Color Point Cloud to Image Alignment

Noam Rotstein  
Technion  
Israel  
snoamr@cs.technion.ac.il

Amit Bracha  
Technion  
Israel  
Amit.bracha@cs.technion.ac.il

Ron Kimmel  
Technion  
Israel  
ron@cs.technion.ac.il

Abstract

Recognition and segmentation of objects in images enjoy the wealth of large volume of well annotated data. At the other end, when dealing with the reconstruction of geometric structures of objects from images, there is a limited amount of accurate data available for supervised learning. One type of such geometric data with insufficient amount required for deep learning is real world accurate RGB-D images. The lack of accurate RGB-D datasets is one of the obstacles in the evolution of geometric scene reconstructions from images. One solution to creating such a dataset is to capture RGB images while simultaneously using an accurate depth scanning device that assigns a depth value to each pixel. A major challenge in acquiring such ground truth data is the accurate alignment between the RGB images and the measured depth and color profiles. In this paper, we introduce a differential optimization method that aligns a colored point cloud to a given color image via iterative geometric and color matching. The proposed method enables the construction of RGB-D datasets for specific camera systems such as shape from stereo. In the suggested framework, the optimization minimizes the difference between the colors of the image pixels and the corresponding colors of the projected points to the camera plane. We assume that the colors produced by the geometric scanner camera and the color camera sensor are different and thus are characterized by different chromatic acquisition properties. We align the different color spaces while compensating for their corresponding color appearance. Under this setup, we find the transformation between the camera image and the point cloud colors by iterating between matching the relative location of the point cloud and matching colors. The successful alignments produced by the proposed method are demonstrated on both synthetic data with quantitative evaluation and real world scenes with qualitative results.

1. Introduction

In recent years, there has been tremendous research and advances in 3D data processing. This progress was not restricted to an academic level, and resulted in technologies that are being used in commercial applications in industries such as automotive and gaming.

The success of most of these products depends on the ability to capture accurate 3D data. Some of the most popular commodity depth sensing devices are based on IR or RGB structured light patterns captured from images from which surface geometry is reconstructed [62, 30, 23]. The various shape reconstruction techniques have been explored with recent attempts to apply supervised deep learning [60, 9, 63, 29, 38, 7, 31, 9, 43, 49, 58, 13, 12, 16, 55]. These methods try to solve the correspondence problem in shape from stereo, monocular depth estimation from motion, etc. Supervised learning methods rely on accurate reference datasets for geometric structures, which are difficult to obtain. A popular solution is to use synthetic datasets [44, 56]. These datasets contain reliable geometric information. However, they fail capture the real world characteristics of distortion and noise. In addition, the use of deep learning methods may require specific training data for each camera model, as the shape reconstruction algorithms are sensitive to model properties and artifacts.

One solution to acquire accurate depth is to use accurate depth measuring devices such as 3D laser scanners. In addition to geometric information, some of these devices also provide color texture. The depth and color information can be combined to produce a colored point cloud. A simple method of using such a device to acquire accurate depth values for a desired camera model requires registering the device pose relative to the camera pose. The direct approach is to place the device at a fixed and calibrated position relative to the camera. Such a setup often suffers from technical difficulties like the need for constant maintenance and calibration [17]. In addition, accurate laser scanners are usually characterized by a long scan time. Therefore, capturing only one image per scan is unsuitable for the construction of a large-scale dataset. Considering this limitation, one so-
The proposed framework operating, for simplicity, on a single point under a single modality setting [65]. (a) Treat the image as a continuous function and project the point onto the image plane. (b) Examine the color of the point and the image color at the projected point location and calculate the difference between the colors and the gradient direction of the resulting difference function. (c) Optimize the difference in the first iteration by moving the point in the gradient direction. (d) Move the point iteratively to optimize the difference. (e) Achieve convergence when the difference between the point color and the image color at the projected point location is minimized. In case of many points in our point cloud, the update is also applied to the rotation parameters.

Figure 1. The proposed framework operating, for simplicity, on a single point under a single modality setting [65]. (a) Treat the image as a continuous function and project the point onto the image plane. (b) Examine the color of the point and the image color at the projected point location and calculate the difference between the colors and the gradient direction of the resulting difference function. (c) Optimize the difference in the first iteration by moving the point in the gradient direction. (d) Move the point iteratively to optimize the difference. (e) Achieve convergence when the difference between the point color and the image color at the projected point location is minimized. In case of many points in our point cloud, the update is also applied to the rotation parameters.
One of the most fundamental tasks in this area is 3D-to-3D alignment between two point clouds. The most common solution is the Iterative closest point algorithm (ICP) [3, 8]. Many improvements and variations of this algorithm have been proposed [53, 4]. One such improvement proposed minimizing the distance between the underlying surfaces from which the point clouds are sampled [47]. Another variant proposes the first constant factor approximation for the alignment of two point clouds, under different cost functions [25]. Other papers proposed to align the colored 3D point cloud using the additional RGB color information [24, 35] or hue values [46]. Park et al. [51] addressed this problem by optimizing a joint photometric and geometric objective function that locks the alignment along both the normal direction and the tangent plane.

