Color smoothing for RGB-D data using entropy information

Javier Navarrete[a], Diego Viejo[a], Miguel Cazorla[a,*]

[a] *Instituto de Investigación en Informática. Universidad de Alicante. Spain Tel.: 34-965-903400 ext: 2072 Fax: +34-965-903902*

Abstract

RGB-D sensors are capable of providing 3D points (depth) together with color information associated with each point. These sensors suffer from different sources of noise. With some kinds of RGB-D sensors, it is possible to pre-process the color image before assigning the color information to the 3D data. However, with other kinds of sensors that is not possible: RGB-D data must be processed directly. In this paper, we compare different approaches for noise and artifacts reduction: Gaussian, mean and bilateral filter. These methods are time consuming when managing 3D data, which can be a problem with several real time applications. We propose new methods to accelerate the whole process and improve the quality of the color information using entropy information. Entropy provides a framework for speeding up the involved methods allowing certain data not to be processed if the entropy value of that data is over or under a given threshold. The experimental results provide a way to balance the quality and the acceleration of these methods. The current results show that our methods improve both the image quality and processing time, as compared to the original methods.

Keywords: `RGB-D data, entropy, color smoothing, noise reduction`

*Corresponding author

Email addresses: `javier.navarrete@ua.es` (Javier Navarrete), `dviejo@dccia.ua.es` (Diego Viejo), `miguel.cazorla@ua.es` (Miguel Cazorla)

1. Introduction

In the last years, an increasing number of applications using three dimensional (3D) data have appeared. 3D data can be obtained from different devices: 3D lasers, stereo cameras, time-of-flight cameras, RGB-D cameras, et cetera. In general, a 3D sensor provides a set of 3D points (usually called point cloud). Each point consists of 3 coordinates (X, Y, and Z) and can also contain color information (a gray, infrared, or RGB value). Here, we are interested in sensors that can provide a point cloud with color information. It is also important to note that all the applications using this information have to deal with measurement errors that arise from the sensor's intrinsic and extrinsic parameters. This error affects both spatial and color spaces.

Reducing the impact of point position measurement error has been solved in many works in the last years. However, errors in color acquisition tend to be addressed by applying classical methods for 2D image noise reduction directly on the 2D color image for sensors that can provide it. Nevertheless, a separated color image is not available for all the 3D sensors. Our work is focused on the application of a method for correcting the color from a 3D data set directly, without using a separate color image. Furthermore, processing 3D data has an additional advantage over processing a color image: the 3D discontinuities avoid processing close points (in 2D) which belong to another object and therefore must not be processed together. Furthermore, working with 3D data implies a higher computational cost so high-efficiency methods have to be developed.

In the present paper, we perform a comprehensive review of the most typical color error sources and the methods commonly used to reduce the color error. In addition, we propose three new improvements of the original methods of handling color noise in 3D point clouds.

1.1. Motivation

This work has been motivated by the potential use of these techniques in robotics, although they can be applied to any 3D point cloud or reconstruction. Typically, the obtained point cloud has some color irregularities caused by

changes in light intensity or vignetting (less intensity at image borders). This effect is specially obvious in areas of the point cloud where the color must be uniform (such as a wall or the floor). All of these optical effects produce a non-uniform color of the point cloud. Depending on the input sensor and the amount of error in the color space, some algorithms may be affected.

Figure 1: Above non-uniform noises, below other example showing the vignetting noise.

Figure 1 shows the effects of noise on a colored 3D point cloud. In the top image, a plane in gray, which should have a uniform color, is observed. However, the color is not uniform and must be corrected. In this case, Gaussian noise is present. In the bottom image, there is a partial view of a color image taken by a RGB-D sensor in which a vignetting effect can be distinguished. Vignetting makes the colors in the center of the image different from those on the borders. The image corners (the bottom right is shown enlarged) have darker colors than the same pixels in the center of the image. These two effects should be corrected

before addressing other tasks (like robotics mapping or object recognition). The different types of noises are explained in Section 3.

1.2. Proposal

In the present paper, we apply three commonly used methods, two linear and one non-linear, to smooth the color of a 3D point cloud. We use these methods to mitigate the effects of color changes among neighboring points in the cloud. The application of these methods on 3D data directly, instead of using the corresponding 2D image, has the advantage that the color of foreground objects is not affected by the background and vice versa. As linear smoothing methods we use the classical mean and Gaussian filters, and as non-linear method, the bilateral filter [21].

We compare the performance of the three filters for different color noise sources and propose three new approaches that can be used to improve any smoothing method. The first method we propose uses an entropy measure to detect discontinuities in the color space, thus speeding up the existing methods by discarding points from the cloud that should not be processed. In our second approach, entropy is used again to obtain the optimum value of the size of the neighborhood that affects the point smoothing calculations. In this way, the color smoothing process is improved. Our third approach is based on the optimal selection of the radius. It assumes that close points will have a similar entropy, so it is possible to reduce the search for an optimal radius. Although all the points are processed, the radius is adapted, allowing to process fewer or more points, depending on the entropy. We perform several experiments for color smoothing in 3D point clouds and draw our conclusions applying each of the three methods described above for each noise source and comparing the results with or without applying our improvements. Although our methods could be applied, with prior adaptation, to a normal 2D image, some RGB-D sensors do not provide a 2D image but only 3D data with color information. Usually, the RGB-D sensor is related with some low cost sensors, like the Kinect camera. But other sensors are included in the term, like some 3D lasers with

color information. In some of these sensors, we are not able to access the color image. For that reason, we have adapted our methods to work with 3D data.

The study and the proposed methods follow a soft computing approach. We search for a method which aims to use heuristic information to speed up a given solution (smoothing). Without that heuristic, the problem could not be boarded under time constraints due to the fact that 3D data is usually much larger than normal 2D images. The obtained solution is not optimal, but suffienciently accurate to the problem.

The rest of the paper is structured as follows: after reviewing the state of the art in Section 2, Section 3 lists the noise sources that we use for testing the smoothing methods response in our experiments. After that, in Section 4 we show the quality index used for comparing quantitatively the results of different methods. Section 5 describes the smoothing filters that are reviewed and compared in this paper. Our proposals for smoothing improvement are described in Section 6 and Section 7 presents the experimental procedure and the results obtained with the different smoothing methods, for different noise sources, and a study of the effect of applying our approaches to each one. Section 8 presents experiments and results with real images, and finally, in Section 9, we present our conclusions and future work.

2. Related work

One area that has benefited from 3D sensors has been mobile robotics. Computing the egomotion, which is defined as the movement made by the robot or the camera between two consecutive poses, is one of the key issues in this area. Several research works have addressed this problem using visual odometry [24] sometimes also called pose registration [25][29], which is considered a good estimation for egomotion in the last years. These works can be considered a good starting point for automatic map building and for solving the Simultaneous Location And Mapping (SLAM) [14] problem.

Using color information along with the 3D points may improve the results

of pose registration methods [20][30]. For a review of one particular type of RGB-D camera, the Kinect, see [19] and [32]. In these works, the 3D data processing is enhanced by using color information. However, errors that occur in capturing the color of the 3D points and which can affect the final result are not addressed.

Recently, several works have developed a mapping or SLAM system using a RGB-D camera. In [16], the use of a RGB-D sensor is proposed to resolve 3D visual SLAM in real-time, but they do not solve the problem addressed in this paper. Something similar happens with [17]. Therefore, our interest is not in solving the matching problem, which can be done using 3D registration, but in improving the color representation of the data to use better information when applying a given method.

To improve the color capturing errors in 3D point sets we adapt two of the classic image filters, namely the mean and Gaussian filters, to the three dimensional space. We also use the non-linear bilateral filtering [28]. It was originally proposed to perform an edge-preserving 2D image smoothing by combining domain and range filtering. Many applications use this filter to improve image representation. [26] and [15] use bilateral filtering to obtain an enhanced image from flash and no-flash image pairs. Tone management is achieved in [1], also using the bilateral filter. Other studies have been carried out for other sensors, such as CMOS [2], but for the best of our knowledge none of them uses a RGB-D sensor. Other advanced color smoothing methods, like BM3D [13] or wavelet have been reported. However, their direct application to 3D data has not been described.

A complete review of bilateral filter variations can be found in [3]. In [4] a comprehensive classification of noise sources is performed. These authors also propose an improvement for the bilateral filter. This improvement consists in finding the pair of parameters which minimizes the mean squared error (MSE) by mean of an exhaustive (and costly) searching process through parameters space. Moreover they use the strength of edges measured by a Laplacian of Gaussian operator [21] for pixel classification. Nevertheless, in this paper, by

using entropy we propose a simpler and lighter way to study the neighborhood as performed in the bilateral filter.

In [8], the authors use standard deviation as a method to adapt the radius parameter of a bilateral filter that is used for smoothing a depth image. A depth image is organized like a 2D image, a x-y coordinate matrix, where pixels values are distances measured from the point of view, usually the position of the sensor. Applying the bilateral filter, they correct the position errors in the point-cloud. The main problem with this approach is that the depth image is required. Some 3D sensors such as stereo cameras or kinect-like cameras provide this kind of image, but other devices like range lasers do not. In our case, we work directly over unorganized point clouds so we do not require a depth image. In this way, our method can be applied to data sets obtained by any kind of 3D sensor.

Other works place the emphasis on refining depth images like [5], [6] and [7], but none of them works with color information. All of them are focus on correct position, and not on the color of the 3D points.

3. Sources of noise

When using RGB-D cameras, images suffer from the same errors as any conventional camera. The first type of noise in digital images is the Gaussian noise, which usually appears from the acquisition phase. It comes from poor illumination, temperature and/or from the electronic sensor. It is an additive noise. In a given image $I(x, y)$, the additive noise can be defined as:

$$I(x, y) = I'(x, y) + g(x, y) \tag{1}$$

where $I'(x, y)$ is the original image without noise and $g(x, y)$ is the noise. When g is defined as a Gaussian (or normal) probability density function (PDF), then the noise is considered Gaussian. The PDF of a Gaussian random variable z is

given by:

$$f(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(z-\mu)^2}{2\sigma^2}} \tag{2}$$

where (μ, σ) are the mean and standard deviation of the distribution. For the rest of the paper we assume an additive white Gaussian noise.

Another kind of noise is produced by errors in the data transmission: salt and pepper noise, also called intensity spikes or speckle. The corrupted pixels are either set at the maximum or minimum value, while unaffected pixels always remain unchanged. We can model this noise with a given probability ρ (with $0 \leq \rho \leq 1$). A pixel will have the maximum value (white) with a probability $\rho/2$ and have the minimum value (black) with the same probability. This kind of noise is due to defects in the camera sensor or in the data transmission and it usually affects only a small number of pixels.

The last source of color noise is vignetting. It consists in the reduction of an image's brightness or saturation at the periphery compared to the image center. Vignetting effect can occur due to different causes. Some arise from the optical properties of camera lenses. Other sources of vignetting are geometric in nature. For example, light arriving at oblique angles to the optical axis may be partially obstructed by the field stop or lens rim [33]. Several computer vision applications could be affected when vignetting is present. For instance, object recognition or image mosaicing (map reconstruction when working with 3D data).

4. Color quality measurement

To compare the results of applying the different filters we need a measurement method to analyze their efficacy. We have to clarify that, given a real image, it is not possible to establish a quantitative metric that indicates how good is a filter for removing noise. Usually, noise or artifacts are added to an original image in order to create an image for testing the different methods. Then, metrics compare the original image with the result of applying a

noise removal method on the testing image. In this way, we can compare the performance of different methods.

