those found in the market currently at around USD\$200–\$700 for simple ones and USD\$10000–\$20000 for much advanced 3D scanners [3]. That being said, our algorithm can be coupled with any image capturing device, a projected laser plane and Python 2 to work. This will open up future avenues for anyone with a smart phone to be able to carry out 3D scanning with a simple algorithm. Currently, some methods that have been adopted include the stack model and plane model of using the line-based laser light triangulation method [4], which gave rise to our method that makes use of the plane equations and intersection of lines and planes. Our report will only focus on different ways of making use of a projected laser plane in image analysis for 3D scanning and how to map the features extracted from the analysis to actual 3D points in space.

2 Engineering Goal

To design a set of algorithms and procedures that will work in tandem with a projected laser plane and a cheap setup that will make 3D scanning less costly. This will make it easier to generate accurate point clouds and 3D models of objects.

3 Methodology and Materials

A brief description of the Geometric Relations (GR) method will be covered first and will be followed by reason for the Linear Regression Models (LRMs) method of choice. The essence of the GR method is to figure out how to map pixel locations in pictures to actual positions of points in 3D space and reconstruct a surface.

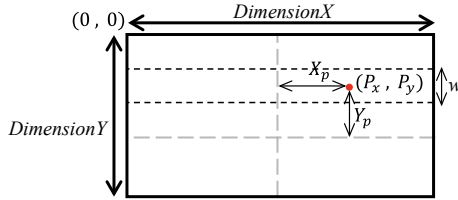
3.1 Image Analysis

Let there be a bright red point $P(P_x, P_y)$ on the image. We split the image into channels, use a sliding w by *DimensionX* window (as seen from the dotted line) and run the `cv2.minmaxLoc` function on the window, which would detect the brightest point $P(P_x, P_y)$ in the window.

Following that, we took 2 features of the image: The distance, X_p of the red pixel from the center of *DimensionX* and Y_p which is that from the center of *DimensionY*.

$$X_p = P_x - \frac{\text{DimensionX}}{2} \quad (1.1)$$

$$Y_p = \frac{DimensionY}{2} - P_y \tag{1.2}$$



Equation (1.2) is opposite as the origin in an image and the origin that the function will give its output is at the top-left hand corner.

3.2 Geometric Relations (GR) Method

Light reflects from an object and passes through the lens of the camera we use and hits the CCD Array, with length $L = 3.68$ mm. For every unique ray of light that is reflected from the object shines on a unique point on the CCD Array, a discrete pixel is formed on the image. Therefore, it is possible to introduce a dependent variable, l , that is the unknown distance in mm from the center of the CCD Array. We can then say that l is dependent on the distance in pixels of the red pixel from the center of the image, X_p which can be calculated using (1.1) because of the discrete nature of the pixel and light. This relationship is given by the following:

$$\frac{l}{L} = \frac{X_p}{DimensionX} \tag{2}$$

where L is the known length of the CCD Array, $DimensionX$ is the known total number of pixels along the x-axis of the image and X_p is the calculated distance in pixels using (1.1) between the middle of the image and the pixel in pixels. By using the concept of similar triangles (Appendix), l, X_p, fL and z can be found to have the following relationship:

$$\frac{z}{fL} = \frac{X_m}{l} \tag{3}$$

where X_m is X_p converted from pixels into mm and is also our final aim. By making l the subject in (3) and substituting into (2), we can write out X_m and Y_m . (in a similar fashion) in terms of z :

$$X_m = \frac{L}{DimensionX \cdot fL} \cdot X_p \cdot z \tag{4.1}$$

$$Y_m = \frac{H}{DimensionY \cdot fL} \cdot Y_p \cdot z \quad (4.2)$$

where H is the known width of the CCD Array, $DimensionY$ is the known total number of pixels along the y-axis of the image and Y_p is the calculated distance in pixels using (1.2) between the middle of the image and the pixel in pixels. The red laser light can be expressed as a plane with the following general equation:

$$ax + by + cz = d \quad (5)$$

where x, y and z are variable points which lie on the plane and a, b, c and d are constants that have already been determined using a real-life coordinate system. Value of z can be obtained by substituting in (4.1) and (4.2) into (5) to get the following equation:

$$a\left(\frac{L \cdot X_p}{DimensionX \cdot fL} \cdot z\right) + b\left(\frac{H \cdot Y_p}{DimensionY \cdot fL} \cdot z\right) + z = d \quad (6)$$