The task of 2D-to-2D image registration has also been widely explored and used in many applications. The most popular approach is to find the corresponding points in both images and then determine the transformation between them [10]. The two leading methods for finding the corresponding points are intensity-based methods and feature-based methods. Intensity-based alignment methods compare intensity patterns in images and image patches [50, 19]. Feature-based methods extract features in each image [40, 2, 1] and then match them with each other. The corresponding image coordinates are then derived from the corresponding matches.

An accurate approach for image registration is to use external targets placed in the scene. One choice of such targets are checkerboards coupled with subpixel detection of their corners [18]. Prior knowledge of the geometric structure of the checkerboards can be used to compute a full 3D registration using the Perspective-n-Point algorithm (PnP) [42, 37]. This algorithm estimates the camera pose given a set of 3D points and their corresponding 2D projections in the image. A method has been proposed for the calibration of stereo systems that uses the target but does not rely on feature detection [62]. This method finds the alignment between cameras by minimizing the geometric error of the reconstructed depth and the known target.

While 3D-to-3D and 2D-to-2D tasks have been thoroughly explored, we address a different challenge. We would like to fit the colors of the 3D geometry to the 2D image. Such a procedure [26], aligns 3D point clouds to overhead images using edge costs and free space costs. Ding et al. match 2D and 3D vertices without using point cloud color information [11]. Visual-based localization (VBL), is a domain that attempts to approximate camera pose relative to known 3D models [52]. The most common method is to use image feature descriptors. Features are extracted on a query image and compared to features coupled with 3D coordinates. In contrast to our task, which focuses on achieving accurate registration, most of the papers and benchmarks in this area focus on efficient and fast matching between image features and features of large-scale geometric models in \( \mathbb{R}^d \) [57, 39, 15]. Moreover, a significant effort is made to deal with changing environments of scenes [41], an issue irrelevant to the problem at hand.

There are several approaches that resemble our pose optimization method. Zhou and Koltun align multiple images to an uncolored point cloud [65]. As opposed to the proposed method, they require a few images, optimize a large number of parameters to find the colors of the points, and operate under a single-modality setup. Pulli et al. align two colored point clouds by minimizing color and range on two image planes, also under single-modality assumptions [54].

We aim to operate in a multimodal environment and compare color values from different devices. This comparison requires color manipulation, which has been approached in various forms. This problem can be viewed as the gamut mapping problem, where the task is to find a transformation of color images from input to output devices. Examples of such solutions are space-dependent gamut mappings [45, 34]. Sochen et al. show how different models of color perception, interpreted as geometries of the color space, lead to different enhanced processing schemes [59]. Much of the work in this area focuses on the perceptual relationship between colors rather than their precise values. For our concern, these solutions are not appropriate, since our problem requires the analytical quantitative comparison of their values. A classical approach to color manipulation of images is histogram equalization. Caselles et al. try to overcome the fact that histogram modification sometimes does not produce good contrast by performing it locally on connected components of the image [6]. These methods succeed in improving image contrast, but were not designed for analytical comparison of color values. Specifically, they do not consider the color relations between corresponding pixels in different images. Many papers attempt to perform color matching for value comparison when converting RGB signals to standard CIE tristimulus values. Typical methods include three-dimensional lookup tables with interpolation [22], neural networks [28], and polynomial regression models [27]. Lookup tables lack the differentiability necessary for our task. In the method we propose, the corresponding colors to be matched are computed per iteration, so training a neural network each time is not a feasible solution. On the contrary, polynomial regression models satisfy the necessary requirements for our goal. Several experiments have investigated the influence of polynomial order on the success of color transformation [21, 61]. From these experiments, it can be concluded that the higher the order used, the better the results. Practically, second order models proved to provide accurate transformations at low computational cost.
3. Rigid Alignment and Color Matching

Given a color image and a colored point cloud, the proposed method allows us to align the point cloud to the image. This alignment provides a depth estimate for each pixel based on the 3D coordinates of the point cloud. In the optimization process our model assumes that nearby pixels in an image tend to have similar colors. Moreover, color change in natural images is gradual and slow. We use this property during an optimization scheme to translate the point cloud to the correct location where its color projection onto the image plane is best aligned with the given image.

