SAKARYA UNIVERSITY JOURNAL OF COMPUTER AND INFORMATION SCIENCES
SAUCIS
http://saucis.sakarya.edu.tr/
e-ISSN: 2636-8129
Publisher: Sakarya University
Vol. 8, No. 2, 260-272, 2025
DOI: 10.35377/saucis..1637290
SAKARYA UNIVERSITY

RESEARCH ARTICLE

Feature Enhancement of TUM-RGBD Depth Images and Performance Evaluation of Gaussian Splatting-Based SplaTAM Method

Cemil Zeyveli[1]*, Ali Furkan Kamanlı[2]

[1]Department of Electrical and Electronics Engineering, Faculty of Engineering, Karabük University, Karabük, Türkiye
[2]Department of Electrical and Electronics Engineering, Faculty of Technology, Sakarya University of Applied Science, Sakarya, Türkiye

Corresponding author:
Cemil Zeyveli,
Department of Electrical and Electronics Engineering, Faculty of Engineering,
Karabük University,
Karabük, Türkiye
cemilzeyveli@karabuk.edu.tr

Article History:
Received: 11.02.2025
Revised: 02.05.2025
Accepted: 05.06.2025
Published Online: 16.06.2025

ABSTRACT
Simultaneous Localization and Mapping (SLAM) methods are used in autonomous systems to determine their locations in unknown environments and map these environments. Autonomous systems need to act autonomously without external intervention. These methods are widely used in robotics and AR/VR applications. Gaussian Splatting SLAM is a Visual SLAM method that performs mapping and localization using depth and RGB images and uses Gaussian structures for scene representation. Popular datasets such as TUM-RGBD, Replica, and Scannet++ are used in the performance evaluation and testing of the visual SLAM methods. However, the depth images in the TUM-RGBD dataset are of lower quality than other datasets. This problem negatively affects the depth data's accuracy and reduces the quality of mapping results. In this study, to increase the quality of depth images, the features of depth images were corrected using the median filter, which is the depth smoothing method, and a cleaner depth dataset was obtained. The new dataset obtained was processed using the Gaussian Splatting SLAM method, and better metric results (PSNR, SSIM, and LPIPS) were obtained compared to the original dataset. As a result, in the dataset with corrected features, an improvement of 8.08% in the first scene and 4.69% in the second scene was achieved according to metric values compared to the original dataset.

Keywords: SLAM, Gaussian Splatting, Median Filter

1. Introduction

Simultaneous Localization and Mapping (SLAM) is the problem of simultaneous mapping and positioning autonomous systems in an unknown environment [1]. SLAM is widely used in indoor and outdoor environments, underwater and aerial vehicles, and computer vision applications [2]. The solution methods for the SLAM problem are based on the work conducted by Smith and Cheeseman in 1986. The focus of this study was estimating the robot's position using sensor data [3]. In subsequent studies, algorithms have been developed to create 2D and 3D maps using various sensors [4]. SLAM methods are divided into two main groups, according to the types of sensors used: laser-based SLAM and vision-based SLAM methods [5]. In the first studies on SLAM, laser-based systems were preferred due to the cameras' low resolution and high cost. However, in recent years, the low cost of cameras, ease of integration and advantages of providing richer environmental information compared to lidar-based systems have led to the preference for monocular, stereo and RGB-D camera-based visual SLAM methods [6].

Visual SLAM methods can be divided into two main types, direct (intensive) and feature-based, depending on how the information extracted from the images is used [7]. While direct methods estimate camera motion using the intensity and color values of each pixel in the images, feature-based methods perform camera pose estimation and mapping by matching corner, line and plane features in multiple images [8].

Dense Visual SLAM methods generally use explicit scene representation. These representations include voxel grids, where the three-dimensional space is discretized by dividing it into regular small cubes, point clouds that represent the surfaces in the scene with a series of discrete point clusters, signed distance fields that provide surface information by calculating the distance of each point in space to the nearest surface and the positive or negative value of this distance, and neural scene representations created by encoding scene features through neural networks [9,10]. Although these scene representations have made progress, particularly in localization, they still face difficulties obtaining dense maps with high levels of detail [11].

To overcome these problems, the SplaTAM [10] method, based on the 3D Gaussian Splatting method, was developed in

2024. SplaTAM is a method that uses 3D Gaussians as scene representation and performs location optimization with scene geometry with these structures [10]. The Gaussian Splatting method provides better noise management. Compared to other methods, it can create dense maps by offering advantages such as continuous surface modeling, low memory consumption and fast image creation [12]. With these features, SplaTAM stands out as a strong alternative for Dense SLAM applications. Popular datasets such as TUM-RGBD are widely used in the literature to compare the performance of visual SLAM methods and evaluate them according to metric values. The TUM-RGBD dataset consists of RGB and depth images. Visual SLAM methods perform camera pose estimation and 3D reconstruction using these datasets. Therefore, the quality of RGB and depth images in the datasets is important for the method's performance. However, low-quality sensors were used to create the TUM-RGBD dataset, and the depth information contained in the depth images became weak and limited [10].

In this study, the depth images of two sub-scenes in the TUM-RGBD dataset were improved using the median filter, the depth smoothing method. The newly obtained datasets were processed using the SplaTAM method, and compared to the original datasets, 3D image quality and structural similarity improvements of 8.08% were achieved in the first scene and 4.69% in the second scene.

2. Related Work

2.1. Dense V-SLAM and Gaussian Splatting-Based SLAM Methods

Dense SLAM is a visual SLAM method that creates a high-resolution map by extracting dense information from each scene pixel. DTAM [13] is the first method that performs dense 3D reconstruction using all pixel information. However, although it provides high accuracy, it has a high computational cost. Kinect Fusion [14] performs camera tracking using the ICP algorithm and scene representation using the TSDF method. Elastic Fusion [15] is a method that optimizes memory usage and increases accuracy in large-scale environments with its loop closure feature. BAD-SLAM [16] performs dense scene representation with the Bundle Adjustment approach but has a high computational cost. Droid-SLAM [17] is a deep learning-based Dense SLAM method that provides successful results, especially in dynamic scenes.

Neural Radiance Field (NeRF) based methods have recently been developed for scene representations. iMAP [18] is a method that represents the scene with a single MLP network. However, this method suffers from catastrophic forgetting problems in large-scale scenarios. NICE-SLAM [19] and Vox-Fusion [20] provide high accuracy in large-scale scenes by optimizing implicit scene representation methods. Co-SLAM [21] combines hash-grid and one-blob encoding methods to achieve high-quality scene reconstruction. However, NeRF-based methods are limited in real-time performance due to high computational cost and memory requirements.