Several form of quality measurement have been proposed in the literature. An example is the Peak Signal-to-Noise Ratio (PSNR) which measures the signal fidelity of a distorted image compared to a reference. It is based on the measure of the Mean Squared Error (MSE) which is based on the square differences of color values. It indicates the similarity between two colored point sets.

For a RGB color, the PSNR is obtained as an average of the three values, one for each color band. This is a simple way to compare color values, but it does not correlate highly with human perception [18]. Furthermore, mean squared error has the disadvantage of being affected by outliers. This is a result of the squaring of each term, which effectively weights large errors more heavily than small ones.

In this work, we use a more perceptual-like metric, the Universal Image Quality Index (UQI) [31]. This index establishes a value of changes in the color of the point cloud. Instead of using traditional error summation methods, the proposed index is designed by modeling any image distortion as a combination of three factors: loss of correlation, luminance distortion, and contrast distortion. The loss of correlation is related to the degree of linear correlation between the color values of the point clouds. The index is defined in the range $[-1, 1]$ between two colored point clouds, being 1 when the color of the point clouds are identical, pixel to pixel, and -1 when the difference is maximum.

Let be $p = \{p_i\}, i = 1...N$ be a set of N 3D points. Each point has 3 coordinates (x, y, z) and 3 colors (r, g, b): $p_i = \{p_{x_i}, p_{y_i}, p_{z_i}, p_{r_i}, p_{g_i}, p_{b_i}\}$. For a given band, for instance the red band, UQI is defined as:

$$UQI_r = \frac{4\sigma_{p_r q_r}\bar{p_r}\bar{q_r}}{(\sigma^2_{p_r} + \sigma^2_{q_r})(\bar{p_r}^2 + \bar{q_r}^2)} \tag{3}$$

where

$$\bar{p_r} = \frac{1}{N}\sum_{i=1}^{N} p_{r_i}, \quad \bar{q_r} = \frac{1}{N}\sum_{i=1}^{N} q_{r_i}, \tag{4}$$

are the mean color values for the two point clouds and

$$\sigma_{p_r} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(p_{r_i} - \bar{p_r})^2}, \quad \sigma_{q_r} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(q_{r_i} - \bar{q_r})^2}, \tag{5}$$

are the color standard deviation. The covariance is defined as:

$$\sigma_{p_r q_r} = \frac{1}{N-1}\sum_{i=1}^{N}(p_{r_i} - \bar{p_r})(q_{r_i} - \bar{q_r}). \tag{6}$$

The final UQI value is the mean of the UQI from the three bands. We use the UQI measure for all our experiments as a way to compare the results of applying different smoothing techniques. This index does not provide a way to measure the noise in an image but a way to compare two images by a combination of three factors: loss of correlation, luminance distortion, and contrast distortion. This is important because it is impossible to know if a given filter is better than another by comparing the results. We need ground-truth data (i.e. image without noise).

5. Smoothing methods for RGB-D sensors

For the rest of the paper, we assume that the noise in this kind of sensors is independent of spatial coordinates and that it is uncorrelated with the image itself. All the filters described in this section use a neighborhood around one point to define the filter. Given a point p_i, the neighborhood is defined by a sphere of radius R and the set of points inside the sphere is $S = \{p_j\}, j = 1...M$.

First, we define linear methods. In linear filters, each point inside the neighborhood is weighted by a given value w_j. All the linear filters use the same procedure and equations. The only difference between them is the weight value. For the red band, the new color of the point is calculated as follows:

$$r_i = \frac{\sum_{p_j \in S} r_j w_j}{\sum_{p_j \in S} w_j}. \tag{7}$$

For the other bands, the procedure is the same, but using g or b for green or blue bands respectively. The final point color is the combination of the obtained band values.

The first filter is the mean filter. It is defined as the average value from the points inside the neighborhood. The sum of the weights is always 1.0. So the former equation is simplified as:

$$r_i = \frac{1}{M} \sum_{p_j \in S} r_j. \tag{8}$$

For the rest of the paper, when we refer to this filter we assume that the neighborhood is spherical.

The second filter is the Gaussian filter [21], which is defined by a discrete approximation of a PDF with mean 0 and σ standard deviation. The weights are set accordingly to the distribution. This filter gives more weight to the points closest to the center of the kernel, decreasing that weight as the distance to that center increases, using a Gaussian PDF.

Another kind of filter is the non-linear filters which do not use (7). In the present work we have chosen the bilateral filter because it provides good quality results. It is an edge-preserving and color-noise-reducing smoothing filter. It could also be applied to reduce noise in the 3D point coordinates [23], but here we apply it to achieve color smoothing. The color value at each point is replaced by an averaged weight of color values from points in the neighborhood S. This weight can be based on a Gaussian distribution. The weights are computed simultaneously in Euclidean and color-space distance. The combination of the two Gaussian distributions provides a way for the filter to preserve color edges. If a given point in the neighborhood has a different color from the one evaluated, its value is not included in the final color, as the Gaussian functions will be low enough. The same occurs when the distance from the evaluated point and the given point is high enough.

Its complexity is $O(N \times M)$ where N is the number of 3D points and M is the number of points in the neighborhood. The bilateral algorithm is more time consuming than the two previous methods, but it provides better smoothing

results. For that reason, we have developed the current research to speed up this method. The algorithm is shown in Algorithm 1. We note that for the point selection (line 2 of this algorithm) any method could be used: sequential, random, uniform sampling, and et cetera. There is no advantage using one method or another.

Algorithm 1 Bilateral filtering algorithm. This algorithm is defined for the R component and must be applied in a similar way for G and B band. So p_{r_i} is the R component of point p_i. $G_\sigma()$ functions are PDF with mean 0 and standard deviation σ.

Require: P3D: 3D point cloud.

$P3D = \{p_i\}, i = 1...N, p_i = \{p_{x_i}, p_{y_i}, p_{z_i}, p_{r_i}, p_{g_i}, p_{b_i}\}$

Require: σ_s: spatial Gaussian standard deviation.

Require: σ_c: color Gaussian standard deviation.

Require: r defines the radius for a given neighborhood.

1: $P3Dout = \emptyset$
2: **for** each point p_i in P3D **do**
3: Select a set of points $S = \{p_j\}, j = 1...M$ around p_i with a Euclidean distance $d(p_i, p_j) < r$.
4: $w = 0$
5: $acum = 0$
6: **for** each point p_j in S **do**
7: $acum = acum + r_j * G_{\sigma_s}(||p_i - p_j||)G_{\sigma_c}(|r_i - r_j|)$
8: $w = w + G_{\sigma_s}(||p_i - p_j||)G_{\sigma_c}(|r_i - r_j|)$
9: **end for**
10: $q = p_i$
11: $q_{r_i} = acum/w$
12: $P3Dout = P3Dout \cup q$
13: **end for**
14: **return** P3Dout: 3D point cloud color smoothed.

We have selected these three filters because their structure facilitates apply-

ing the improvements explained in the next section. For the original implementation methods and for the experiments, we have selected a mask size of 0.11 cm for the three methods (mean, Gaussian, and bilateral). This value has been selected empirically. Note that the greater the size, the greater is the smoothness obtained. However, the blur effect and processing time are also increased. The standard deviations σ_s and σ_c are also chosen empirically.

6. Improvements for smoothing methods

All the methods presented in the previous section use a determined neighborhood to apply the smoothing. In this section, we propose several improvements that can be used along with the smoothing filters to speed up the whole process or to obtain better results in terms of color error. Applying one or other of the given improvements will depend on the final application and the processing requirements. All the methods presented below can be applied to any of the previous smoothing methods.

6.1. Discrimination of points using entropy and a fixed radius

This improvement is based on the use of the entropy to discriminate whether a point should be evaluated or discarded in the smoothing process. The goal of this improvement is to maintain the level of detail in color edges, while smoothing is applied in areas with less color variation. The speed-up achieved depends on the environment. Environments with high presence of textured surfaces will be processed faster. The main advantage of this technique is that it allows details in high textured areas to be preserved, while allowing to process only uniform areas.

Entropy is a measure of the uncertainty of a random variable [12]. Let X be a discrete random variable with probability mass function $P(x) = Pr\{X = x\}, x \in X$. The entropy $H(X)$ of a discrete random variable X is defined by

$$H(X) = -\sum_{x \in X} p(x) \log_2 p(x) \tag{9}$$

For color images, given that c_j is the color for 3D point p_j we redefine entropy as:

$$H_c(X) = -\sum_{c_j \in S} p(c_j) \log_2 p(c_j) \tag{10}$$

where $P(c_j)$ is the probability of the color $c_j = \{r_j, g_j, b_j\}$ in the given neighborhood S. We use the mean value from the three bands R, G and B as the final entropy value. We define a given threshold and only process the points with an entropy value below this threshold, indicating that the area in S is somewhat uniform. If the entropy is high enough, all the points will be processed. The computation of the entropy adds a small time cost, but this is counterbalanced by benefits in the entire cloud processing. The right-hand side picture in Figure 2 shows (marked in red) the points not processed due to high entropy.

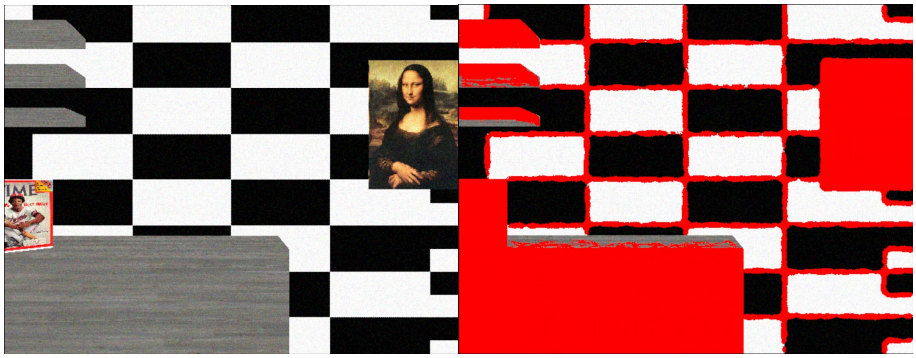

Figure 2: Left: original image. Right: in red, points not processed due to entropy threshold.

6.2. Neighborhood variation using entropy

The goal of this improvement is to find the optimum window size for the neighborhood of the smoothing filter. Formally, we seek a radius r^* for a given point p_i:

$$r^* = \arg\min_{r \in \mathbb{R}} E(r, e_t, p_i) = ||e_c - e_t||^2 \tag{11}$$

where e_t is the entropy threshold and e_c is the calculated entropy using the color information of the points with a distance to p_i lower than r. In the experimental section, for synthetic data we use different values for the entropy threshold, with $0 < e_t \leq 8$. For the real data, we fix $e_t = 4$ because this value is the one yielding better results. Figure 3 shows an example of the search of the optimal radius process. The process begins with a radius of 0.11m and entropy threshold of 4. It iterates reducing the radius until the calculated entropy is below the threshold. This occurs when radius is equal to 0.05m.