As a simple example, let us consider the case where a point cloud $P$ is correctly aligned with an image. We assume that the colors of the point cloud and the colors of the image were captured by the same device and therefore have the same color gamut. In this scenario, we examine a point $p_j$ in the point cloud. This point has 3D coordinates $p_j^{xyz} \in \mathbb{R}^3$, RGB colors $p_j^{rgb} \in \mathbb{R}^3$ and 2D coordinates in the projected point location on the image plane $p_j^{uv} \in \mathbb{R}^2$. In addition, the colors of the image in $p_j^{uv}$ are $I(p_j^{uv}) \in \mathbb{R}^3$.

Under the above assumptions, $p_j^{rgb} \approx I(p_j^{uv})$. If the coordinates $p_j^{xyz}$ are slightly misaligned $p_j^{xyz+\Delta}$, the projected point location will change $p_j^{uv+\delta}$, $\delta \in \mathbb{R}^2$ and the color of the image at the projected location will change as well $I(p_j^{uv+\delta}) \neq I(p_j^{uv})$. We would expect a discrepancy between this color and the point color $p_j^{rgb} = I(p_j^{uv+\delta})$. Since this discrepancy should approach zero if the alignment is correct, the optimization goal is to minimize it. If we create a fully differentiable pipeline, this discrepancy depends on the 3D point coordinates $p_j^{xyz+\Delta}$. For this simple case, we can use the discrepancy gradient to compute a 3D translation $\Gamma \in \mathbb{R}^3$ where the 3D point coordinates are $p_j^{xyz+\Delta+\Gamma}$. We find $\Gamma$ to be the translation at which the difference between the color of the point $p_j^{rgb}$ and the projected point $p_j^{uv+\delta+\gamma}$, $\gamma \in \mathbb{R}^2$ colors $I(p_j^{uv+\delta+\gamma})$ is reduced (see Fig 1). We use an iterative optimization procedure and progress towards $\Gamma_i$ at each iteration $i$.

Given a point cloud containing $n$ points, we repeat this calculation for each point $p_k$ and move the point cloud as a whole to minimize the total color difference of the points. To translate the points, we apply a 3D Euclidean transformation to them, which involves a translation vector $t \in \mathbb{R}^3$ and an orthogonal rotation matrix $R \in \mathbb{R}^{3 \times 3}$. Then, we apply the transformation to each of the points $R p_k^{xyz} + t$.

In this simple case, the point cloud and the image were assumed to have the same color gamut. This assumption is inaccurate because different camera sensors produce different color responses for the same scene. If the same image is captured by two different devices, there would be differences in the colors of the corresponding pixels. Each camera is characterized by its unique color properties, which result from the camera model, settings, and internal sensor features. In contrast to the previous example, for an aligned point cloud, the color of a point and the image color at the projected point location would not necessarily be the same $p_j^{rgb} \neq I(p_j^{uv})$. In order to obtain meaningful, accurate alignments, we need to compensate for such color discrepancies in the two different types of sensors we use.

We propose to solve this problem without prior color manipulation or color calibration. At each iteration, we essentially create a table of point cloud colors and a corresponding table of image colors, both of size $3 \times n$. We use these tables to find a color transformation in the RGB color space that fits between the two sets of colors. A suitable direct solution might be to find a linear transformation $C_{\text{linear}} \in \mathbb{R}^{3 \times 3}$ in the 3D color space. However, in practice, a linear mapping of the colors cannot capture the complexity of color discrepancies [21]. To this end, we find the linear transformation $C_{\text{polynomial}} \in \mathbb{R}^{3 \times 10}$ that minimizes the difference between second-order polynomials from one set of RGB colors to the other. The choice of higher order polynomial provides higher accuracy, although the marginal error between the second and higher order is small [61]. Therefore, using second order provides the required accuracy while maintaining computational efficiency.

3.1. Initialization

The goal of the proposed method is to find the translation and rotation that would align the coordinate system of a colored point cloud to that of a given camera using a single RGB image. To accomplish this, six parameters $\theta = [\theta^1, \theta^2]$ are initialized to zero. These parameters represent the three rotation DOFs that form a rotational transformation matrix $R^\theta \in \mathbb{R}^{3 \times 3}$ and three translation DOFs $t^\theta \in \mathbb{R}^3$. The six of them define the transformation $T^\theta$. We iteratively update $\theta$ during our optimization procedure until it converges. We assume a pinhole camera model, a rectilinear projection, a known intrinsic camera matrix, and assume that the optical distortions have been pre-calibrated and fixed.