Gaussian Splatting-based SLAM methods use 3D Gaussian structures for scene representation. This method uses differential rasterization to perform scene modeling with high accuracy and low cost. MonoGS [22] can accurately reconstruct small and transparent objects, while GS-SLAM [23] performs optimized camera pose estimation using 3D Gaussian representations with RGB-D data. SplaTAM provides an explicit volumetric model using 3D Gaussian structures for scene representation. This method calculates the photometric losses more efficiently than other dense visual SLAM (V-SLAM) methods. It determines the gaps in the scene using a silhouette mask and thus performs a high-accuracy and high-quality mapping process. In addition, the map can be expanded by adding Gaussian components to the existing map. This feature allows SplaTAM to work effectively in large-scale environments. The SplaTAM method simplifies system setup and allows wide application areas to work with a single unposed RGB-D camera [10].

2.2. Datasets and Depth Smoothing Methods

Visual SLAM methods are evaluated using benchmark datasets such as Replica [24], ScanNet [25], ScanNet++ [26], and TUM-RGBD [27]. The ScanNet++ dataset is large and contains high-quality DSLR and iPhone RGB-D images. The Replica dataset provides photo-realistic indoor scenes with HDR renderings and dense 3D mesh structures. The TUM-RGBD dataset is generated using the Microsoft Kinect v1 sensor and is typically used only for camera tracking evaluations due to its low-quality depth data.

In computer vision systems, environmental factors and hardware limitations cause distortions in-depth images and low-resolution image problems. These problems make it difficult to represent fine details in-depth images accurately [28]. Although traditional linear filtering methods (Gaussian [29], Mean [30], Wiener [31]) are effective in eliminating problems in-depth images, they are insufficient in preserving critical features such as edges and corners [32]. Therefore, the nonlinear Median Filter [33] method is preferred to reduce distortions at the edges of object features in-depth images and improve image quality. The median filter performs this operation using the median value of the pixels, thus providing the advantage of preserving the edge and sharp structure features in the image compared to other filtering methods [34].

3. Method

The SplaTAM method is a dense SLAM method that can perform precise camera tracking and high-precision map reconstruction in challenging real-world scenes. The online optimization method uses an explicit volumetric representation approach called 3D Gaussian Splatting. This representation can process color and depth information with high accuracy and speed.

3.1. Gaussian Scene Representation

In the SplaTAM method, the scene map is represented by a set of 3D Gaussians. Each Gaussian structure consists of color (RGB), center position (μ), radius (r) and opacity (o). The Gaussian function below in Equation 1 calculates the contribution of each 3D Gaussian structure used for scene representation at a specific point in the scene.

$$\mathrm{f(x)} = o \cdot \exp\left(-\frac{\| x - \mu \|^2}{2\mathrm{r}^2}\right) \tag{1}$$

In this function, the distance of a given point in the scene of the 3D Gaussian distribution is calculated to determine the area of influence of the Gaussian distribution. The obtained distance value is normalized with the radius, and an exponential function is applied. Consequently, points close to the center of the distribution are assigned to a higher degree of influence, and distant points are assigned to a lower degree of influence.

3.2. Rendering Process of RGB, Depth and Silhouette Images

The SplaTAM method generates RGB, depth and silhouette images for each camera frame by representing the scene representation with 3D Gaussian structures. This method optimizes the scene map and camera parameters by applying a differential rendering process. When processing RGB images, 3D Gaussian structures are arranged from front to back according to their distance from the camera (depth value). Each Gaussian structure is projected onto a two-dimensional plane in the following step. Subsequently, the color and opacity contributions of each Gaussian in the pixel plane are calculated. A pixel's final color is formed by combining the colors and opacities of the Gaussian structures overlapping that pixel.

The pixel plane's color, depth, and silhouette values are obtained using the common formula of Equation 2. The coefficient v_i in the equation represents the weight value specific to the processed image. The value $f_i(p)$ indicates the opacity coefficient of the i-th Gaussian component at pixel p. The inverse opacity effect of previously processed Gaussian structures is calculated in the remaining part of the equation.

$$G(p) = \sum_{i=1}^{n} v_i f_i(p) \prod_{j=1}^{i-1} \left(1 - f_j(p)\right) \tag{2}$$

The color of a pixel is calculated using the following Equation 3. Each Gaussian component is represented by a color value ($v_i = c_i$). In the Gaussian function, the first Equation 1, the contribution of the Gaussian component to the pixel is obtained by multiplying the calculated opacity value with the color value. The inverse effect of the opacities of all Gaussian components preceding the relevant Gaussian representation in the current pixel is calculated. Therefore, the color value of each Gaussian component is weighted by the opacity effect in the current pixel and the inverse effect of the opacities of the previous structures, and the sum of the contributions of all structures determines the final color of the pixel.

$$C(p) = \sum_{i=1}^{n} c_i f_i(p) \prod_{j=1}^{i-1} \left(1 - f_j(p)\right) \tag{3}$$

The positions of the 3D Gaussian structures are calculated as their distances from the camera reference frame. A depth map representing scene geometry is then created using this information. Equation 4 is used to calculate the depth value for each pixel. This depth value is calculated by weighting the depth value ($v_i = d_i$) of the corresponding Gaussian component with its opacity value and the inverse effect of the opacities of its previous components.

$$D(p) = \sum_{i=1}^{n} d_i f_i(p) \prod_{j=1}^{i-1} \left(1 - f_j(p)\right) \tag{4}$$

A silhouette image is rendered to verify the presence of information in a pixel within the current map and its representation in the current scene map. The contribution of each Gaussian component to the silhouette is weighted by its opacity value and aliasing effects of the preceding components in Equation 5. This process identifies map gaps and underrepresented areas while optimizing camera tracking and map update processes.

$$S(p) = \sum_{i=1}^{n} f_i(p) \prod_{j=1}^{i-1} \left(1 - f_j(p)\right) \tag{5}$$

3.3. SLAM System

The SLAM system is based on 3D Gaussian representation and differential rendering methods. The system consists of camera tracking, Gaussian densification and map update steps. In the first frame, the camera position is not optimized. The camera position is determined as the starting position. This position is regarded as a reference point for the pose information that is subsequently provided. Similarly, new Gaussian structures were recreated using all pixels in the first frame. The camera position is optimized according to the current map information in the novel frames after the initial frame. Camera tracking is based on a loss function calculated with RGB and depth information. This loss function takes the difference between the rendered RGB, depth, and silhouette images and the real RGB-D images. Using the loss function, the camera position is determined.