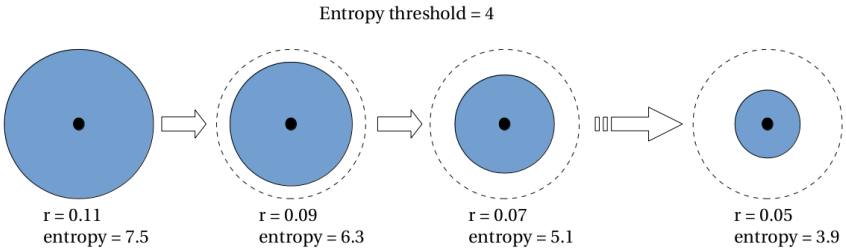

Figure 3: Sample of optimal radius process calculation.

To this end, a search for the optimal radius is performed starting at a maximum radius value R_{max}. For each possible radius value, a homogeneity test based on entropy of the region with the radius is performed. If this entropy value is below a threshold, the search process ends returning the current radius value to be used in the smoothing filter. Otherwise, the radius value is decremented for the next iteration. The process continues until the desired entropy threshold is reached or the radius value is decreased until a minimum value previously set. In our implementation, the radius is decreased in $R_{max}/20$ at each iteration until the condition (the entropy value is below the threshold) or the minimum radius (in our case $R_{max}/20$) are reached. The use of a variable radius allows to select a bigger radius when processing a uniform color area, and a smaller one when the area is close to a color edge.

6.3. Batch processing with optimal radius

The previous approach obtains good results at the cost of much calculation due to the task of searching the optimum radius for a given entropy at each point of the image. Our third approach aims to achieve equally good results in a much lower execution time. This new approach starts computing, for a given point p_i, the optimum radius R^p_{opt} using the method described in section 6.2. Nevertheless, instead of repeating this work for all the points in the image we can simplify the process by assuming that the computed R^p_{opt} for p_i does not differ greatly from the values we will obtain if we compute their optimum radius. Using this assumption, we can use R^p_{opt} to be the optimum radius for a subset of points in the neighborhood of p_i. In this way, we need to determine the size of this neighborhood around p_i. Let R_{proc} be the radius for the subset of points around p_i, its value must be the highest possible ensuring that we reduce the use of data outside the analyzed area during the selection of R^p_{opt}, that is determined by R_{max}. Thus, we propose computing R_{proc} using the following equation:

$$R_{proc} = \begin{cases} R_{max} - R_{opt}, & \text{if } R_{opt} < R_{max}/2 \\ R_{max}/2, & \text{otherwise.} \end{cases} \tag{12}$$

which ensures that when the area around p_i that is being analyzed presents a high entropy value, in which the computation of R^p_{opt} produces small values compared to R_{max}, R_{proc} is set to the maximum value that ensures not using points further than R_{max} for processing the neighborhood of p_i using R^p_{opt}. Nevertheless, the smaller the entropy area we find, the bigger R^p_{opt} is obtained and thus less points are included in the neighborhood of p_i. In the case of a uniform area when the likelihood of obtaining similar values of R_{opt} for all the points in the area is high, we have that $R^p_{opt} \simeq R_{max}$ and thus no points are included in the neighborhood which reduces drastically the efficiency of this method. In order to solve this effect we include here an optimality relaxation term by allowing some of the most distant points in the neighborhood of p_i to be smoothed using points further than R_{max} when $R_{opt} \geq R_{max}/2$. Thus,

we achieve a considerable reduction of the computation time with a minimum impact in the overall quality.

With this technique, we assume that close points have a similar processing radius, so if an optimal radius is selected, neighboring points will have a similar optimal radius. At the same time, we reduce the execution time because it is not necessary to compute the optimal radius for all the points (the part more time consuming of the previous method). Moreover, radius adaptation to selected entropy parameter avoid unnecessary large radius processing, saving more time.

Finally, once R^p_{opt} is computed and R_{proc} is determined by (12), all the points in the neighborhood of p_i are smoothed using R^p_{opt}, as the optimum radius for each of them, and are then marked as visited. Then, a new point p_j is randomly selected among the points not yet visited and the process described here starts again. This is repeated until all the points in the image are marked as visited.

7. Results and Discussion

In our experiments, we generate 3D synthetic data sets. These sets are the ground truth data we use in the calculation of the Universal Image Quality Index (UQI) used to compare the results of applying the different filters. Then, from each set p, three new sets are generated from each type of noise considered in this paper: additive Gaussian $G(p)$, Salt and Pepper $S\&P(p)$ and Vignetting $Vig(p)$. Therefore, for each experiment, the original data set (the ground truth) and the result of applying a filter to the noise distorted sets are available. Both sets are used in the computation of the UQI value for the given filter. All the smoothing filters presented in Section 5 have been implemented and adapted to 3D data, processing each color band individually. We have selected several scenes with different depths and texture to provide a balanced scene.

Although our methods can be applied to any RGB-D sensor, we have focused on the low cost RGB-D, as the primary sensor. This kind of sensor has a resolution of 640×480 (used for the experiments) or 320×240 points, depending on the camera.

We have conducted the experiments using the Point Cloud Library (PCL [27]) in C++. We have also used a software able to generate synthetic data. The synthetic data are generated simulating the way the Kinect gets their 3D data. It is not just adding a Z coordinate to each pixel but also transforming the x, y, d information to a 3D point using the Kinect library. Our experiments were performed using a dual CPU Intel Xeon X5660 with 6 cores at 2.8GHz.

In this section, we describe several experiments to analyze the behavior of the different smoothing filters with respect to the different types of noise. We also show the advantage of using the improvements proposed in this paper. The experiments are structured as follows: for each method proposed, we have conducted experiments for different types of noise and for each one we show the effect of the different noise filters.

For the bilateral filter and for all the experiments and types of noises, we have chosen empirically $\sigma_c = 5.0$, as it provides the best UQI values. The different radii are set as the σ_s parameter. For the mean and Gaussian filters, the different radii are also set as the size of the kernel.

First, we show the results for the filters without any of the improvements presented in this paper so they can then be compared with our improvements. Figure 4 shows the UQI value of the bilateral filter for the three types of noises. Figure 5 shows the same for the Gaussian and the mean filter.

Analyzing these results, the Gaussian filter provides better UQI results for Gaussian noise, which is a reasonable result. For salt-and-pepper noise, we can conclude that the bilateral filter provides better UQI value. Finally, for the vignetting noise, the bilateral filter again provides better UQI values. Generally, using different radii, the larger the radius is, the lower the UQI obtained. However, for the Gaussian noise with the bilateral and Gaussian filters, the maximum UQI value is achieved with a radius of 0.03 cms.

7.1. Results for discrimination of points using entropy and a fixed radius

Here we analyze the performance of the first improvement proposed in Section 6. The number of processed points is independent of the smoothing filter

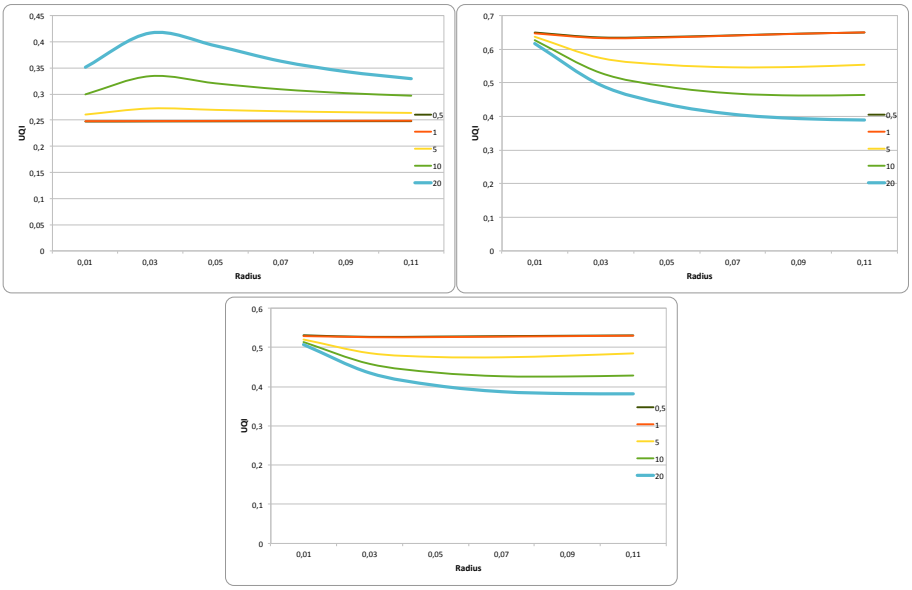

Figure 4: UQI value for the bilateral filter with Gaussian noise (top-left), Salt&Pepper (top-right) and Vignetting (bottom) noise, with respect to different radii and σ_c parameter.

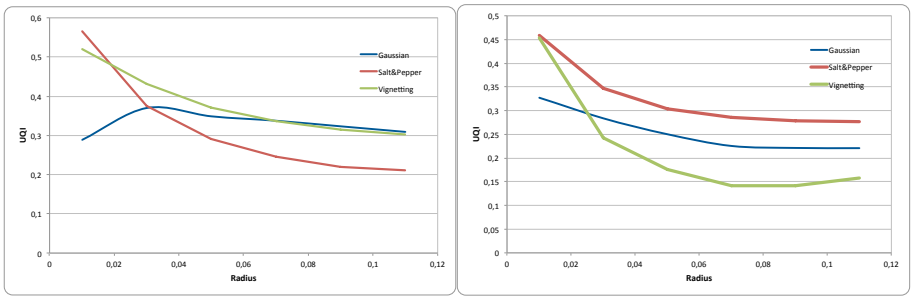

Figure 5: UQI value obtained from the Gaussian (left) and mean filter (right) for the three types of noise, with respect to different radii.

used, as the entropy computation does not depend on the posterior filter. We first show the number of processed points for each noise and then the UQI value obtained for each filter. The number of processed points for each filter can be found in Figure 6. Figure 7 shows the UQI values obtained after applying the three filters for the Gaussian noise. Figures 8 and 9 show the UQI values for the salt&pepper and vignetting noise, respectively.

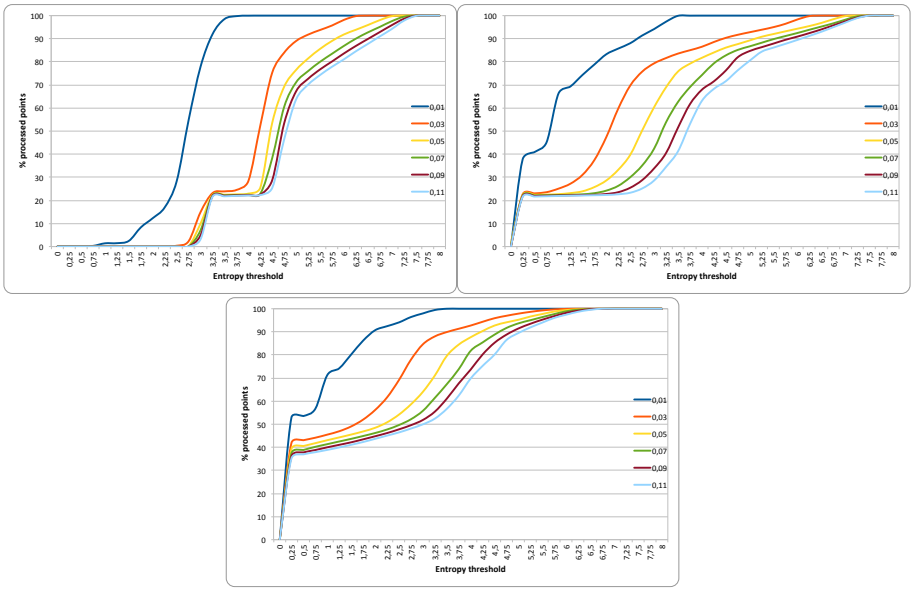

Figure 6: Number of processed points with respect to the entropy threshold and the different noises for the discrimination with entropy method. Top-left: Gaussian noise; top-right: salt&pepper noise; bottom: vignetting noise.