3.2. Informative Differential Surface

A digital image can be viewed as a discrete function
\[
I^d(a, b) \in [0, 1]^3 \quad a, b \in \mathbb{Z}.
\] (1)

The classical image representation quantizes space into units. Each continuous coordinate belongs to a unit that has a constant color value of the nearest discrete coordinate. If we consider an image as a function of subpixel coordinates, it could be interpreted as a piecewise constant function,
\[
I^e(u, v) = I^d(r(u), r(v)) \in [0, 1]^3 \quad u, v \in \mathbb{R}
\] (2)
The function \( r() \) rounds the number to the closest integer. The gradients of this type of function are zero or undefined. Using this function does not provide the required sub-pixel gradients to perform gradient-based optimization. To obtain a differential surface, we use bilinear interpolation \([65]\). Taking \( u, v \) at sub-pixel coordinates, and \( R_{u,v}, G_{u,v}, B_{u,v} \) as the interpolated image colors at these coordinates, we obtain a smooth differential function

\[
I(u, v) = (R_{u,v}, G_{u,v}, B_{u,v}).
\]  

We use the point cloud \( P \) which has \( n \) samples, and denote that each point has six dimensions, three position coordinates and three color channel values, namely,

\[
P = [p_{XYZ} \in \mathbb{R}^{3 \times n}, p_{RGB} \in \mathbb{R}^{3 \times n}].
\]

At each optimization iteration \( i \) we perform the following steps,

1. Apply the current transformation \( T^{\theta_i} \) to the position coordinates \( p_{XYZ} \).
2. Apply a Z-buffer \( B \) to mask out occluded points.
3. Project the 3D coordinates onto the image plane using perspective projection Proj.
4. Calculate the interpolated image colors of the projected points using \( I \).

The computed color values, which all depend on \( \theta_i \), are

\[
I^P(\theta_i) = I(\text{Proj}(B(T^{\theta_i}(p_{XYZ})))) \in \mathbb{R}^{3 \times n}.
\]

### 3.3. Color Transformation

The point cloud colors \( p_{RGB} \), and the corresponding interpolated image colors \( I^P \) are obtained from two different camera sensors. Therefore, for comparing them, we would like to find the proper color relation between them. We assume that we can write each color of the image as a function \( C \) of the colors of the corresponding point in the point cloud,

\[
\{I^C = C(I^P)\}.
\]

To approximate this unknown function, we apply a second-order polynomial kernel to the colors \( \{I^P\} \),

\[
K(I^P) = K(R^P, G^P, B^P)
\]

\[
= [1, R, G, B, RGB, RB, RB^2, G^2, B^2] \in \mathbb{R}^{10 \times n}.
\]

In contrast to the framework of Hong et al. [21], we do not add the 3rd order term RGB as an additional dimension. The reason is that the experimental results have shown no advantage when this dimension is added. The alignment of the point cloud is improved in each iteration. Therefore, the table of corresponding color values between the point cloud and the image is changed and refined at each iteration. To exploit this, we find the color transformation repeatedly for each iteration \( i \). The series of coefficients \( A_i \in \mathbb{R}^{3 \times 10} \) of the polynomial terms minimizing the sum of color differences is computed by the least squares method

\[
A_i = \arg_{A} \min \|A K(I^P) - P_{RGB}\|^2. \tag{8}
\]

We apply the transformation to compute the transformed colors derived from the image

\[
I_{AK}(\theta_i) = A_i K(I^P(\theta_i)). \tag{9}
\]

This transformation can turn color values into values that exceed \([0, 1]\). This could introduce a bias to the comparison with \( P_{RGB} \), which is within bounds. We use a simple clip operation for each color value \( v \),

\[
v = \min(\max(v, 0), 1). \tag{10}
\]

However, the use of this clipping method results in the clipped values no longer depending on \( \theta_i \) and therefore, undesirably not being included in the optimization process. To fix this, the original gradient before clipping is used for the optimization process. In this way, we keep the differentiability and gradients while truncating the problematic color values. We perform this clipping operation on the transformed colors

\[
I_{RGB}(\theta_i) = \text{clip}(I_{AK}(\theta_i)). \tag{11}
\]

### 3.4. Optimization

With the above definitions, the loss function can be computed. We choose the \( L_{1,1} \) norm as our loss to reduce sensitivity to outliers, for matrix

\[
l(\theta_i) = \| I_{RGB}(\theta_i) - P_{RGB} \|_{1,1} \tag{12}
\]

\[
= \sum_{c=1}^{3} \sum_{j=1}^{n} \| I_{RGB}(\theta_i)[c, j] - P_{RGB}[c, j] \|_1.
\]

During the optimization, we want the color differences to be invariant to the color intensity, we normalize each of the color differences by their color sums. We set the sum to be invariant to the color intensity, we normalize each of the color differences.