Rearranging (6), z can be expressed as the following:

$$z = \frac{d}{\lambda} \quad (7)$$

where

$$\lambda = a\left(\frac{L \cdot X_p}{DimensionX \cdot fL}\right) + b\left(\frac{H \cdot Y_p}{DimensionY \cdot fL}\right) + 1 \quad (8)$$

and (8) and (a, b, c, d) are the Plane's constants. Thus, the coordinates of the red pixel can be obtained for plotting inside a 3D coordinate system by using (4.1), (4.2) and (7) after taking the required measurements of our equipment.

Notably, there are 2 things that this original method fails at. Firstly, if the camera does not use a CCD Array, or uses a non-uniform, non-linear type of image sensor [5], the similar triangles property that we use for our calculations falls apart and we are unable to take the sensor length as a measurement. Secondly, the wide variety of measurements we have to take while absorbing the effect of manufacturing defects will result in inaccuracy. By looking at the expression (4) alone, there are a lot of unknowns we have to measure about the setup (a, b, d, L, H, fL) , and it may not even be possible for us to manually measure the internal hardware of the camera at times.

3.3 Linear Regression Models (LRMs) Method

In order to combat the limitations of the GR method, we proposed our Linear Regression Models(LRMs) method. Our LRMs method takes advantage of the fact that no

matter what type of image sensor is used, the red pixel location within the image is the same due to the mechanics of a camera [6]. Regardless of image sensor type, we can train the parameters in our LRMs to fit the images taken by the camera as long as we have pre-existing data. An intuitive way to think of our LRMs method is that we are finding the specifications of an arbitrary CCD Array in a camera that could have took the exact same images in some arbitrary position. Rather than taking the manual hardware measurements, we train these measurements by telling our models what to output given a certain input.

3.4 Initialising Linear Regression Models (LRMs)

Our LRMs method consists of the input matrix, the input that will be gathered using our code from a single image, in the following form:

$$I = \begin{pmatrix} x_1^1 & x_2^1 & 1 \\ \vdots & \vdots & \vdots \\ x_1^n & x_2^n & 1 \end{pmatrix} \tag{9}$$

where x_1^i denotes the X_p (see (1.1)) of the i -th red pixel detected and x_2^i denotes the Y_p of the i -th red pixel detected. Upon observation of the type of expressions that the x , y and z coordinates hold, namely (4.1), (4.2) and (7), we see that the expression for z , (7) can be manipulated to give the following form:

$$\frac{1}{z} = \frac{a}{d} \left(\frac{L}{DimensionX \cdot fL} \right) \cdot X_p + \frac{b}{d} \left(\frac{H}{DimensionY \cdot fL} \right) \cdot Y_p + \frac{1}{d} \tag{10}$$

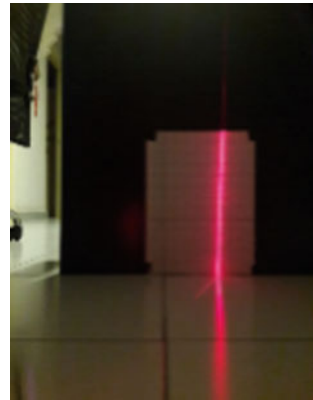
where (a, b, c, d) are the Plane's constants, L and H are the length and the height of the CCD Array, fL is the focal length of the camera and X_p and Y_p are shown in (1.1) and (1.2). Observation of (10) can tell us that $\frac{1}{z}$ can be modelled as an output of the following linear regression model:

$$Z = \frac{1}{h_\alpha(I)} \tag{11.1}$$

$$h_\alpha(I) = I\alpha^T \tag{11.2}$$

where $\alpha = (\alpha_1 \alpha_2 \alpha_0)$ is the parameters vector to be trained and $Z \in \mathfrak{R}^n$ is the output vector of dimensions, $(z_1 \dots z_n)$ and all z coordinates that correspond to the red pixels in Image with input I in (9). We can note that (11.1) looks very similar to (7) and (11.2) very similar to (10) with a certain group of constants, $\frac{a}{d} \left(\frac{L}{DimensionX \cdot fL} \right)$ and $\frac{b}{d} \left(\frac{H}{DimensionY \cdot fL} \right)$ are weights to X_p and Y_p and $\frac{1}{d}$ as an intercept term. Two