First, we analyze the number of processed points with respect to the entropy value. The higher the entropy value, the fewer are the points processed. This is an expected behavior as more points will be discarded when raising the entropy value. Regarding the radius value, it is also expected that when the radius is increased more points will be discarded.

For the Gaussian noise, the Gaussian filter together with the entropy-based discrimination of points is the one which provides the best UQI, even better than the original implementation of the filter. The highest UQI value is obtained for an entropy value of 5 with radius between 0.05 and 0.11. With this entropy value the number of processed points is 50% of the original points, which means that we have obtained a 2x acceleration and also an improvement of the UQI value. The mean filter also improves the original implementation for the same entropy value. Finally, the bilateral filter reaches the same UQI value as the original implementation with entropy value of 5. Due to the fact that the Gaussian

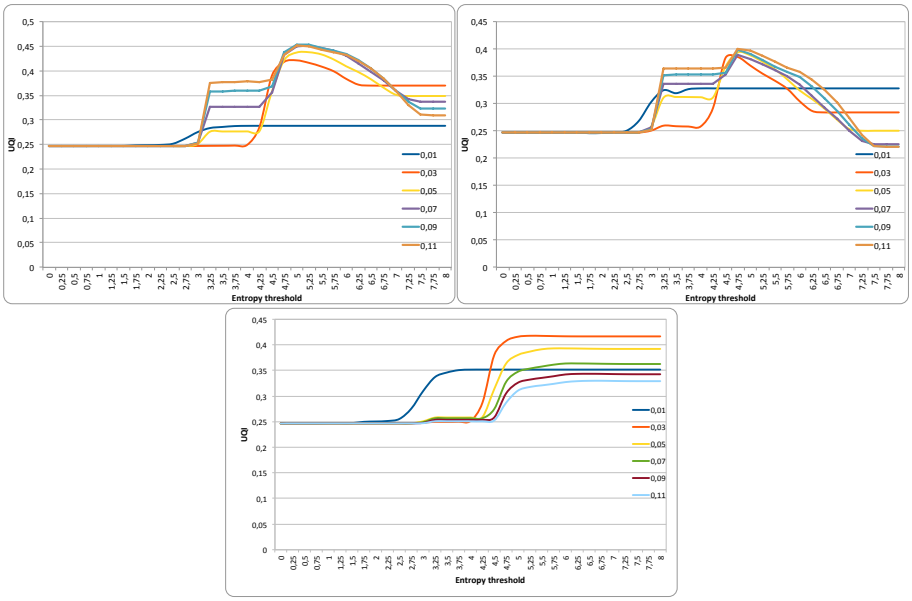

Figure 7: Type of noise: Gaussian. UQI value for the Gaussian (top-left), mean (top-right) and bilateral (bottom) filter for the discrimination with entropy method.

filter is faster and simpler than the bilateral filter, it is preferred for this kind of noise. For radius values of 0.05 and above, there is no significant difference in the number of processed points and therefore a value of 0.05 will be preferred as it will have fewer points to process.

For the Salt&Pepper and the vignetting noises, the three filters have similar UQI values. All of them provide higher UQI value with low entropy values. With these UQI values, the number of processed points is low. Again, the Gaussian or mean filters are preferred instead of the bilateral one, as they are simpler and faster.

7.2. Results for neighborhood variation using entropy

In this case, there is no speed-up in the process but an improvement of the quality of the UQI value. This method processes all the points, and for this reason the number of processed points is not shown in the following figures as it is always 100%. The resulting UQI values after applying the three filters

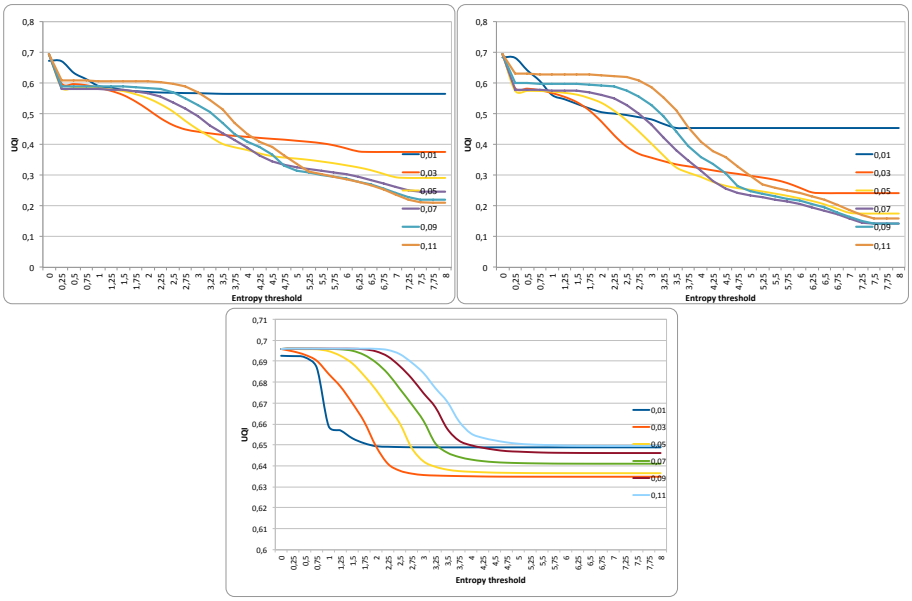

Figure 8: Type of noise: salt&pepper. UQI value for the Gaussian (top-left), mean (top-right) and bilateral (bottom) filter for the discrimination with entropy method.

for different entropy thresholds and using different radii can be observed in Figures 10, 11 and 12 for Gaussian, salt&pepper and vignetting noises, respectively. The experimentation was conducted using several initial radii, as, although this method seeks the optimal radius, different initialization could provide different results (as, in fact, happens).

This improvement is focused on the quality of the color information. It is the most time-consuming method. Regarding the Gaussian noise, the Gaussian and mean filters provide higher UQI values than the original implementations. However, the bilateral filter does not achieve better UQI values for any and, therefore, this improvement is not good for this filter. Entropy values of 4.5 (for the Gaussian filter) and 4.0 (for the mean one) have the highest UQI value. For all the filters, larger radii provide better UQI values, so it is preferable to select an initial high radius since the method is able to reduce it.

The selection of this initial high radius is guided by the sensor device. It must ensure a minimum number of points (10 points is empirically a good number).

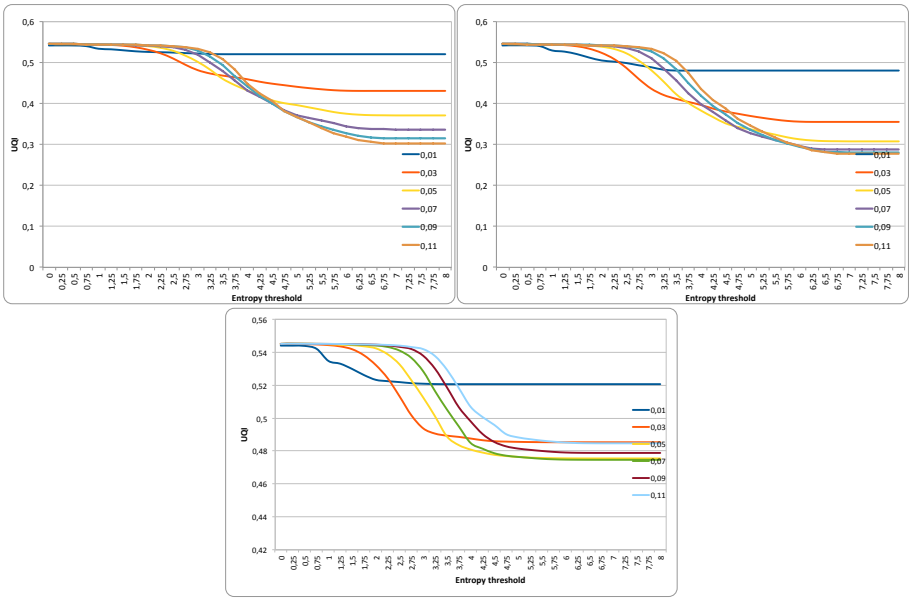

Figure 9: Type of noise: vignetting. UQI value for the Gaussian (top-left), mean (top-right) and bilateral (bottom) filter for the discrimination with entropy method.

It depends on the point density at the furthest distance to the sensor. In the case of a RGB-D camera, we have selected 11 cm.

For the Salt&Pepper noise, the three methods provide better UQI values than the original implementations, although the UQI values for the accelerated and original bilateral filter are not significant. With this noise, a low, even zero, entropy value is preferred as it provides the highest UQI values. It has also been noticed that a low radius (0.01) is the one with highest UQI.

Finally, for the vignetting noise, the Gaussian and bilateral filter have approximately the same UQI as the original method, and therefore this improvement is not appropriate for these filters and noise. However, for the mean filter the UQI value is improved. Consequently, the mean filter is the most appropriate for this kind of noise.

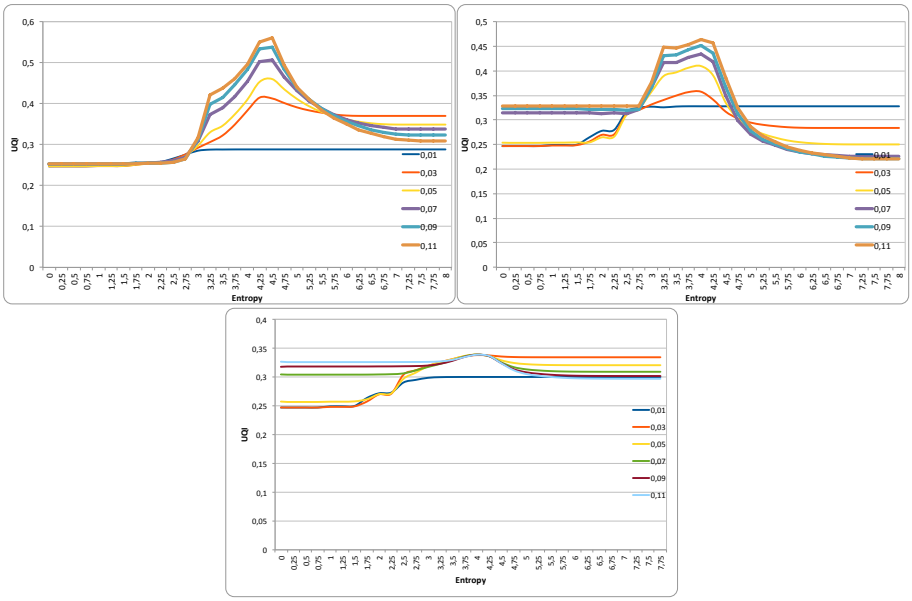

Figure 10: Type of noise: Gaussian. Filter: Gaussian (top-left), mean (top-right) and bilateral (bottom) filter. UQI value modifying the radius with respect to entropy for the neighborhood variation method.