The final loss function is,

\[
L(\theta_i) = \| I_{RGB}(\theta_i) - P_{RGB} \|_{1,1} \tag{13}
\]

\[
= \sum_{c=1}^{3} \sum_{j=1}^{n} \| I_{RGB}(\theta_i)[c, j] - P_{RGB}[c, j] \|_1.
\]

\[
I_{RGB} = d(I_{RGB}(\theta_i)). \tag{14}
\]
The operations within the norm are performed element-wise. The \( d() \) function keeps the value of the variable, but detaches its gradient calculation. We implemented and optimized the proposed algorithm with Pytorch and Pytorch Autograd and used the Adam algorithm for optimization. Since the angle and translation parameters are of different units and orders of magnitude, we initialize their learning rate accordingly. For a megapixel image, the method converges in a few hundred iterations (see Fig 2), which takes about a minute on a single GeForce GTX 1080 GPU.

4. Experimental Results

In this section we show how the proposed method performs in two scenarios. The first, involves a synthetic database. By controlling the image generation process, we can quantify the accuracy of the method. We apply known color transformations and test the results of the proposed method subject to these discrepancies. Next, we apply the proposed method to real images. We demonstrate success in aligning a FARO laser scanner point cloud with an image captured by another RGB camera. The alignment is evaluated qualitatively by projecting the intensity edges of the rendered point cloud onto the edges of the given image.

4.1. Synthetic Data

To demonstrate the proposed method a photorealistic dataset that resembles real scenes is needed. Therefore, we use the synthetic Dataset Hypersim [56]. We randomly sample 2000 images and the corresponding depth values to create a point cloud for each image. Some of the images in this dataset were poorly rendered and contain negligible color information. For example, images that contain mostly white values or images with mostly black values. Our method uses color information for alignment, so it is not expected to work on images that contain almost no information. For example, in a real scenario, we would prevent overexposure. To avoid these images in our experiment, we identify them automatically. Only for identification, we convert the color image into a grayscale image. If more than one-third of the grayscale values are below 0.05 or above 0.95 the image is disqualified.

To demonstrate the success of the proposed method, let us show how a misaligned colored point cloud successfully aligns to an image with a different color gamut. To create this colored point cloud, we use the depth values of the dataset and the RGB image color values. To miss align it, we apply a random 3D Euclidean transformation. This transform is created by uniformly sampling a translation in the interval \([0\text{mm}, 20\text{mm}]\) for each axis and a rotation in the
Figure 3. A demonstration of distortions applied to the data, and attempts to compensate with different color transformations. Top row - an image from the Hypersim dataset [56]. Middle row - the modified image after applying the effects detailed in Section 4.1 (left), the modified image after linear color transformation (middle), and the modified image after the second-order polynomial transformation (right). The bottom row shows the absolute difference between the transformed modified image and the original image. Blue corresponds to small values, while yellow corresponds to large values. While the linear transformation manages to reduce some of the color discrepancies, the second polynomial manages to eliminate most of them.

Figure 4. Cumulative normalized histograms of translation and rotation errors for synthetic data. The proposed method with second-order polynomial color alignment outperforms the one with first-order polynomial, the one with original colors, and the naive Root Sift feature-based approach.

To simulate cameras with different characteristics, we apply a number of effects to the RGB image (Fig 3). Many effects can be applied, such as effects that take into account the spatial color configuration [34] or perform retinex modifications [33, 14, 36]. To simplify the experiment, a set of elementary effects was chosen,

1. Apply random color transformation of brightness, contrast, saturation, and hue.

2. Apply gamma correction with a random gamma value.

3. Simulate different point spread functions and different sensor properties by applying a Gaussian blur to the image and adding Gaussian noise.

After this conversion, we verify that the modified image is not impaired and still contains enough color information. Similarly to the pre-color transformation, for the automatic imperative detection, we transform the modified image to a grey scale image. We count the number of pixels near extreme white or black values - below 0.025 or above 0.975. If these pixels account for more than 1% of the image and their number has increased by 250% due to the color transformation, we disqualify the modification and randomly select another one. Finally, to mimic a point cloud with a larger field of view from the image, we crop 5% from each side of the image. We note that we consider all scenes to consist of Lambertian surfaces, although there are many examples of reflections and specularities in this dataset. This approach is justified because 3D lasers do not provide accurate data for non-Lambertian surfaces, and scenes with these effects are avoided when collecting images and creating this type of dataset.

In this experiment, we test the proposed method with three setups. The first uses the second-order polynomial color transformation. The second, uses a first order color linear transformation. The third, uses the original image and point cloud colors and does not try to match the color. To compare the success of the suggested method to classical approaches, a fourth alignment method is tested. It is a naive feature-based approach that builds on a common methodology in VBL implementations. Root Sift features [40, 1] are found in both the image and a rendered image from the point cloud. Each feature in the rendered image is matched to a point from the point cloud which projection is closest to it. This point is then used to determine the coordinates of the feature in \( \mathbb{R}^3 \). After matching the feature descriptors between the images, the Ransac algorithm is applied together with the PnP algorithm to compute the Euclidean transformation.