Fig. 1 Example of an image with a red pixel



other similar models are constructed based on observations of (4.1) and (4.2) and are modelled based on the following (Fig. 1):

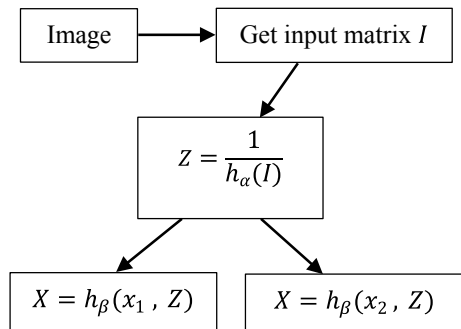
$$X = h_{\beta}(x_1, Z) = (Z_{x_1})\beta^T \tag{12.1}$$

$$Y = h_{\gamma}(x_2, Z) = (Z_{x_2})\gamma^T \tag{12.2}$$

where $\beta = (\beta_1, \beta_0)$ and $\gamma = (\gamma_1, \gamma_0)$ are parameters to be trained and $Z_{x_k} = \begin{pmatrix} x_k^1 z_1 & 1 \\ \vdots & \vdots \\ x_k^n z_n & 1 \end{pmatrix}$ for $k = 1$ and 2 . For both cases however, an intercept term is added.

Figure 2. is a simple flowchart showing our algorithm once we have trained our parameters.

Fig. 2 Flowchart of algorithm



3.5 Training Linear Regression Models (LRMs)

In prediction using regression, it is important that our training data and input follow the same distribution [7]. In this case, our images have to follow the same “distribution”. We define this “distribution” by the position of our laser plane with respect to our camera and the resolution of the image. As long as the “distribution” of our input and training data are consistent, the model is valid.

Upon setting our regression models, the parameters then have to be trained using real data. Our training data, $D = \{I, \tau = (\tau_x, \tau_y, \tau_z)\}$ where the input matrix I for the images on the left and is the collection of (x, y, z) ground-truths. For training of parameter α in (11.2), two images (Fig. 3) were taken where a piece of graph paper was fixed a distance away, $z = 60$ and $z = 30$ on a flat surface and stood perpendicular to the ground with origin on the graph paper labelled at the same (x, y) position as the camera. Next, we design a cost function to determine how well fits the truth value of 30 and 60. For this, we adopt the mean-square error (MSE) [8] as our cost function:

$$J_z(\alpha) = (I\alpha^T - \tau_z)(I\alpha^T - \tau_z)^T \tag{13.1}$$

where $\tau_z = (\frac{1}{30}, \dots, \frac{1}{30}, \frac{1}{60}, \dots, \frac{1}{60})$, $\tau_z \in \mathfrak{R}^{2n}$ is the truth vector, and there are n $\frac{1}{30}$'s and n $\frac{1}{60}$'s. We can use the normal equation method to train the parameters. In our case, it is the following:

$$\alpha = (I^T I)^{-1} I^T \tau_z \tag{13.2}$$

For training of γ in (12.2), the truth-vector τ_y was gathered by finding the difference y -coordinates of the pixels both at the bottom, B and the top of the graph paper, A (Fig. 4) and divide it by the actual length of the paper. This gives us the change in length for every pixel, $\delta = \frac{L}{L'}$. Next, we identified the of the y -coordinate of A laser at bottom of the graph paper, as well as the real-world y -coordinate of

Fig. 3 At $z = 60$ (Left); $z = 30$ (Right)