7.3. Results for batch processing with optimal radius

As in the previous method, all the points are processed. However, this time, the optimum radius is only computed for a set of randomly selected points in the image and used as a smoothing radius for their neighbors. This method is designed to represent an important speed-up, compared to the previous method, and obtaining similar UQI values. Figures 13, 14 and 15 show the UQI values for Gaussian, salt&pepper and vignetting noise, respectively. Here, we also try with different initial radii. Depending on the initial radius, the method can provide better results.

This method improves both, processing time and quality. Regarding the original methods with Gaussian noise, the three filters provide better UQI values than the original methods. Considering the UQI values, the best results are obtained when using the Gaussian filter with an entropy value of 4. For the other two kinds of noises, Salt&Pepper and vignetting, the Gaussian filter is

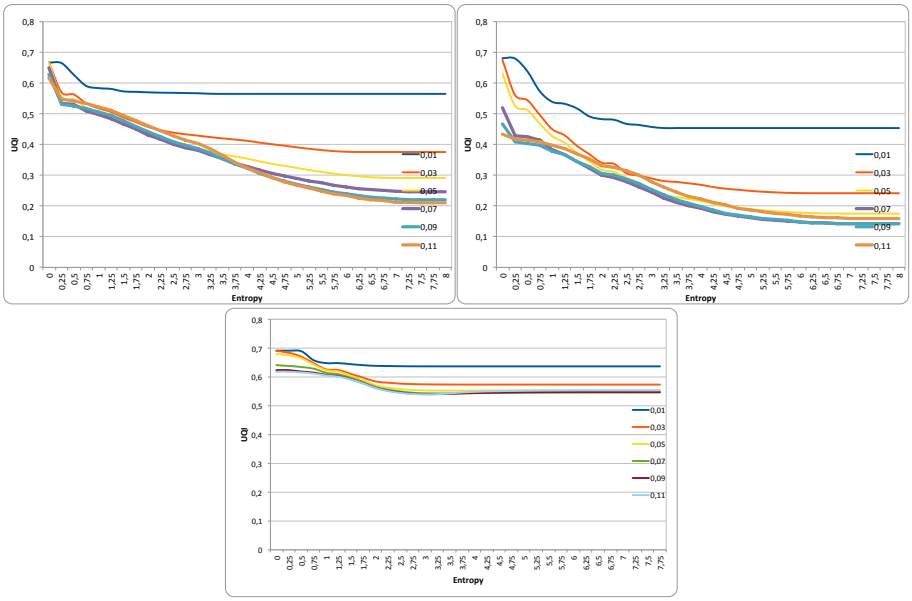

Figure 11: Type of noise: salt&pepper. Filter: Gaussian (top-left), mean (top-right) and bilateral (bottom) filter. UQI value modifying the radius with respect to entropy for the neighborhood variation method.

again the one providing better UQI values. With these noises, a low entropy value gives the best UQI value.

7.4. Processing time

Figure 16 shows the processing time for the different smoothing methods with the improvements proposed in this paper. As we expected, the two methods for speeding up the process have a processing time below the original method. The mean filter is an exception where points are discriminated using entropy improvement, which has a similar processing time than the original method. This is because the processing time of the mean filter itself is similar to the time required to calculate the entropy to decide whether or not to proceed with this point, so the speed-up improvement is finally not obtained.

The *discrimination of points using entropy and a fixed radius* yields a processing time below the original method. This behavior is expected, as several

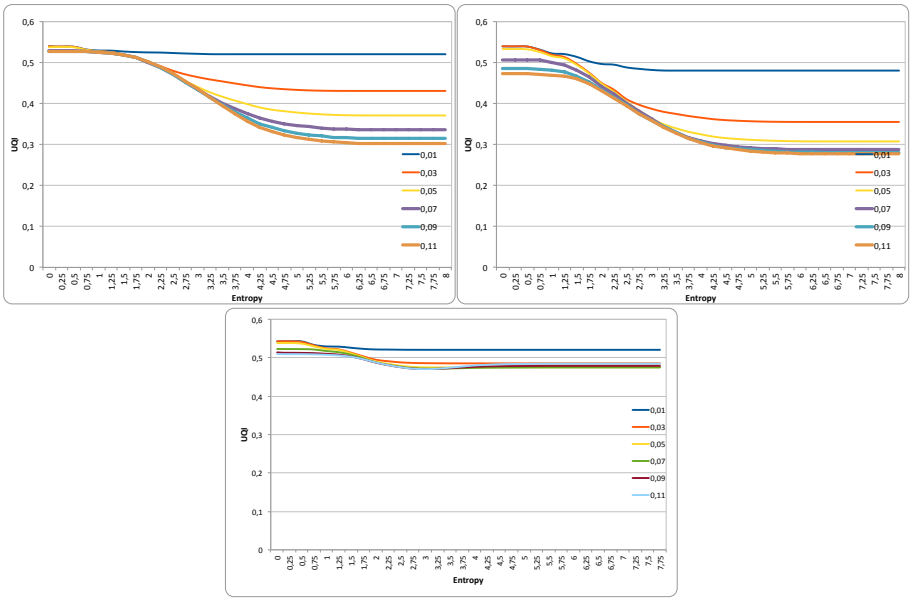

Figure 12: Type of noise: vignetting. Filter: Gaussian (top-left), mean (top-right) and bilateral (bottom) filter. UQI value modifying the radius with respect to entropy for the neighborhood variation method.

points are not processed when entropy is above a certain threshold. The *neighborhood variation using entropy* is between 3 and 4 times slower than the original algorithm. But this behavior is also expected. This last method was designed to improve the quality of the smoothing at the expense of higher processing time. Finally, the *batch processing with optimal radius* method provides the lower processing time.

Regarding the selected radius, the larger it is, the more processing time is needed. This is unimportant, as more points inside the selected neighborhood must be processed. For low radii, there is almost no difference between the methods.

7.5. Discussion

In this section, we summarize and discuss the results of the proposed methods. Figures 17, 18 and 19 show the results obtained from the application of

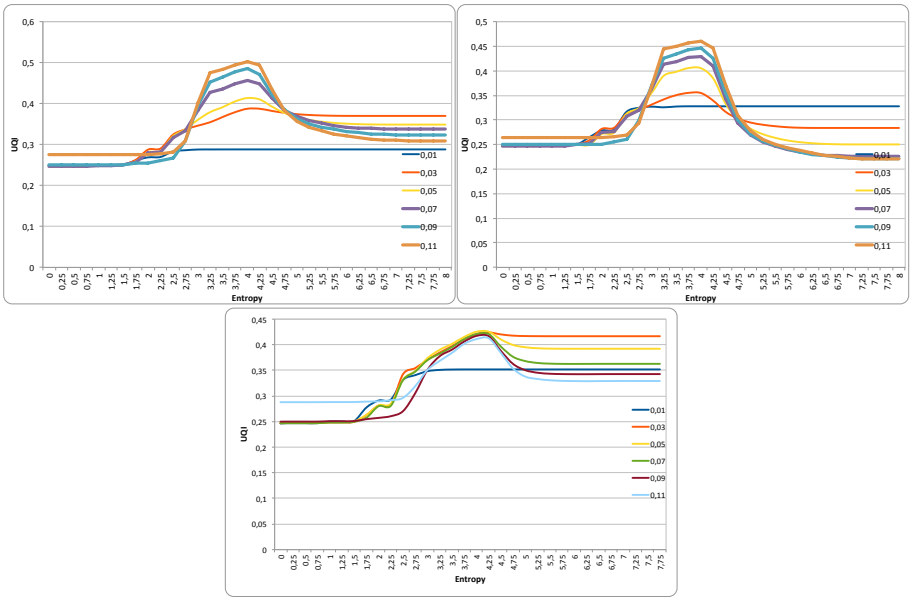

Figure 13: Type of noise: Gaussian. UQI value for the Gaussian (top-left), mean (top-right) and bilateral (bottom) filter for the batch processing method.

the methods with the respective observed noise. For the Gaussian noise, the Gaussian filter provides better UQI and processing time results (Figure 17). It also seems that qualitatively it provides better results in removing the noise. For the salt-and-pepper noise, although the bilateral filter has the better UQI, the Gaussian filter also seems to qualitatively provide the better result. For the previous noises, the Gaussian filter uses less processing time than the bilateral filter. Finally, for the vignetting noise, the bilateral filter is better in both UQI value and qualitative evaluation.

We can conclude that the three methods improve the results compared to the original methods, providing a better UQI value. In our experiments, the neighborhood variation method provides the best UQI value while the batch processing with optimal radius method yields a good UQI value, close to that of the neighborhood variation method. Furthermore, the batch processing with optimal radius method is the fastest.

With regard to the different noises, the Gaussian filter is the most appro-

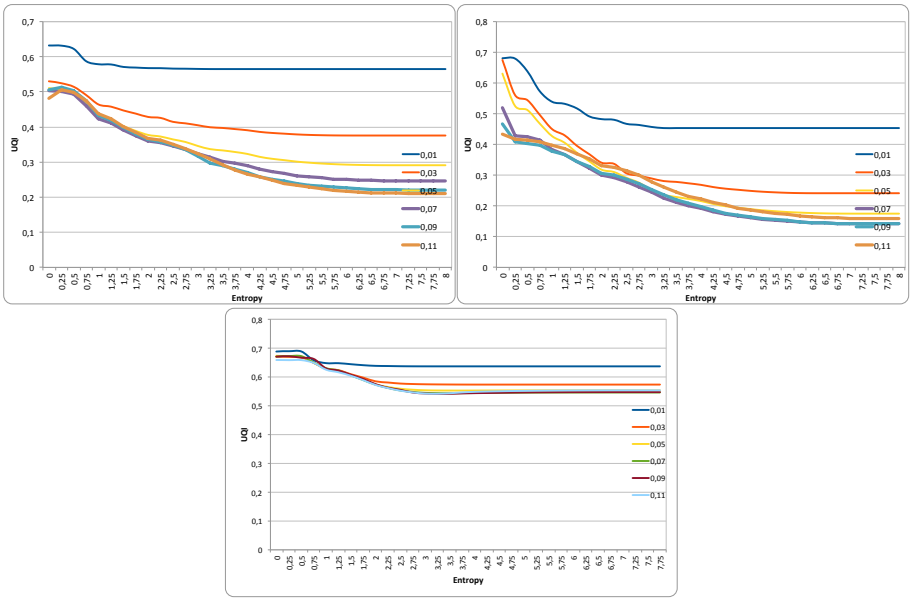

Figure 14: Type of noise: salt&pepper. UQI value for the Gaussian (top-left), mean (top-right) and bilateral (bottom) filter for the batch processing method.

priate for the Gaussian noise. It provides the best UQI value in combination with any of the three proposed improvements. However, for the salt-and-pepper noise, the Gaussian filter has a better visual appearance than the other two, although the UQI values are lower than the bilateral one.

To summarize, we present Table 1, in which we describe the main conclusions of the experimental results reported herein.

8. Results with real images

In this section, we show some results of applying the proposed methods to real images. We present two kind of experiments: qualitative and quantitative.