The error is captured by two measures, translation error and rotational error. The translation error is the Euclidean distance of the original translation to the derived translation. To calculate the rotation error, the combined rotation axis is found by using the rotation error about each axis. Then, the error is calculated by computing the rotation about each axis.

We present the cumulative normalized histogram of these errors under different color transformation configurations in Fig. 4. The second-order polynomial approach clearly outperforms the linear transformation and the naive approaches that ignore color transformations. The naive feature-based approach performs worse compared to the al-
alternatives. We denote that under the Hypersim data set and using the second-order polynomial scheme, we achieve a median sub-millimeter translation error of 0.44mm and a median rotation error of 0.01°.

4.2. Real Data

To show that the proposed method works on real data, we used a FARO 3D Focus Laser Scanner and an Intel RealSense Depth Camera D435. We only use the RGB image from this camera and not its depth sensing capabilities. We perform this experiment by scanning a series of scenes. After each scan is completed, we place a camera near the scanner position. As mentioned earlier, the method relies on a coarse estimate of the camera pose, which can be calculated in various forms. We place a single checkerboard in the scene and use it to roughly estimate the camera pose relatively to the scanner. This will be the initial transformation we use before applying the proposed method. Unlike the scenario with the synthetic data, we do not have the ground truth transformation. Therefore, we estimate the success of the proposed method by visually comparing the RGB image and the rendered image from the point cloud after applying the computed transformation. Since different cameras are used, a simple color difference is not a good visual measure. Image edges are less affected by the camera characteristics and reflect whether the images are aligned correctly. Therefore, we find the edges of each image using the Canny-Haralick edge detector [20, 5, 32] and compare the edge images.

Fig 5, shows the edges extracted from both rendered and camera images. Due to the imperfect extraction of edges, not all edges in each image are detected. We can see that the edges in both images are aligned. As we can observe, this is in contrast to the edge comparison of the first misalignment of the edges. Although we cannot quantify the exact error in the real data scenario, the edge representation shows how accurate the proposed method is.

5. Conclusions

We introduced an iterative differential method that aligns a colored point cloud to an image by geometric and second-order polynomial color matching gradient-based optimization. The proposed framework introduces an algorithmic pipeline that uses the entire point cloud and image information to minimize the discrepancy between the point cloud colors properly projected onto the image colors. We explain and numerically support the advantages of using second-order polynomials for color transformation between different camera devices. We believe that the proposed concepts could facilitate and improve the creation of real 3D datasets in the future and could be applied to any camera model.
References

[1] Relja Arandjelović and Andrew Zisserman. Three things everyone should know to improve object retrieval. In 2012 IEEE Conference on Computer Vision and Pattern Recognition, pages 2911–2918. IEEE, 2012. 3, 7

[2] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In European conference on computer vision, pages 404–417. Springer, 2006. 3

[3] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In Sensor fusion IV: control paradigms and data structures, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992. 3

[4] Alexander Bronstein, Michael Bronstein, and Ron Kimmel. In the rigid kingdom. In Numerical Geometry of Non-Rigid Shapes, pages 119–135. Springer, 2008. 3

[5] John Canny. A computational approach to edge detection. IEEE Transactions on pattern analysis and machine intelligence, (6):679–698, 1986. 8

[6] V. Caselles, J.-L. Lisani, J.-M. Morel, and G. Sapiro. Shape preserving local histogram modification. IEEE Transactions on Image Processing, 8(2):220–230, 1999. 3

[7] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5410–5418, 2018. 1

[8] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. Image and vision computing, 10(3):145–155, 1992. 3

[9] Xuelian Cheng, Yiran Zhong, Mehrtash Harandi, Yuchao Dai, Xiaojun Chang, Tom Drummond, Hongdong Li, and Zongyuan Ge. Hierarchical neural architecture search for deep stereo matching. arXiv preprint arXiv:2010.13501, 2020. 1

[10] Suma Dawn, Vikas Saxena, and Bhudev Sharma. Remote sensing image registration techniques: A survey. In International Conference on Image and Signal Processing, pages 103–112. Springer, 2010. 3

[11] Min Ding, Kristian Lyngbaek, and Avideh Zakhour. Automatic registration of aerial imagery with untextured 3d lidar models. In 2008 IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8. IEEE, 2008. 3

[12] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In Proceedings of the IEEE international conference on computer vision, pages 2650–2658, 2015. 1

[13] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. arXiv preprint arXiv:1406.2283, 2014. 1

[14] Michael Elad, Ron Kimmel, Doron Shaked, and Renato Keshet. Reduced complexity retinex algorithm via the variational approach. Journal of visual communication and image representation, 14(4):369–388, 2003. 7