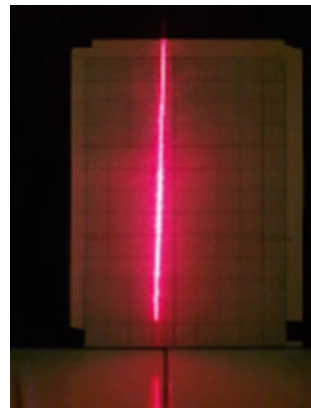
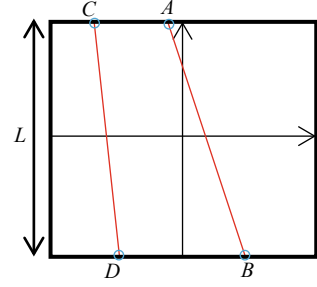


Fig. 4 Example of graph paper



that position to be our starting, s_0 . Then, our truth vector, τ_y , was generated in the following form: $t_i = s_0 + i\delta$ for $i = 0, \dots, n$. Afterwards, we use the same normal equation method similar to (12.2) minimise a new cost function:

$$J_y(\gamma) = (h_\gamma(x_2, Z) - \tau_z)(h_\gamma(x_2, Z) - \tau_z)^T \quad (14.1)$$

$$\gamma = (Z_{x_2}^T Z_{x_2})^{-1} Z_{x_2}^T \tau_y \quad (14.2)$$

where (14.1) is the MSE of our regression model, $h_\gamma(x_2, Z)$ against the truth-vector, y . For training of β in (12.1), two separate lines, (Fig. 4) AB and CD were taken at different distances. AB being the line captured at $z = 60$ and CD being the line captured at $z = 30$. Then, the difference in x -coordinates between point A and B as well as C and D were computed. Then, we repeat the procedure similar to the training of where we find the change of length for every pixel, δ_x and generate our training data in a similar fashion as γ . Then, the same normal equation method is used to minimise a similar cost function:

$$J_x(\beta) = (h_\beta(x_1, Z) - \tau_x)(h_\beta(x_1, Z) - \tau_x)^T \quad (15.1)$$

$$\beta = (Z_{x_1}^T Z_{x_1})^{-1} Z_{x_1}^T \tau_x \quad (15.2)$$

3.6 Translation and Rotation of Points

After the eventual cloud of (x, y, z) coordinates are generated for a particular image, the object was rotated on a makeshift turntable by degree (radians). A total of $\frac{2\pi}{\theta}$ images are taken this way. The points obtained will also have to be translated and rotated to fit the rotation of the turntable. This procedure consists of the following:

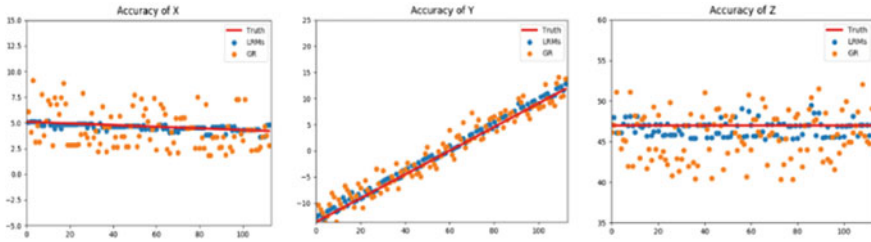


Fig. 5 Test result of X, Y and Z

$$p_{finalr} = \begin{pmatrix} \cos(\varphi) & 0 & \sin(\varphi) & dx \\ 0 & 1 & 0 & dy \\ -\sin(\varphi) & 0 & \cos(\varphi) & dz \\ 0 & 0 & 0 & 1 \end{pmatrix} p_r^T, \varphi = \theta r \tag{16}$$

where $p_r = (x \ y \ z \ 1)$ and (x, y, z) is a coordinate output from the r th Image taken.

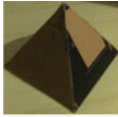


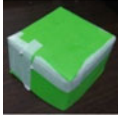

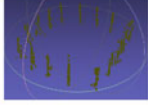
4 Results and Discussion

We tested the accuracy of our LRMs against the GR method by taking a test input at a distance of $z = 47$. The results (Fig. 5) show the accuracy of the LRMs method against the GR method. It can be seen visually that the LRMs method follows more closely to the Truth than the GR method. We then applied hypothesis testing and used a z-test to compare between the mean differences of the two methods with the Truth. The method that has a smaller difference is considered to be more accurate than the other as its output coordinates are closer to the ground truths. Since p -value is smaller than 0.05, we conclude that at 5% significance level, there is sufficient evidence that $\bar{E} < \bar{D}$. Therefore, the LRMs were more accurate in reproducing the point cloud of the object. We then tried scanning two different 3D objects and took 16 images of both input objects, the cardboard pyramid and the cube with $\theta = \frac{\pi}{8}$ in (16). Figure 6 shows the visual difference in results for the original method against the alternative method. Visually, the point cloud produced by the LRMs method follows the original structure better.