Figure 20 shows an example of applying the different methods to a real image. In this case, the size of the kernel for all the filters is 0.04cm. The first row represents a section of a real Kinect image. The Gaussian noise present in the data can be observed. The second row shows the result of applying the original methods (from left to right: Gaussian, mean and bilateral). The third

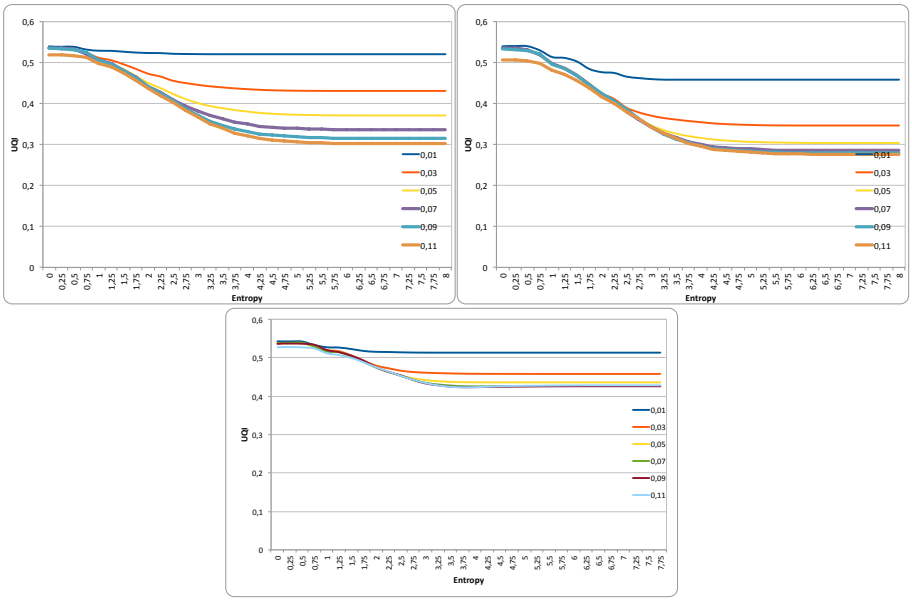

Figure 15: Type of noise: vignetting. UQI value for the Gaussian (top-left), mean (top-right) and bilateral (bottom) filter for the batch processing method.

row shows the results after applying one of our improvements (the fixed radius). It can be seen that the fixed radius improves the original method by smoothing the image area only where the color is uniform. The discrimination of points by entropy using fixed radius avoids processing points in areas where color is variable (high entropy), while the original methods did not make distinctions and smoothed all areas. This effect is especially evident for the mean filter (Figure 20, third row, column two).

The fourth row shows the results after applying the neighborhood variation. In this case, the results are even more improved, although the effect is not so visually evident as with the previous improvements. The improvement between the original method (second row) and the neighborhood variation method (fourth row) is remarkable.

Finally, the fifth row shows the results from the batch processing method. In this case, results are visually very similar to the original ones (second row). Processing time has been reduced, as shown in Figure 22.

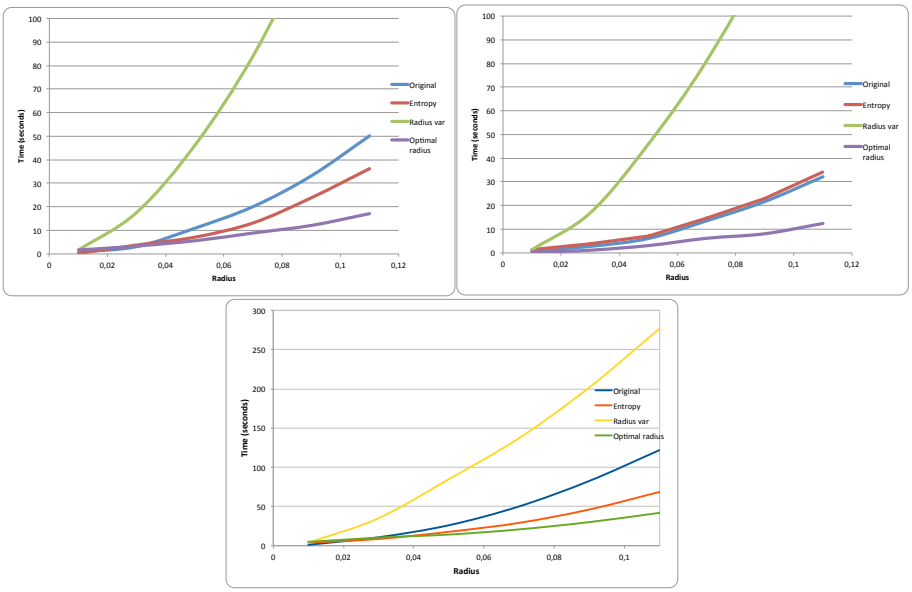

Figure 16: Processing time for the Gaussian (top-left), mean (top-right) and bilateral (bottom) filter.

We can visually compare the results of the proposed methods with respect to the original image. If we focused on uniform color areas and details like the laptop logo or the poster in the wall, the bilateral filter is the best smoothing method. For example, the laptop logo is blurred with Gaussian and mean filter. The bilateral filter is able to preserve the edge information while smoothing the uniform areas. However, the bilateral filter with the fixed radius provides even better results compared to the original bilateral filter, as demonstrated mainly by the decrease in processing time.

Another experiment with real images was conducted, applying the different methods (original and proposed ones). We used part of the Vidrilo dataset [22]. We made a subset extracting 10% frames of the dataset, running all filters and approaches processing 238 point clouds.

Figure 21 shows the comparison between the original and the three proposed methods for each filter using the dataset described. The original method is displayed in red, the discrimination of points using entropy method is displayed

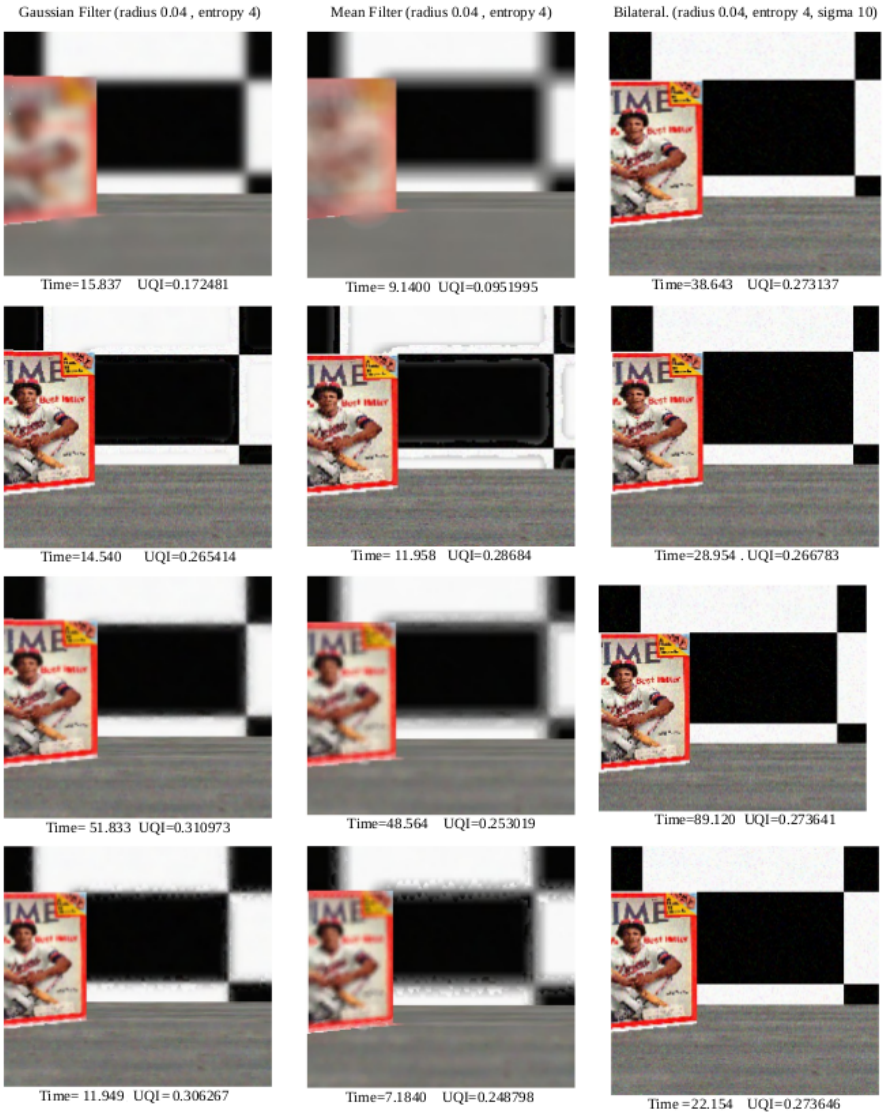

Figure 17: An example of applying our methods with a synthetic image and Gaussian noise. The first row is the original method. The second row is for the fixed radius, the third for the neighborhood variation and, finally, the fourth for the optimal radii.

in green, the neighborhood variation using entropy in blue, and batch processing in yellow.

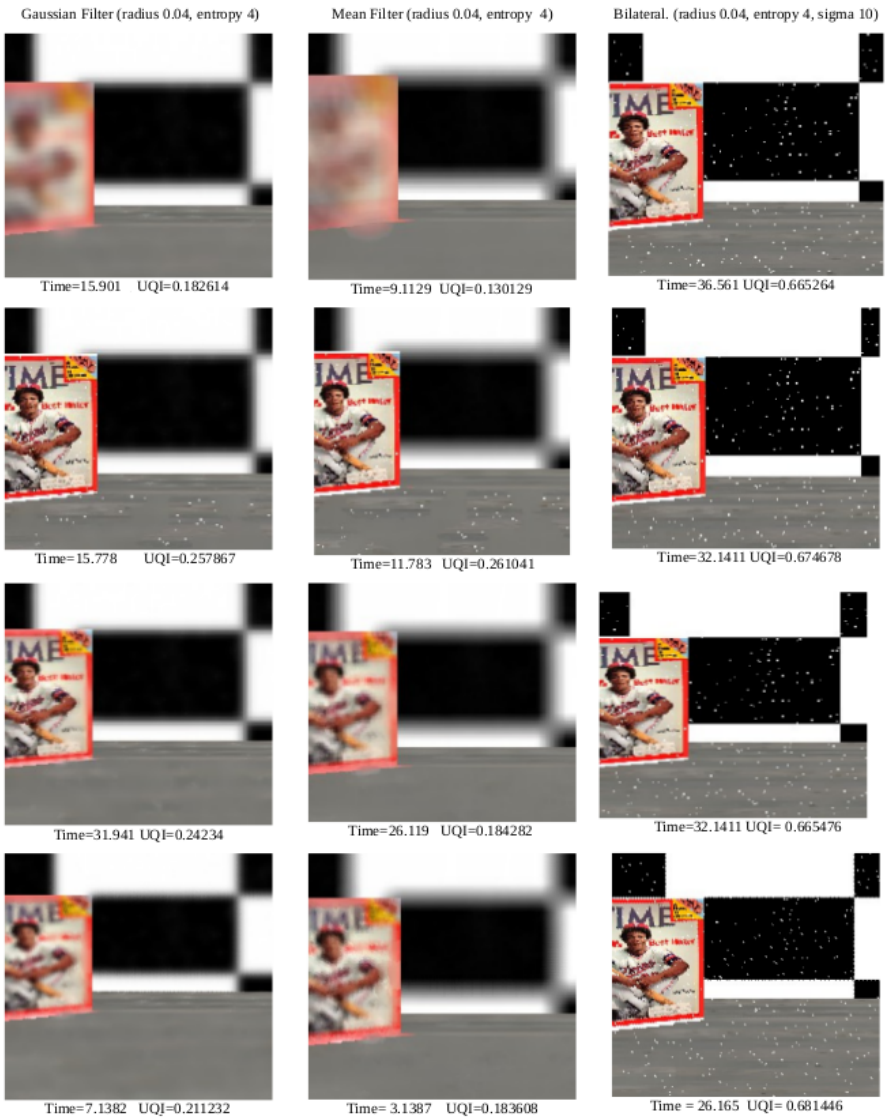

Figure 18: An example of applying our methods with a synthetic image and salt&pepper noise. The first row is the original method. The second row is for the fixed radius, the third for the neighborhood variation and, finally, the fourth for the double radii.