The following loss function in Equation 6 is developed to measure the difference between the rendered depth and color values and the real values on pixels sufficiently represented in the scene map $(S(p) > 0.99)$. For each pixel, the depth difference $L_1(D(p))$ and color difference $0.5L_1(C(p))$ are calculated and summed. This total value is used to optimize the camera position. The 0.5 coefficient applied to the color difference term was determined experimentally by observing that depth difference values typically range between [0.002, 0.006] while color difference values range between [0.01, 0.03]. This scaling factor balances the different value ranges, preventing the color term from dominating the loss function during optimization [10].

$$L_t = \sum_p (S(p) > 0.99)\left(L_1(D(p)) + 0.5L_1(C(p))\right) \tag{6}$$

The Gaussian densification step is a process that provides a more detailed and dense representation of the map by adding new Gaussians to the scene for new frames. The densification mask in Equation 7 is used to identify regions in the map that are weak or underrepresented.

$$M(p) = (S(p) < 0.5) + \left(D_{\mathrm{GT}}(p) < D(p)\right)\left(L_1(D(p)) > \lambda\mathrm{MDE}\right) \tag{7}$$

If the silhouette value of the pixel is below the threshold value in the formula and the input depth value $\left(D_{\mathrm{GT}}(p)\right)$ is smaller than the rendered depth $\left(D(p)\right)$, a new Gaussian structure is added to the pixel. Additionally, if the absolute difference between the rendered and actual depth is greater than the median depth error (MDE represents the median value of all absolute differences between the rendered and actual depth values across the image), a Gaussian structure is created based on this error. Finally, in the map update step, the current 3D Gaussian map is optimized using the differential rendering process and gradient-based optimization method, starting from fixed camera poses. In this process, only the keyframe and previous frames that overlap with the current frames are considered, thereby reducing the processing load. All pixels are optimized without using a silhouette mask. Consequently, a more efficient and accurate map representation is provided. The operational system of the SLAM method is illustrated in Figure 1.

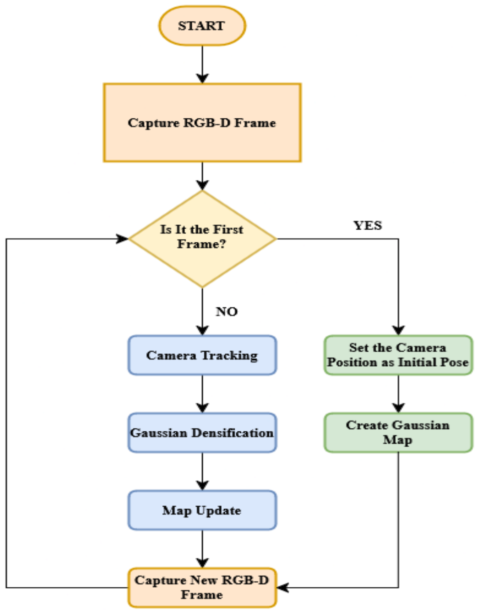

Figure 1. The Operational System of the SLAM Method

3.4. TUM-RGBD Dataset

The TUM-RGBD dataset was developed using a Microsoft Kinect v1 sensor to objectively evaluate the performance of visual odometry and SLAM systems and compare them with different methods. The dataset contains synchronized RGB and depth images and camera position information in PNG format with a resolution of 640*480. Camera position information is used in SLAM systems to measure the error rate of camera pose estimations and evaluate the generated map's accuracy.

RGB images are recorded in 8-bit format, and depth images are in 16-bit single-channel monochrome (grayscale) format. Depth images contain problems caused by low-quality sensors, ambient light and environmental conditions. This negatively affects the accuracy of depth maps and the performance of SLAM systems.

For instance, the distortions edges of the depth image of the joystick object in Figure 2 and the distortions in the regions marked in Figure 3 increase the error rate of the SplaTAM method in processing the depth map.

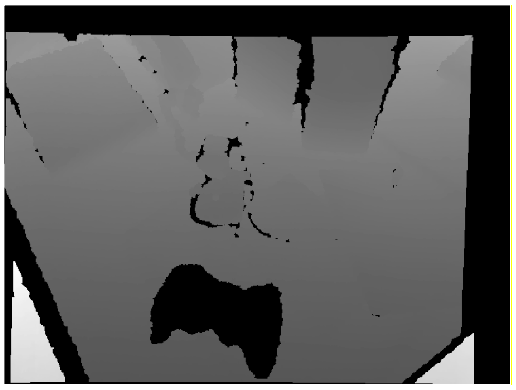
Figure 2: Distortions on the Edges of the Joystick Object in the Depth Image

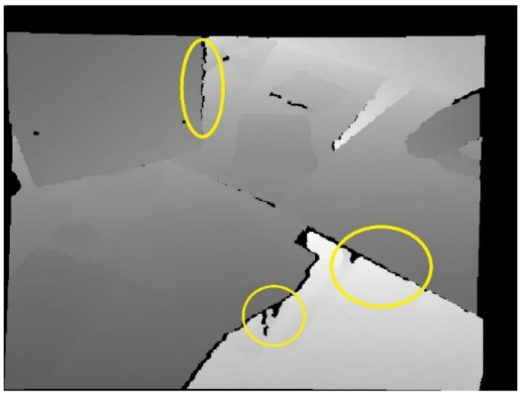
Figure 3: Edge Distortions in Depth Image

Because of these problems in the depth dataset, the TUM-RGBD dataset is generally not preferred for comparing metrical results of visual SLAM methods. To solve these problems, the median filter method improved objects' edge and corner features in-depth images.

3.5. The Median Filter

The median filter is a non-linear filter widely used in image processing. It is especially used to preserve and enhance sharp features such as edges and corners in images. Although different types of filters perform this function, the key advantage of the median filter is that it preserves the quality of the features in the image during processing and prevents blurring. Therefore, the median filter is preferred in applications where sharpness and details must be preserved.