[15] Mengdan Feng, Sixing Hu, Marcelo H Ang, and Gim Hee Lee. 2d3d-matchnet: Learning to match keypoints across 2d image and 3d point cloud. In 2019 International Conference on Robotics and Automation (ICRA), pages 4790–4796. IEEE, 2019. 3

[16] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2002–2011, 2018. 1

[17] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. The International Journal of Robotics Research, 32(11):1231–1237, 2013. 1

[18] Andreas Geiger, Frank Moosmann, Ömer Car, and Bernhard Schuster. Automatic camera and range sensor calibration using a single shot. In 2012 IEEE International Conference on Robotics and Automation, pages 3936–3943. IEEE, 2012. 3

[19] Guy Godin, Marc Rioux, and Rejean Baribeau. Three-dimensional registration using range and intensity information. In Videometrics III, volume 2350, pages 279–290. International Society for Optics and Photonics, 1994. 3

[20] Robert M Haralick. Digital step edges from zero crossing of second directional derivatives. IEEE transactions on pattern analysis and machine intelligence, 6(1):58–68, 1984. 8

[21] Guowei Hong, M Ronnier Luo, and Peter A Rhodes. A study of digital camera colorimetric characterization based on polynomial modeling. Color Research & Application: Endorsed by Inter-Society Color Council, The Colour Group (Great Britain), Canadian Society for Color, Color Science Association of Japan, Dutch Society for the Study of Color, The Swedish Colour Centre Foundation, Colour Society of Australia, Centre Français de la Couleur, 26(1):76–84, 2001. 3, 4, 5

[22] Po-Chieh Hung. Colorimetric calibration in electronic imaging devices using a look-up-table model and interpolations. Journal of Electronic imaging, 2(1):53–61, 1993. 3

[23] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Rushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In Proceedings of the 24th annual ACM symposium on User interface software and technology, pages 559–568, 2011. 1

[24] Andrew Edie Johnson and Sing Bing Kang. Registration and integration of textured 3d data. Image and vision computing, 17(2):135–147, 1999. 3

[25] Ibrahim Jubran, Alaa Maalouf, Ron Kimmel, and Dan Feldman. Provably approximated icp. arXiv preprint arXiv:2101.03588, 2021. 3

[26] Ryan S Kaminsky, Noah Snavely, Steven M Seitz, and Richard Szeliski. Alignment of 3d point clouds to overhead images using edges and free space cost. In 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pages 63–70. IEEE, 2009. 3

[27] Henry R Kang. Color scanner calibration. Journal of imaging science and technology, 36(2):162–170, 1992. 3

[28] Henry R Kang and Peter G Anderson. Neural network applications to the color scanner and printer calibrations. Journal of Electronic Imaging, 1(2):125–133, 1992. 3
[29] Alex Kendall, Hayk Martirosyan, Saumitra Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In Proceedings of the IEEE International Conference on Computer Vision, pages 66–75, 2017.

[30] Leonid Keselman, John Iselin Woodfill, Anders Grunnet-Jepsen, and Achintya Bhowmik. Intel realsense stereoscopic depth cameras. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 1–10, 2017.

[31] Sameh Khamis, Sean Fanello, Christoph Rhemann, Adarsh Kowdle, Julien Valentin, and Shahram Izadi. Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction. In Proceedings of the European Conference on Computer Vision (ECCV), pages 573–590, 2018.

[32] Ron Kimmel and Alfred M Bruckstein. Regularized laplacian zero crossings as optimal edge integrators. International Journal of Computer Vision, 53(3):225–243, 2003.

[33] Ron Kimmel, Michael Elad, Doron Shaked, Renato Keshet, and Irwin Sobel. A variational framework for retinex. International Journal of Computer Vision, 52(1):7–23, 2003.

[34] Ron Kimmel, Doron Shaked, Michael Elad, and Irwin Sobel. Space-dependent color gamut mapping: A variational approach. IEEE Transactions on Image Processing, 14(6):796–803, 2005.

[35] Michael Korn, Martin Holzkothen, and Josef Pauli. Color supported generalized-icp. In 2014 International Conference on Computer Vision Theory and Applications (VISAPP), volume 3, pages 392–599. IEEE, 2014.

[36] Edwin H Land and John J McCann. Lightness and retinex theory. Josa, 61(1):1–11, 1971.

[37] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. International Journal of Computer Vision, 81(2):155, 2009.

[38] Zhengfa Liang, Yiliu Feng, Yulan Guo, Hengzhu Liu, Wei Chen, Linbo Qiao, Li Zhou, and Jianfeng Zhang. Learning for disparity estimation through feature constancy. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2811–2820, 2018.

[39] Liu Liu, Hongdong Li, and Yuchao Dai. Efficient global 2d-3d matching for camera localization in a large-scale 3d map. In Proceedings of the IEEE International Conference on Computer Vision, pages 2372–2381, 2017.