Due to our working with real data, we do not have ground truth data for comparison. However, in the light of the good results achieved using the original bilateral filter, we could set the UQI value obtained by that filter as a target.

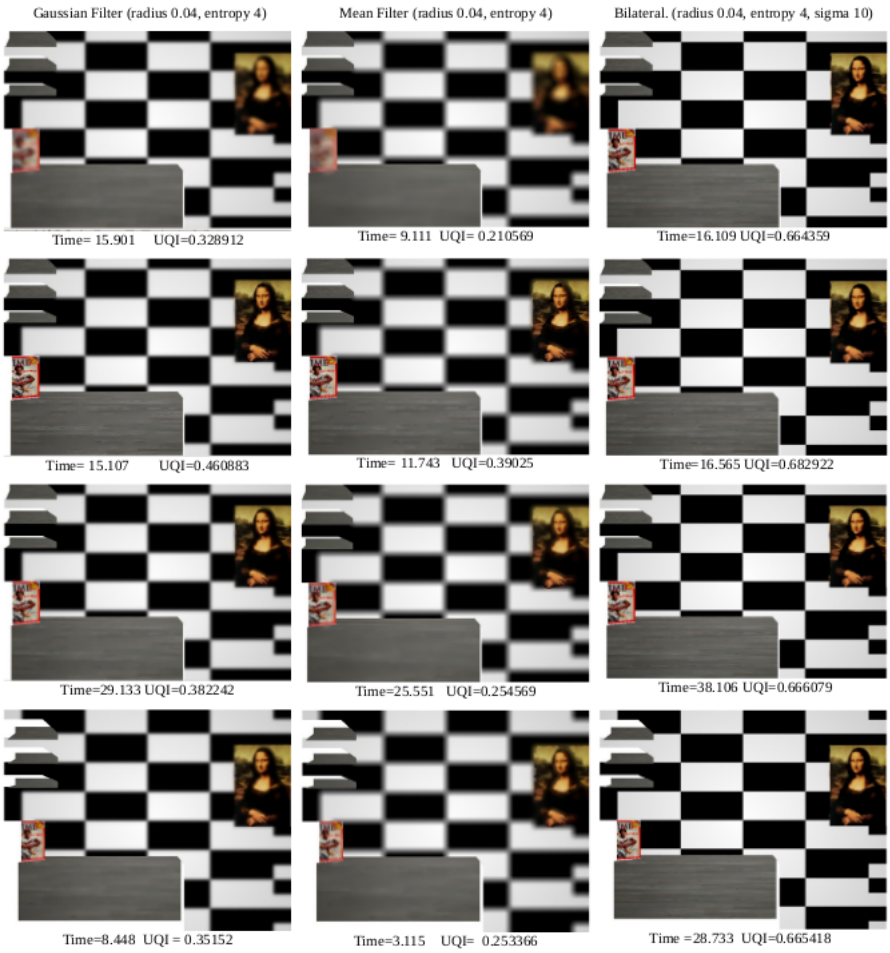

Figure 19: An example of applying our methods with a synthetic image and vignetting noise. The first row is the original method. The second row is for the fixed radius, the third for the neighborhood variation and, finally, the fourth for the double radii.

The quality results in Figure 21 show that:

- With discrimination of points using entropy (green), high UQI values are obtained, but this indicates that the original point cloud is not overlay modified with this method, and many points are discarded (UQI=1 indicates that two images are the same).
- Using neighborhood variation (blue) and batch processing (yellow), all

Table 1: Summary of the main results of the experimentation

Method \ Noise	Gaussian noise	Salt&Pepper	Vignetting
Naive (Best UQI)	Filter:Gaussian Radius=0.03 UQI=0.36 Time=3.11	Filter:Bilateral Radius=0.01 UQI=0.64 Time=1.12 σ_c=1	Filter:Bilateral Radius=0.11 UQI=0.53 Time=104.15 σ_c=1
Discrimination of points (similar UQI than naive)	Filter:Gaussian Radius=0.03 Entropy=4.5 UQI=0.39 Time=3.10	Filter:Bilateral Radius=0.01 Entropy=2.0 UQI=0.64 Time=2.1 σ_c=5.0	Filter:Bilateral Radius=0.11 Entropy=1 UQI=0.54 Time=70.1 σ_c=5.0
Neighborhood variation (Best UQI)	Filter:Gaussian Radius=0.11 Entropy=4.5 UQI=0.56 Time=49.9	Filter:Bilateral Radius=0.01 Entropy=0.25 UQI=0.69 Time=13.11 σ_c=5.0	Filter:Bilateral Radius=0.01 Entropy=0.25 UQI=0.54 Time=10.3 σ_c=5.0
Batch processing (Best UQI)	Filter:Gaussian Radius=0.11 Entropy=4 UQI=0.50 Time=17.1	Filter:Bilateral Radius=0.01 Entropy=0.25 UQI=0.69 Time=12.1 σ_c=5.0	Filter:Bilateral Radius=0.11 Entropy=0.25 UQI=0.532 Time=44.1 σ_c=5.0

points are processed, thus we can compare with the original bilateral UQI values. For the bilateral filter, values are very similar, and for the mean and Gaussian filter, output is better than with the original methods.

Regarding the process time in Figure 22:

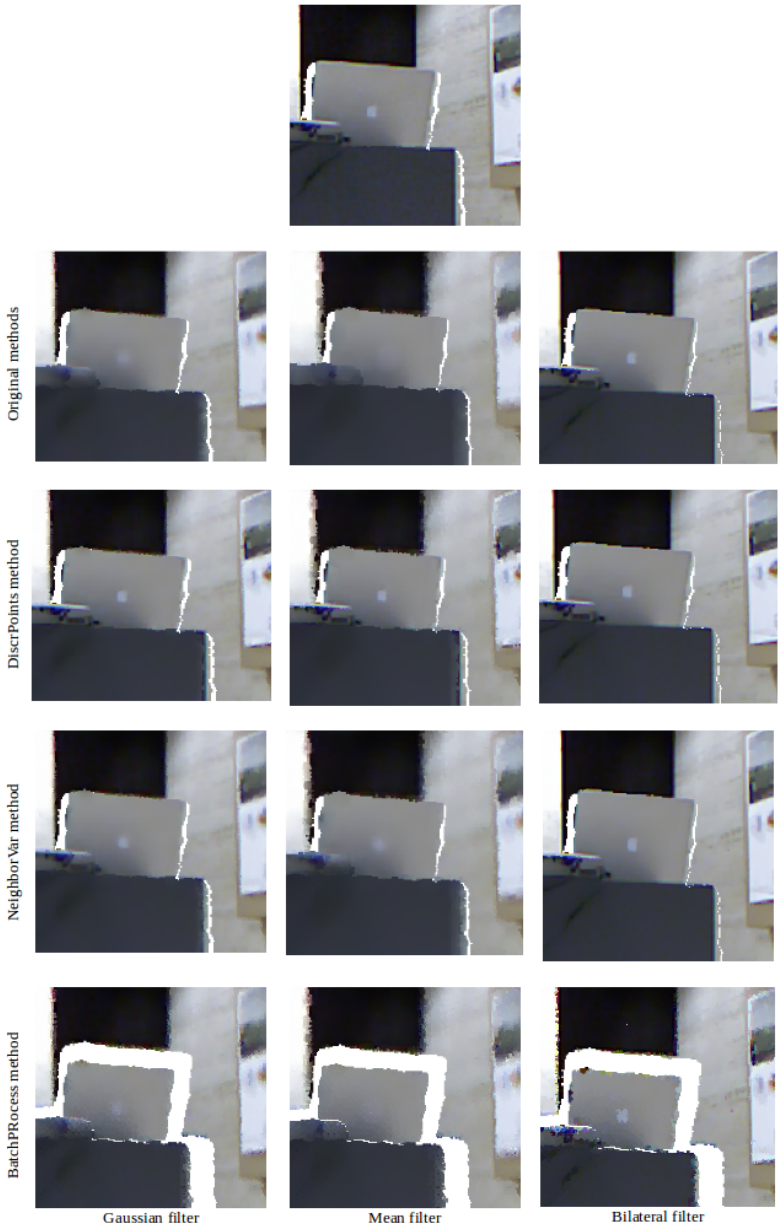

Figure 20: Results of applying the proposed methods to a real image. The figure shows the original (Gaussian, mean, and bilateral) results with each proposed improvement.

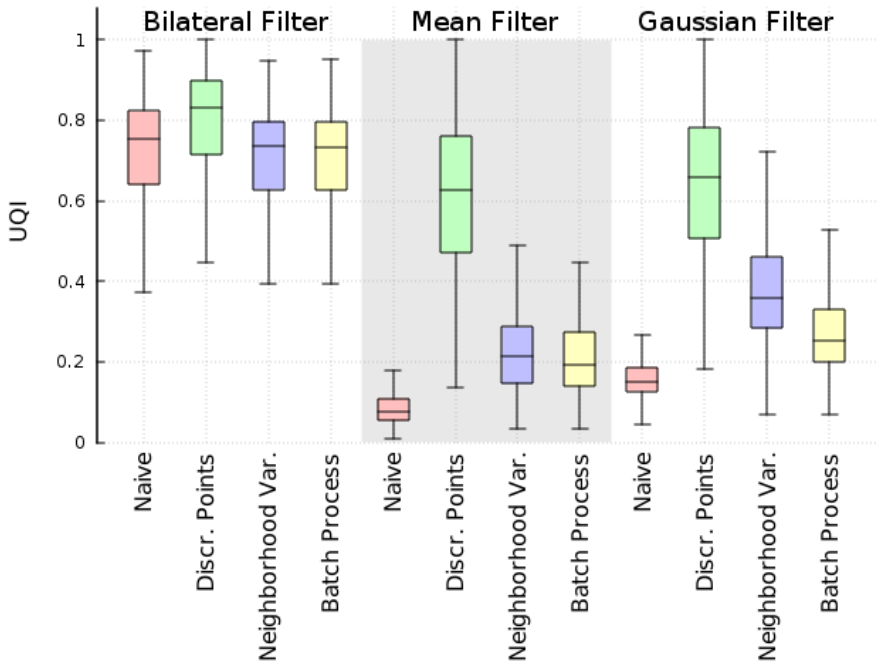

Figure 21: Results with real images. Comparison of UQI results with respect to original images.

- Discrimination of points is faster than with other approaches. This is clear because fewer points are smoothed.
- Neighborhood variation is the most expensive method due to the additional cost of locating the optimal radius before smoothing each point.
- The batch processing with optimal radius achieves a notable acceleration with respect to the original methods and other approaches.