The median filter is applied by operating all pixels in an image. Figure 4 shows the working principle of the median filter. The median filtering process consists of the following steps for each pixel in the image:

1. **Determining the Kernel Size:** A kernel size is determined around the relevant pixel to which the filter is applied.
2. **Sorting Pixel Values:** All pixel values within the kernel size are sorted from smallest to largest, and these values' median (median value) are calculated.
3. **Updating Center Pixel:** The value of the pixel located in the center of the kernel size is replaced with the calculated median value.

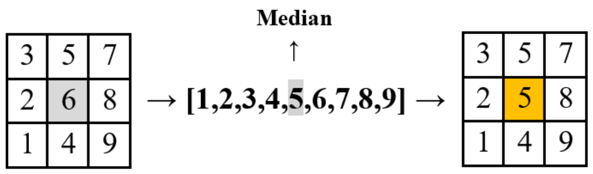

Figure 4: Median Filter Working Principle

In this study, images were processed using the median filter function of the SciPy library. This function uses size, mode and cval parameters to control the median filter.

The size parameter determines the kernel size used during the filtering. The mode parameter determines how to fill empty values when the specified kernel size is outside the image. Reflect, Constant, Nearest, Mirror and Wrap modes are mode parameters that can be used for the median filter.

The Reflect mode fills the pixel values outside the image borders in the specified kernel size by symmetrically reflecting the pixel values at the image borders (d c b a | a b c d | d c b a).

The Constant mode fills the pixel values outside the image boundaries in the specified kernel size with a constant value determined by the cval parameter (k k k k | a b c d | k k k k).

The Nearest mode fills the pixels that are outside the image borders in the specified kernel size with the value of the nearest border pixel to the relevant pixel (a a a a | a b c d | d d d d).

The Mirror mode operates similarly to the Reflect mode. The values in an inner pixel from the boundary pixel values are reflected to the empty pixels using the “mirroring” method (d c b | a b c d | c b a).

The Wrap mode places pixel values at the image boundaries by wrapping them around pixels outside the kernel size (a b c d | a b c d | a b c d).

All median filter modes explained above were applied to the depth images in the TUM-RGBD dataset, and new depth images were obtained. The kernel size used in the median filter is an important parameter affecting the filtering process's effectiveness and quality.

Determining the kernel size too large effectively removes image distortions. However, it causes the loss of fine details in the image. On the contrary, very small kernel sizes preserve edge and corner features in the depth image. However, it is insufficient to improve the features of the image. Therefore, determining the optimum kernel size is important for improving the features of in-depth images and increasing the mapping quality obtained with the SplaTAM method.

To evaluate the effect of kernel size and filter modes on the performance of depth images, kernel sizes were set to 5, 9, 11, 13, 15 and 19, respectively, and new datasets were created using each filter mode. To ensure balance and optimal results, the minimum kernel size was set to 5 and the maximum to 19.

Figure 5 shows the effect of the median filter on the distortions at the object's boundaries in the depth image by applying it to different kernel sizes.

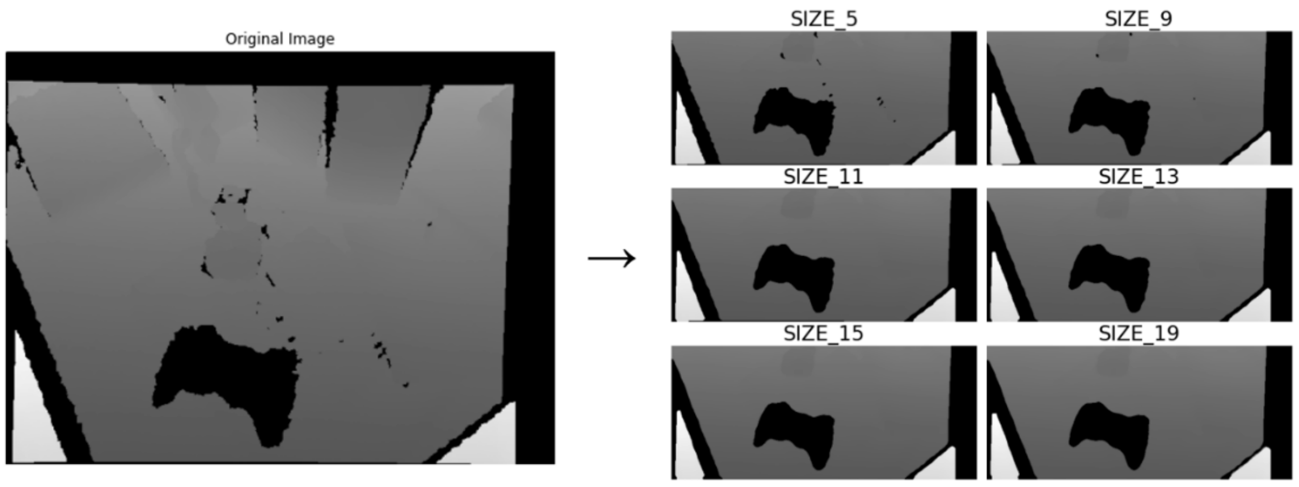

Figure 5: The Effect of The Median Filter on Depth Images at Different Kernel Sizes

Figure 6 shows the effect of the median filter on the distortions at the edge of the table in the depth image by applying it to different kernel sizes.