[40] David G Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2):91–110, 2004.

[41] Stephanie Lowry, Niko S¨underhauf, Paul Newman, John J Leonard, David Cox, Peter Corke, and Michael J Milford. Visual place recognition: A survey. IEEE Transactions on Robotics, 32(1):1–19, 2015.

[42] C-P Lu, Gregory D Hager, and Eric Mjolsness. Fast and globally convergent pose estimation from video images. IEEE transactions on pattern analysis and machine intelligence, 22(6):610–622, 2000.

[43] Wenjie Luo, Alexander G Schwing, and Raquel Urtasun. Efficient deep learning for stereo matching. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 5695–5703, 2016.

[44] Nikolaus Mayer, Eddy Ilg, Philipp Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4040–4048, 2016.

[45] John McCann. Lessons learned from mondrians applied to real images and color gamuts. In Color and imaging conference, volume 1999, pages 1–8. Society for Imaging Science and Technology, 1999.

[46] Hao Men, Biruk Gebre, and Kishore Pothmann, and Leonidas Guibas. Registration of point cloud data from a geometric optimization perspective. In Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing, pages 22–31, 2004.

[47] Ha Thai Nguyen and Minh N Do. Error analysis for image-based rendering with depth information. IEEE Transactions on Image Processing, 18(4):703–716, 2009.

[48] Jiahao Pang, Wenxiu Sun, Jimmy SJ Ren, Chengxi Yang, and Qiong Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In Proceedings of the IEEE International Conference on Computer Vision Workshops, pages 887–895, 2017.

[49] Xenophon Papademetris, Andrea P Jackowski, Robert T Schultz, Lawrence H Staib, and James S Duncan. Integrated intensity and point-feature nonrigid registration. In International Conference on Medical Image Computing and Computer-Assisted Intervention, pages 763–770. Springer, 2004.

[50] Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Colored point cloud registration revisited. In Proceedings of the IEEE international conference on computer vision, pages 143–152, 2017.

[51] Nathan Piasco, Désiré Sidibé, Cédric Demonceaux, and Valérie Gouet-Brunet. A survey on visual-based localization: On the benefit of heterogeneous data. Pattern Recognition, 74:90–109, 2018.

[52] François Pomerleau, Francis Colas, Roland Siegwart, and Stéphane Magnenat. Comparing icp variants on real-world data sets. Autonomous Robots, 34(3):133–148, 2013.

[53] Kari Pulli, Simo Piroinen, Tom Duchamp, and Werner Stuetzle. Projective surface matching of colored 3d scans. In Fifth International Conference on 3-D Digital Imaging and Modeling (3DIM’05), pages 531–538. IEEE, 2005.

[54] Siyuan Qiao, Yukun Zhu, Hartwig Adam, Alan Yuille, and Vladlen Koltun. Vip-deeplab: Learning visual perception with depth-aware video panoptic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 3997–4008, 2021.

[55] Mike Roberts and Nathan Piazza. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. arXiv 2020.

[56] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In 2011
[58] Amit Shaked and Lior Wolf. Improved stereo matching with constant highway networks and reflective confidence learning. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4641–4650, 2017.

[59] Nir Sochen, Ron Kimmel, and Ravi Malladi. A general framework for low level vision. IEEE transactions on image processing, 7(3):310–318, 1998.

[60] Vladimir Tankovich, Christian Hane, Yinda Zhang, Adarsh Kowdle, Sean Fanello, and Sofien Bouaziz. Hitnet: Hierarchical iterative tile refinement network for real-time stereo matching. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 14362–14372, 2021.

[61] Ibrahim Yilmaz, I Oztug Bildirici, Murat Yakar, and Ferruh Yildiz. Color calibration of scanners using polynomial transformation. In XXth ISPRS Congress Commission V, Istanbul, Turkey, pages 890–896, 2004.

[62] Aviad Zabatani, Vitaly Surazhsky, Erez Sperling, Sagi Ben Moshe, Ohad Menashe, David H Silver, Zachi Karni, Alexander M Bronstein, Michael M Bronstein, and Ron Kimmel. Intel® realsense™ sr300 coded light depth camera. IEEE transactions on pattern analysis and machine intelligence, 42(10):2333–2345, 2019.

[63] Jure Zbontar, Yann LeCun, et al. Stereo matching by training a convolutional neural network to compare image patches. J. Mach. Learn. Res., 17(1):2287–2318, 2016.

[64] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 185–194, 2019.

[65] Qian-Yi Zhou and Vladlen Koltun. Color map optimization for 3d reconstruction with consumer depth cameras. ACM Transactions on Graphics (TOG), 33(4):1–10, 2014.