9. Conclusions and future work

In this paper, we have presented and compared the application of different methods to remove color noise from a 3D set captured by a RGB-D sensor. These sensors provide a large amount of colored 3D data. Correcting color

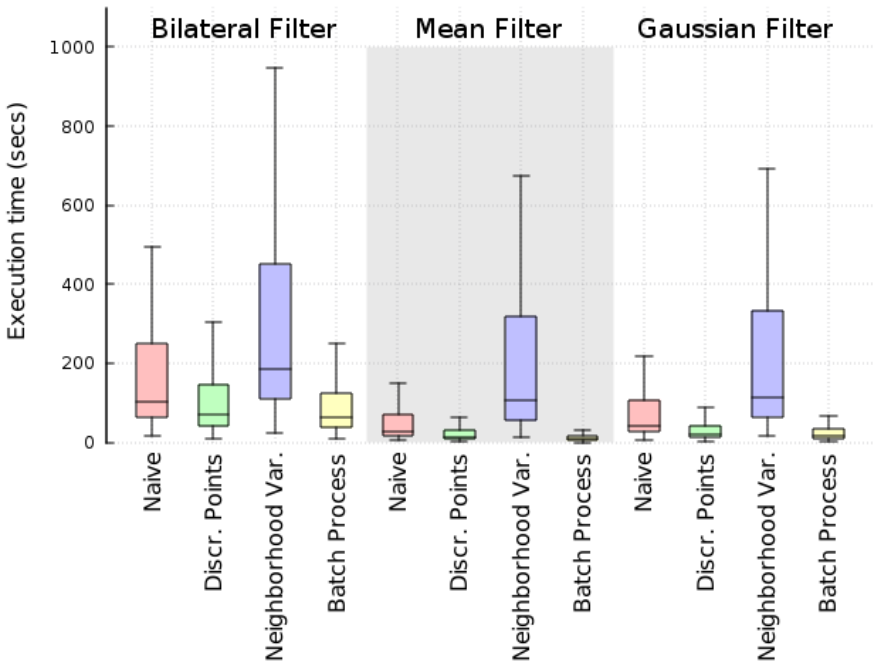

Figure 22: Comparison of execution time using real images.

noise in data from this kind of devices is a time-consuming task. At the start of this paper, we present different sources of noise that commonly affect the images from this kind of sensors. We also present an index value, the UQI, as a way to measure the differences between two colored data sets, normally the original set and the color smoothed one. This value allows us to compare the accuracy of the different smoothing methods that we review in this paper (two linear, Gaussian and mean, and one non linear, the bilateral filter).

To improve the quality of the results and the processing time of the original methods, this work presents three new improvements. The first one is designed to speed up the original methods. It uses entropy to decide whether the point has to be smoothed or not. The second method can be used to obtain the best results as it finds the optimal neighborhood radius for smoothing at each point of the image. The results using this method are considerably better than the original ones but are very time-consuming. Our third method overcomes the time

problem of the previous method by introducing an optimality relaxation term in the search for the optimal radius. The resulting images using this approach are comparable in quality with those of our second method but maintaining the computing time close to the time obtained when using our first method. The experiments carried out in this paper lay down in which situations each of the proposed methods is more suitable.

For each kind of noise we have established which filter works best and which of the proposed methods can be used to improve the filter to obtain the best noise reduction. The selection of the best filter is based on the acceleration and the UQI value obtained. The experimental results also include several real images which are useful for assessing a balance between acceleration and quality. In this way, a person can select a given UQI value and find the acceleration reached or, if the application requires a short time response, select the desired acceleration and seek the amount of color quality loss to be produced based on the UQI index difference between the original smoothing method and the accelerated one.

Regarding the relationship between the neighborhood radius and the processing time, we conclude that when using a small neighborhood radius, the proposed methods do not change the computational time obtained with the original filters. Therefore, the proposed methods should be used for a radius larger than 5 cm. This is due to the fact that smaller radii do not provide smoothing results.

As future works, we plan to develop new methods for color smoothing which also include the processing of sequences of 3D data sets, as they are affected by another type of noise. These methods can be useful for automatic map building in mobile robotics. Another way to speed up the whole process consists in developing a massive parallel implementation for Graphical Processor Units (GPUs) which will be explored shortly.

It seems that all the proposed methods could be applied for smoothing depth data so we plan to develop a similar study with depth data. Furthermore, the bilateral filter has been used as a normal smoothing method so our approach could be applied for normal smoothing. This requires another study.

We also plan to include other kinds of noise and filter types to study their behavior when working with 3D data.

Acknowledgments

This work has been partially supported by grant DPI2013-40534-R from Ministerio de Economia y competitividad of the Spanish Government, supported by Feder funds, and Valencian's Government project GV/2014/097.

References

[1] Bae, S., Paris, S., Durand, F.: Two-scale tone management for photographic look. In: ACM SIGGRAPH 2006 Papers, SIGGRAPH '06, pp. 637–645. ACM, New York, NY, USA (2006)

[2] Bosco, A., Battiato, S., Bruna, A., Rizzo, R.: Noise reduction for CFA image sensors exploiting HSV behaviour. Sensors **9**(3), 1692–1713 (2009)

[3] Caraffa, Laurent and Tarel, Jean-Philippe and Charbonnier, Pierre. The Guided Bilateral Filter: When the Joint/Cross Bilateral Filter Becomes Robust. Image Processing, IEEE Transactions on. 24(4), 1199-1208 (2015)

[4] Kaur, Manjeet and Gupta, Shailender and Bhushan, Bharat. An Improved Adaptive Bilateral Filter to Remove Gaussian Noise from Color Images. International Journal of Signal Processing, Image Processing and Pattern Recognition. 8(3), 49-64 (2015)

[5] Liu, Junyi and Gong, Xiaojin and Liu, Jilin. Guided inpainting and filtering for Kinect depth maps Pattern Recognition (ICPR), 2012 21st International Conference on. 2055-2058 (2012)

[6] Shen, Ju and Cheung, Sen-Ching Samson. Layer depth denoising and completion for structured-light RGB-D cameras. Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on. 1187-1194 (2013)

[7] Miao, Dan and Fu, Jingjing and Lu, Yan and Li, Shipeng and Chen, Chang Wen. Texture-assisted kinect depth inpainting. Circuits and Systems (ISCAS), 2012 IEEE International Symposium on. 604-607 (2012)

[8] Chen, Li and Lin, Hui and Li, Shutao. Depth image enhancement for Kinect using region growing and bilateral filter. Pattern Recognition (ICPR), 2012 21st International Conference on. 3070-3073. (2012)

[9] Moorfield, Bradley and Haeusler, Ralf and Klette, Reinhard. Bilateral Filtering of 3D Point Clouds for Refined 3D Roadside Reconstructions Computer Analysis of Images and Patterns. 394-402 (2015)

[10] Gao, Zhenzhen and Neumann, Ulrich. Feature enhancing aerial LiDAR point cloud refinement. IS&T/SPIE Electronic Imaging. 901303-901303 (2014)

[11] Brown, M., Lowe, D.G.: Automatic panoramic image stitching using invariant features. Int. J. Comput. Vision **74**(1), 59–73 (2007)

[12] Cover, T., Thomas, J.: Elements of Information Theory. Wiley (2006)

[13] Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image denoising by sparse 3-d transform-domain collaborative filtering. Image Processing, IEEE Transactions on **16**(8), 2080–2095 (2007)

[14] Dissanayake, M., Dissanayake, M., Newman, P., Clark, S., Durrant-Whyte, H., Csorba, M.: A solution to the simultaneous localization and map building (slam) problem. Robotics and Automation, IEEE Transactions on **17**(3), 229–241 (2001)

[15] Eisemann, E., Durand, F.: Flash photography enhancement via intrinsic relighting. ACM Trans. Graph. **23**(3), 673–678 (2004)

[16] Endres, F., Hess, J., Engelhard, N., Sturm, J., Burgard, W.: 6D visual slam for RGB-D sensors. at - Automatisierungstechnik **60**, 270–278 (2012)

[17] Engelhard, N., Endres, F., Hess, J., Sturm, J., Burgard, W.: Real-time 3D visual slam with a hand-held camera. In: Proc. of the RGB-D Workshop on 3D Perception in Robotics at the European Robotics Forum. Vasteras, Sweden (2011)

[18] George, A., Prabavathy, A.K.: A survey on different approaches used in image quality assessment. International Journal of Emerging Technology and Advanced Engineering **3**(2), 197–203 (2013)

[19] Han, J., Shao, L., Xu, D., Shotton, J.: Enhanced computer vision with microsoft kinect sensor: A review. Cybernetics, IEEE Transactions on **43**(5), 1318–1334 (2013)

[20] Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., Fitzgibbon, A.: Kinectfusion: Real-time 3D reconstruction and interaction using a moving depth camera. ACM Symposium on User Interface Software and Technology (2011)

[21] Gonzalez, R., Woods, R.: Digital Image Processing, 3rd edition Prentice Hall (2008)

[22] Martinez-Gomez, Jesus and Garcia-Varea, Ismael and Cazorla, Miguel and Morell, Vicente: Vidrilo: The visual and depth robot indoor localization with objects information dataset. In:The International Journal of Robotics Research (2015)

[23] Miropolsky, A., Fischer, A.: Reconstruction with 3D geometric bilateral filter. In: Proceedings of the Ninth ACM Symposium on Solid Modeling and Applications, SM '04, pp. 225–229. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland (2004)

[24] Nister, D., Naroditsky, O., Bergen, J.: Visual odometry. In: Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the

2004 IEEE Computer Society Conference on, vol. 1, pp. I–652 – I–659 Vol.1 (2004)

[25] Nüchter, A., Lingemann, K., Hertzberg, J., Surmann, H.: 6D slam - 3D mapping outdoor environments. J. Field Robot. **24**(8-9), 699–722 (2007)

[26] Petschnigg, G., Szeliski, R., Agrawala, M., Cohen, M.F., Hoppe, H., Toyama, K.: Digital photography with flash and no-flash image pairs. ACM Trans. Graph. **23**(3), 664–672 (2004)

[27] Rusu, R.B., Cousins, S.: 3D is here: Point Cloud Library (PCL). In: IEEE International Conference on Robotics and Automation (ICRA). Shanghai, China (2011)

[28] Tomasi, C., Manduchi, R.: Bilateral filtering for gray and color images. In: Computer Vision, 1998. Sixth International Conference on, pp. 839–846 (1998). DOI 10.1109/ICCV.1998.710815

[29] Viejo, D., Cazorla, M.: A robust and fast method for 6DoF motion estimation from generalized 3D data. Autonomous Robots **36**(4), 295–308 (2014)

[30] Viejo, D., Garcia-Rodriguez, J., Cazorla, M.: A study of a soft computing based method for 3D scenario reconstruction. Applied Soft Computing **12**(10), 3158 – 3164 (2012)

[31] Wang, Z., Bovik, A.: A universal image quality index. Signal Processing Letters, IEEE **9**(3), 81 –84 (2002)

[32] Zhang, Z.: Microsoft kinect sensor and its effect. IEEE Multimedia **19**(2), 4–10 (2012)

[33] Zheng, Y., Lin, S., Kambhamettu, C., Yu, J., Kang, S.B.: Single-image vignetting correction. IEEE Transactions on Pattern Analysis and Machine Intelligence **31**(12), 2243–2256 (2009)