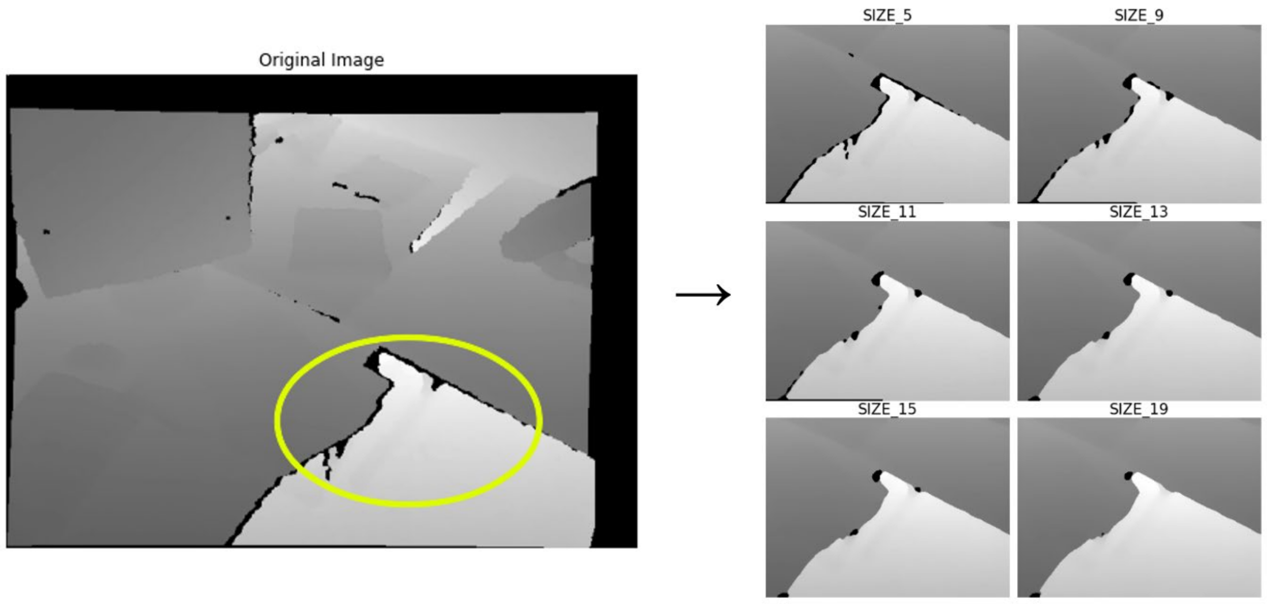

Figure 6: The Effect of The Median Filter on Depth Image Edge Distortions at Different Kernel Size

4. Result and Discussion

This section compares the results of median filtering applied to depth images of two scenes in the TUM-RGBD dataset. The effects of new depth datasets created as a result of the filtering process on the rendering quality in the SplaTAM method are examined according to Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS) metrics.

The PSNR measures the deviation of the rendered image from the original image on a pixel basis. A high PSNR value means that the rendered image is closer to the original image and that the image quality is high [35]. SSIM value provides an evaluation method based on the human visual system. It compares the structural similarities of images by considering their brightness, contrast and structural components. A high SSIM value indicates that the original image and the rendered image are structurally similar to each other [36]. LPIPS value is a metric that uses a deep learning-based method to measure the perceptual similarity between two images. It extracts features from images using a convolutional neural network model previously trained on image classification. LPIPS value takes values between zero and one, and images with values closer to zero indicate higher similarity to the original image [36].

The sub-dataset named “fr1/desk” of the TUM-RGBD dataset consists of 613 RGB images and 595 depth images. When this dataset's original depth and RGB images are rendered with the SplaTAM model, the PSNR value is 21.78, the SSIM value is 0.849, and the LPIPS value is 0.241.

Table 1 shows the metric results of the new depth datasets obtained by applying the median filter to the depth images in the "fr1/desk" sub-dataset of the TUM-RGBD dataset. The table presents the median filter results separately for each filter mode and different kernel sizes. The coloring on the table highlights the best results among the results obtained with different kernel sizes in each filter mode. Green, yellow, and orange represent the best, second-best, and third-best results, respectively.

The PSNR value of the original dataset is 21.78, and applying the median filter to the depth images resulted in a general increase in the PSNR values. The highest PSNR value is 23.54 in the Wrap mode (kernel size 13). This value shows a significant improvement compared to the original dataset. Furthermore, the second and third highest PSNR values are 23.35 in Mirror mode (kernel size 15) and 23.32 in Nearest mode (kernel size 19). If the SSIM values are compared, the SSIM result of the original dataset is 0.849, which is a similar improvement to the PSNR results. After applying the median filter, the highest SSIM value is 0.907 in Wrap mode (kernel size 13). The second highest SSIM value is 0.897 in Mirror mode (kernel size 15), and the third best value is 0.895 in Nearest mode (kernel size 19). Finally, when the LPIPS results are evaluated, the LPIPS value of the original dataset is 0.241. The best LPIPS value is 0.178 in Wrap mode (kernel size 13). The second lowest value is 0.199 in Nearest mode (kernel size 19), and the third best value is 0.200 in Mirror mode (kernel size 15).

Overall, the results show that the median filtered dataset has higher metric results than the original dataset. However, increasing the kernel size above a certain value leads to losing fine details and feature loss of in-depth images. When the table is analyzed, it is seen that although there are exceptions in some modes, the general trend is that when the kernel size is increased excessively, the metric results decrease. This shows that selecting the optimum kernel size is critical in processing depth images. Detailed results are shown in Table 1.

The processing applied to the first scene of the TUM-RGBD dataset was also applied to the second scene, "fr1/desk2". The PSNR value obtained from the original depth image of the second scene is 20.30. According to the median filter applied, the highest PSNR value is 21.30 in Mirror mode (kernel size 11). The second-best value is 21.07 in the Constant mode (kernel size 15), and the third-best value is 20.84 in the Nearest mode (kernel size 5). According to the original dataset results, the SSIM value is 0.781. When the Median filter was applied, the highest SSIM value is 0.830 in Mirror mode (kernel size 11). The second highest value is 0.813 in Constant mode (kernel size 15), and the third highest is 0.806 in Reflect mode (kernel size 15). The LPIPS value is 0.271 in the original dataset. It can be observed that the LPIPS values obtained after applying the median filter are not higher than the original result.

Compared to the first scene, the second scene of the TUM-RGBD dataset consists of RGB and depth images with a wider perspective, even though the dataset was created from the same scene environment. This suggests that the depth images in the scene "fr1/desk2" contain less detail than in the first scene. Therefore, although the median filter applied to the depth images yielded favorable results regarding PSNR and SSIM, it performed worse than the original dataset regarding the LPIPS parameter measuring perceptual similarity. This suggests that in less detailed scenes, the effect of the median filter on perceptual similarity is limited, and therefore, LPIPS values increase. Table 2 shows the effects of the median filter on PSNR, SSIM and LPIPS metrics different parameters.

The Bilateral Filter was also tested to improve the depth of images. The Bilateral Filter is a filtering method for preserving edges and fine details. The results of the Bilateral Filter applied to the "fr1/desk" sub-dataset of the TUM-RGBD dataset are shown in Table 3. When the results were examined, a decrease rather than an improvement in metric values was observed compared to the original dataset. The decrease in PSNR and SSIM values and the increase in LPIPS values as the filter window size increased indicated that this filter was not a suitable method for the study's objectives.