5 Conclusions and Recommendations for Future Work

In this report, we have presented 3D scanning with LRMs method along with the geometrical relations method. There is a significant improvement made from the original method as shown in the results obtained as well as in terms of usability,

Fig. 6 Table of comparison

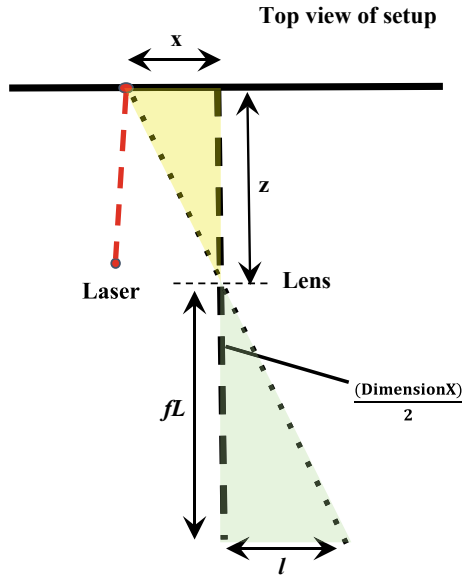
Actual Object	Point Cloud under Geometric Relations Method	Point Cloud under Linear Regression Model Method
Pyramid 		
Cube 		

where there is no need for the user to know the specifics of the camera-capturing device, which may be hard to find, unlike for the geometrical relations method where variables must be known. We have constructed an integrated setup coupled with procedures and algorithms which involves rotating the object on a platform linked to a stepper motor controlled via a Raspberry Pi 3 module. It also takes images using the Picamera controlled through a pattern of outputs to the GPIO (General Purpose Input/Output) pins.

While our results are still a far-cry from the state-of-the-art and nowhere near suitable for industrial use, the LRMs approach shows great promise given that we were able to replicate the shape of objects with just 2 images used as training data and a simple linear model.

We suggest training the LRMs with data from slopes instead of a blank graph paper so as to provide more variance of 'z's for the LRMs to be trained. Future works can also include using Anomaly detection methods first to rid the input of noise as well as varying the resolution of the image. More sophisticated models (Neural Networks, Support Vector Machines) can also be used, however, the original geometric basis will then be lost. Other methods of optimizing the cost (e.g. Gradient descent with regularization) can also be used.

Appendix: Similar triangle



References

1. Modeling, D. (2017, 23 December). Retrieved from Wikipedia The Free Encyclopedia. https://en.wikipedia.org/wiki/3D_modeling.
2. Raspberry Pi 3 Model B SBC. (2012). Retrieved from RS Components Ltd. <https://uk.rs-online.com/web/p/processor-microcontroller-development-kits/8968660/>.
3. 3D Scanners Comparison. (2018). Retrieved from Aniwaa Pte. Ltd. <https://www.aniwaa.com/comparison/3d-scanners/>.
4. Bradshaw, G. (1998/1999). *Non-contact surface geometry measurement techniques*. Ireland.
5. Golowczynski, M. (2016, 23 June). *Digital camera sensors explained*. Retrieved from what digital camera. <http://www.whatdigitalcamera.com/technical-guides/technology-guides/sensors-explained-11457>.
6. Camera. (2017). Retrieved from Wikipedia the Free Encyclopedia. <https://en.wikipedia.org/wiki/Camera#Lens>.
7. Training, Test and Validation Sets. (2017, 29 December). Retrieved from wikipedia the free encyclopedia. https://en.wikipedia.org/wiki/Training_test_and_validation_sets.
8. McCormick, C. (2014, 04 March). *Gradient descent derivation*. Retrieved from Chris McCormick. <http://mccormickml.com/2014/03/04/gradient-descent-derivation/>.