Table 1: The Metrical Results on TUM-RGBD "fr1/desk" Sub-dataset

Kernel Size	Metrics			Constant	Mirror	Nearest	Reflect	Wrap
Original	PSNR	↑	21.78					
	SSIM	↑	0.849	-	-	-	-	-
	LPIPS	↓	0.241					
Size 5	PSNR	↑		22.55	21.38	22.75	22.32	22.48
	SSIM	↑	-	0.869	0.844	0.882	0.856	0.869
	LPIPS	↓		0.217	0.243	0.219	0.230	0.211
Size 9	PSNR	↑		22.01	21.84	22.27	22.45	22.91
	SSIM	↑	-	0.843	0.856	0.871	0.891	0.880
	LPIPS	↓		0.257	0.246	0.222	0.205	0.218
Size 11	PSNR	↑		22.12	23.23	21.83	22.55	22.49
	SSIM	↑	-	0.856	0.884	0.857	0.862	0.865
	LPIPS	↓		0.250	0.213	0.241	0.240	0.235
Size 13	PSNR	↑		22.50	21.74	22.38	22.83	23.54
	SSIM	↑	-	0.866	0.856	0.892	0.873	0.907
	LPIPS	↓		0.245	0.255	0.207	0.218	0.178
Size 15	PSNR	↑		22.57	23.35	21.69	23.02	21.94
	SSIM	↑	-	0.877	0.897	0.858	0.876	0.881
	LPIPS	↓		0.233	0.200	0.253	0.228	0.229
Size 19	PSNR	↑		21.97	22.08	23.32	21.65	22.45
	SSIM	↑	-	0.864	0.872	0.895	0.866	0.882
	LPIPS	↓		0.246	0.244	0.199	0.241	0.220

Table 2: The Metrical Results on TUM-RGBD “fr1/desk2” Sub-dataset

Kernel Size	Metrics			Constant	Mirror	Nearest	Reflect	Wrap
Original	PSNR	↑	20.30					
	SSIM	↑	0.781	-	-	-	-	-
	LPIPS	↓	0.271					
Size 5	PSNR	↑		19.76	19.35	20.84	20.04	20.11
	SSIM	↑	-	0.765	0.761	0.797	0.800	0.780
	LPIPS	↓		0.305	0.310	0.281	0.287	0.298
Size 9	PSNR	↑		20.11	19.79	19.80	20.06	20.79
	SSIM	↑	-	0.791	0.783	0.768	0.773	0.773
	LPIPS	↓		0.301	0.314	0.305	0.302	0.287
Size 11	PSNR	↑		18.57	21.30	20.72	20.04	20.58
	SSIM	↑	-	0.734	0.830	0.787	0.799	0.777
	LPIPS	↓		0.330	0.278	0.293	0.301	0.303
Size 13	PSNR	↑		19.23	19.42	20.67	20.28	19.61
	SSIM	↑	-	0.760	0.760	0.796	0.772	0.765
	LPIPS	↓		0.329	0.336	0.308	0.313	0.323
Size 15	PSNR	↑		21.07	20.05	20.64	20.35	20.48
	SSIM	↑	-	0.813	0.770	0.775	0.806	0.790
	LPIPS	↓		0.282	0.310	0.291	0.303	0.296
Size 19	PSNR	↑		20.69	19.81	20.62	20.61	20.78
	SSIM	↑	-	0.800	0.785	0.792	0.784	0.764
	LPIPS	↓		0.300	0.315	0.304	0.319	0.314

Table 3: The Bilateral Filter Metrical Results on TUM-RGBD “fr1/desk” Sub-dataset

Metrics		Original	Size 5	Size 9	Size 11	Size 13	Size 15	Size 19
PSNR	↑	21.78	21.20	17.24	15.85	13.35	13.65	13.95
SSIM	↑	0.849	0.823	0.679	0.617	0.502	0.469	0.500
LPIPS	↓	0.241	0.269	0.415	0.469	0.563	0.565	0.511

Although the main focus of the study was to examine the effects of depth image enhancements on 3D visual quality metrics (PSNR, SSIM, and LPIPS), the effect of filtered depth images on the camera-pose estimation ATE RMSE (Absolute Trajectory Error Root Mean Square Error) of the SplaTAM method was also investigated. Tables 4 and 5 show the camera pose estimation errors processed with the Median Filter method using different kernel sizes and filter modes for the "fr1/desk" and "fr1/desk2" sub-datasets, respectively. The ATE RMSE value for the original dataset is 3.35 for the "fr1/desk" sub-dataset and 6.54 for "fr1/desk2". When examining the results, it can be observed that the Median Filter process improves visual quality and enhances camera pose estimation accuracy.

Table 4: Camera-Pose Estimation Results on TUM-RGBD “fr1/desk” Sub-dataset

Kernel Size		Constant	Mirror	Nearest	Reflect	Wrap
Original	3.35	-	-	-	-	-
Size 5	-	3.35	3.36	3.32	3.33	3.39
Size 9	-	3.50	3.29	3.30	3.33	3.33
Size 11	-	3.42	3.34	3.33	3.29	3.29
Size 13	-	3.41	3.32	3.32	3.31	3.31
Size 15	-	3.39	3.33	3.31	3.30	3.28
Size 19	-	3.31	3.28	3.29	3.32	3.30

Table 5: Camera-Pose Estimation Results on TUM-RGBD “fr1/desk2” Sub-dataset

Kernel Size		Constant	Mirror	Nearest	Reflect	Wrap
Original	6.54	-	-	-	-	-
Size 5	-	6.63	6.59	6.48	6.50	6.20
Size 9	-	6.64	6.43	6.43	6.44	6.30
Size 11	-	6.53	6.32	6.62	6.62	6.18
Size 13	-	6.64	6.33	6.50	6.58	6.07
Size 15	-	6.56	6.13	6.60	6.26	6.03
Size 19	-	6.70	6.45	6.31	6.34	6.22

The results of the positive effects of the median filter are presented in Figure 7. Figure 7 shows the depth and rendered images of two scenes of the TUM-RGBD dataset. In the first row with the joystick object, when the lowest PSNR value is compared to the best one, it is seen that there is less distortion on the edges of the game controller, and the transitions are smoother. Focusing on the ends of the papers on the edge of the table in the second row, the image with the lowest PSNR shows significant distortion on the edges and corners. In the image with the best PSNR value, the corners of the papers are clearer, and the transitions are sharp.

The images in the third row are obtained from the second scene “fr1/deks2” dataset and present a more general and wide-angle scene than the first. Comparing the images with the lowest and highest PSNR, it can be seen that there is less distortion in the objects on the table and the details on the monitors.

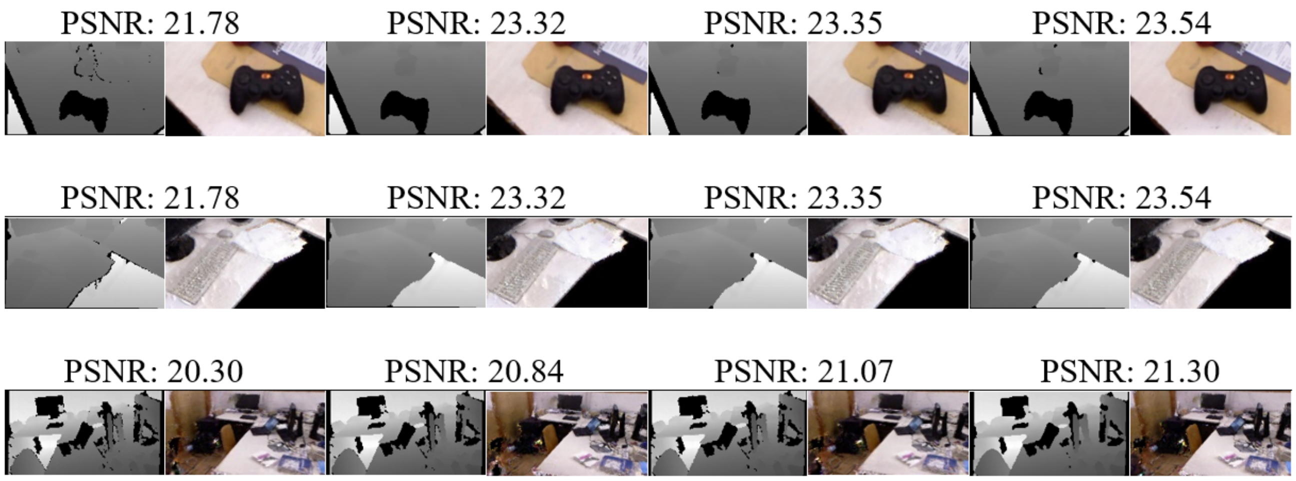

Figure 7: The Effect of Median Filter on Rendering Results

5. Conclusion

In this study, focusing on the problems in the quality of depth images in the TUM-RGBD dataset, the median filter method is used to improve the features of in-depth images, and the SplaTAM method, which uses Gaussian structure in scene representation, is used to improve 3D reconstruction performance. The TUM-RGBD, Replica and Scannet datasets are used to evaluate the performance of visual SLAM methods. Compared to Replica and Scannet datasets, which are high-quality datasets, the TUM-RGBD dataset contains lower-quality depth data with poor feature information. The original SplaTAM study [10] mentioned that the TUM-RGBD dataset consists of depth images with low image quality. Therefore, it was determined that the TUM-RGBD dataset is not preferred in studies representing visual quality. This study's contribution has demonstrated that the quality of depth images directly affects the performance of the Gaussian Splatting-based SplaTAM method. The improvements have significantly increased SplaTAM's mapping quality, thus highlighting the critical importance of depth image quality in SLAM systems.

In this study, the original depth images of two scenes in the TUM-RGBD dataset were processed with the SplaTAM method and PSNR, SSIM and LPIPS metric results were extracted. Then, a median filter is applied to the existing depth images to improve their properties. In the first scene representation, an 8.08% improvement was achieved compared with the original result, and in the second scene, a 4.69% improvement was achieved. These results show that the median filter effectively improves the weak features of in-depth images and its performance when applied to the Gaussian Splatting-based SplaTAM method.

Based on this study, in future work, the effects of depth data sets used in visual SLAM methods on 3D reconstruction performances can be examined more comprehensively using different filtering methods or deep learning-based techniques.

Moreover, since the SplaTAM method depends on depth data, generating the depth data can be improved. In this context, using deep learning-based models to generate metric depth maps from RGB images can obtain RGB and depth data using only a monochrome camera. Thus, the hardware requirements of SLAM systems can be reduced and integrated into wider application areas.

References

[1] H. Durrant-Whyte, D. Rye, and E. Nebot, "Localization of Autonomous Guided Vehicles," Robotics Research, pp. 613–625, 1996, doi: 10.1007/978-1-4471-1021-7_69.

[2] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robotics and Automation Magazine*, vol. 13, no. 2, pp. 99–108, Jun. 2006, doi: 10.1109/MRA.2006.1638022.

[3] R. C. Smith and P. Cheeseman, "On the Representation and Estimation of Spatial Uncertainty," *The international journal of Robotics Research*, vol. 5, no. 4, pp. 56–68, Dec. 1986, doi: 10.1177/027836498600500404.

[4] H. Taheri and Z. C. Xia, "SLAM; definition and evolution," *Engineering Applications of Artificial Intelligence*, vol. 97, p. 104032, Jan. 2021, doi: 10.1016/J.ENGAPPAI.2020.104032.

[5] T. J. Chong, X. J. Tang, C. H. Leng, M. Yogeswaran, O. E. Ng, and Y. Z. Chong, "Sensor Technologies and Simultaneous Localization and Mapping (SLAM)," *Procedia Computer Science*, vol. 76, pp. 174–179, Jan. 2015, doi: 10.1016/J.PROCS.2015.12.336.

[6] W. Chen *et al.*, "An Overview on Visual SLAM: From Tradition to Semantic," *Remote Sensing 2022, Vol. 14, Page 3010*, vol. 14, no. 13, p. 3010, Jun. 2022, doi: 10.3390/RS14133010.

[7] A. R. Sahili *et al.*, "A Survey of Visual SLAM Methods," *IEEE Access*, vol. 11, pp. 139643–139677, 2023, doi: 10.1109/ACCESS.2023.3341489.

[8] A. Macario Barros, M. Michel, Y. Moline, G. Corre, and F. Carrel, "A Comprehensive Survey of Visual SLAM Algorithms," *Robotics 2022, Vol. 11, Page 24*, vol. 11, no. 1, p. 24, Feb. 2022, doi: 10.3390/ROBOTICS11010024.

[9] E. Sandström, Y. Li, L. van Gool, M. R. Oswald, E. Zürich, and K. Leuven, "Point-SLAM: Dense Neural Point Cloud-based SLAM." pp. 18433–18444, 2023.

[10] N. Keetha *et al.*, "SplaTAM: Splat Track & Map 3D Gaussians for Dense RGB-D SLAM." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 21357–21366, 2024.

[11] C. Yan *et al.*, "GS-SLAM: Dense Visual SLAM with 3D Gaussian Splatting." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 19595–19604, 2024.

[12] B. Kerbl, G. Kopanas, T. Leimkuehler, and G. Drettakis, "3D Gaussian Splatting for Real-Time Radiance Field Rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, p. 14, Aug. 2023, doi: 10.1145/3592433.

[13] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2320–2327, 2011, doi: 10.1109/ICCV.2011.6126513.

[14] R. A. Newcombe *et al.*, "KinectFusion: Real-time dense surface mapping and tracking," *2011 10th IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2011*, pp. 127–136, 2011, doi: 10.1109/ISMAR.2011.6092378.

[15] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "ElasticFusion: Real-time dense SLAM and light source estimation," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1697–1716, Sep. 2016, doi: 10.1177/0278364916669237.

[16] T. Schops, T. Sattler, and M. Pollefeys, "BAD SLAM: Bundle Adjusted Direct RGB-D SLAM." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 134–144, 2019.

[17] Z. Teed and J. Deng, "DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras," *Advances in Neural Information Processing Systems*, vol. 20, pp. 16558–16569, Aug. 2021.

[18] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "iMAP: Implicit Mapping and Positioning in Real-Time," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6209–6218, 2021, doi: 10.1109/ICCV48922.2021.00617.

[19] Z. Zhu *et al.*, "NICE-SLAM: Neural Implicit Scalable Encoding for SLAM," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2022-June, pp. 12776–12786, 2022, doi: 10.1109/CVPR52688.2022.01245.

[20] X. Yang, H. Li, H. Zhai, Y. Ming, Y. Liu, and G. Zhang, "Vox-Fusion: Dense Tracking and Mapping with Voxel-

based Neural Implicit Representation," *Proceedings - 2022 IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2022*, pp. 499–507, Oct. 2022, doi: 10.1109/ISMAR55827.2022.00066.

[21] H. Wang, J. Wang, and L. Agapito, "Co-SLAM: Joint Coordinate and Sparse Parametric Encodings for Neural Real-Time SLAM," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2023-June, pp. 13293–13302, Apr. 2023, doi: 10.1109/CVPR52729.2023.01277.

[22] H. Matsuki, R. Murai, P. H. J. Kelly, and A. J. Davison, "Gaussian Splatting SLAM." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18039–18048, 2024.

[23] C. Yan *et al.*, "GS-SLAM: Dense Visual SLAM with 3D Gaussian Splatting," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition,* pp. 19595–19604, 2024.

[24] J. Straub *et al.*, "The Replica Dataset: A Digital Replica of Indoor Spaces," Jun. 2019, Accessed: Jan. 24, 2025. [Online]. Available: https://arxiv.org/abs/1906.05797v1

[25] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Niessner, "ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5828–5839, 2017.

[26] C. Yeshwanth, Y.-C. Liu, M. Nießner, and A. Dai, "ScanNet++: A High-Fidelity Dataset of 3D Indoor Scenes," *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12–22, 2023.

[27] J. Sturm, W. Burgard, and D. Cremers, "Evaluating Egomotion and Structure-from-Motion Approaches Using the TUM RGB-D Benchmark," *Proc. of the Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RJS International Conference on Intelligent Robot Systems (IROS)*, vol. 13, 2012.

[28] Q. Yang, R. Yang, J. Davis, and D. Nistér, "Spatial-depth super resolution for range images," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007, doi: 10.1109/CVPR.2007.383211.

[29] G. Deng and L. W. Cahill, "Adaptive Gaussian filter for noise reduction and edge detection," *IEEE Nuclear Science Symposium & Medical Imaging Conference*, no. pt 3, pp. 1615–1619, 1994, doi: 10.1109/NSSMIC.1993.373563.

[30] I. Pitas and A. N. Venetsanopoulos, "Nonlinear Mean Filters in Image Processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, no. 3, pp. 573–584, Jun. 1986, doi: 10.1109/TASSP.1986.1164857.

[31] M. Kazubek, "Wavelet domain image denoising by thresholding and Wiener filtering," *IEEE Signal Processing Letters*, vol. 10, no. 11, pp. 324–326, Nov. 2003, doi: 10.1109/LSP.2003.818225.

[32] P. Jain and V. Tyagi, "A survey of edge-preserving image denoising methods," *Information Systems Frontiers*, vol. 18, no. 1, pp. 159–170, Feb. 2016, doi: 10.1007/S10796-014-9527-0/TABLES/1.

[33] T. S. Huang, G. J. Yang, and G. Y. Tang, "A Fast Two-Dimensional Median Filtering Algorithm," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 27, no. 1, pp. 13–18, 1979, doi: 10.1109/TASSP.1979.1163188.

[34] A. Ravishankar, S. Anusha, H. K. Akshatha, A. Raj, S. Jahnavi, and J. Madhura, "A survey on noise reduction techniques in medical images," *Proceedings of the International Conference on Electronics, Communication and Aerospace Technology, ICECA 2017*, vol. 2017-January, pp. 385–389, 2017, doi: 10.1109/ICECA.2017.8203711.

[35] F. Artuğer and F. Özkaynak, "Görüntü Sıkıştırma Algoritmalarının Performans Analizi İçin Değerlendirme Rehberi," *International Journal of Pure and Applied Sciences*, vol. 8, no. 1, pp. 102–110, Jun. 2022, doi: 10.29132/IJPAS.1012013.

[36] S. Ghazanfari, S. Garg, P. Krishnamurthy, F. Khorrami, and A. Araujo, "R-LPIPS: An Adversarially Robust Perceptual Similarity Metric," Jul. 2023.

Authors' Contribution

Study conceptualization, design: C. Z., A. F. K. Supervision: A. F. K. Data collection and analysis: C. Z. Literature review: C. Z., A. F. K. Manuscript writing: C. Z. Final review: A.F.K.

Conflict of Interest Notice

The authors declare that there is no conflict of interest regarding the publication of this paper.

Artificial Intelligence Statement

No artificial intelligence tools were used while writing this article.