2018-01-01

Improving Time-Of-Flight And Other Depth Images: Super-Resolution And Denoising Using Variational Methods

Salvador Canales Andrade
University of Texas at El Paso, scait99@gmail.com

IMPROVING TIME-OF-FLIGHT AND OTHER DEPTH IMAGES: SUPER-RESOLUTION AND DENOISING USING VARIATIONAL METHODS

SALVADOR CANALES ANDRADE

Doctoral Program in Electrical and Computer Engineering

APPROVED:

Sergio D. Cabrera, Ph.D., Chair

Boris Mederos, Ph.D., Co-Chair

Miguel Velez-Reyes, Ph.D.

Miguel Argaez, Ph.D.

Charles Ambler, Ph.D.
Dean of the Graduate School

Dedication

To Gely

and

To Tony, Bryan, Stephanie and Angelita

IMPROVING TIME-OF-FLIGHT AND OTHER DEPTH IMAGES: SUPER-RESOLUTION AND DENOISING USING VARIATIONAL METHODS

by

SALVADOR CANALES ANDRADE, M.C.E.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

Electrical and Computer Engineering

THE UNIVERSITY OF TEXAS AT EL PASO

May 2018

Acknowledgements

I am in debt with a large number of persons along my academic life and professional career in regards of this research. The modest way to correspond with gratitude is to recognize their impact in my decisions that made possible the realization of this research and the accomplishment of this academic degree. The most logical way that I found to acknowledge them is in the chronological order that their influence had over this matter. In first place, I appreciate the encourage to enroll in the Graduate School provided by Kurt Schneider my first supervisor at Delphi Automotive Systems as well as the economic support given by Antonia (Tony) Andrade, my mother, to cover certain percentage of the tuition, college and research expenses. In second place, my appreciation extends to my current supervisors at Aptiv, Naishadh Desai and Roukoz Atallah, for their comprehension and flexibility that my academic needs required overall during the years of the course work. Chronologically, the next group of persons have been fundamental to the theoretical and practical value of this research, my two advisors, Dr. Sergio Cabrera at the Electrical and Computer Engineering Department from UTEP in the area of Image Processing and Dr. Boris Mederos at Departamento de Matemáticas from UACJ in the area of Variational Methods. Additional, my sincerely appreciation to Dr. Miguel Argaez (UTEP) and Mtro. Isamel Silva (UACJ), for their contributions in the area of Numerical Optimization and Matlab resources respectively. Special thanks to my wife Gely and my kids for the numerous hours of sacrifice and by handling my absence during the time that I spent performing this investigation. Finally, I wish to thank to my colleagues at Aptiv who continuously have encouraged me to complete this work: Monica de la O, Patricia Bravo, Douglas Champion, Oscar Alcantara, Alfonso Mendoza and Ricardo Quiñones.

Abstract

Depth information is a new important source of perception for machines, which allow them to have a better representation of the surroundings. The depth information provides a more precise map of the location of every object and surfaces in a space of interest in comparison with conventional cameras. Time of flight (ToF) cameras provide one of the techniques to acquire depth maps, however they produce low spatial resolution and noisy maps. This research proposes a framework to enhance and up-scale depth maps by using two different regularization terms: Total Generalized Variation (TGV) and Total Generalized Variation with a Structure Tensor (TTGV). Furthermore, the proposed technique implements the Alternating Direction Method of Multipliers (ADMM) not just to solve a denoising problem as previous efforts, it employs this numerical method to inpaint, reduce impulsive noise and up-scales the low-resolution observation from ToF camera frames by means of fusion of them instead of getting help for the super-resolution process from a second RGB color camera as others have considered. The proposed technique's performance relies on the precision of the multi-frame registration and the denoising capability. The registration performance is addressed with the iterative motion estimation proposed by Lukas-Kanade while the noise elimination is based on an impulsive noise detection scheme recently adapted to the multi-frame super-resolution problem. The proposed algorithm of this dissertation is objectively validated using simulated depth maps and toy images while the subjective evaluation is performed using real ToF depth image sequences. In general, the algorithm shows reduction of staircasing phenomena and enhancement of simulated and real depth maps under the presence of Gaussian noise and two types of impulsive nose: salt and pepper and random value.

Table of Contents

List of Tables

List of Figures

Research Objective

This research pretends to develop new techniques to increase the spatial resolution of ToF depth maps by using multiple frames rather than using the texture information from a co-located high resolution conventional RGB camera image, which requires that a second imager sensor be available. The expected output is a R times (R=4 at least) up-scaled depth map with increased level of image details provided by the fusion of multiple frames only from the ToF sensor. The research is being developed with depth maps from different sources: collected frames with camera models from Softkinetic, the DS311 (long range), DS325 (close range) and the Intel Senz3D Depth Gesture Camera and known datasets of 3D depth maps, generated using stereo camera pair.

In the best case, most of the high-resolution details will be transferred to the depth map with reflectivity and other non-depth information suppressed as a result of the ADMM TGV and ADMM TGV with tensor upscaling techniques. In the worst case, the up scaled result could have minimal extra detail from the multi-frame fusion and tensor computation and the quality could be no worse than a conventional bilinear and bicubic interpolation results of resizing by a factor of R.

Chapter 1: Depth Map Images

There are several techniques to acquire directly or indirectly depth information from a specific scene by using different types of devices such as structured-light systems, stereo vision cameras and ranging devices. Some of these techniques have been used for several years, nevertheless, semiconductor integration has allowed the development of ranging systems based on the Time-of-Flight (ToF) principle. These have been integrated into low cost imagers in the last few years.

In general terms, ToF cameras work like any other ranging device in which a signal delay is processed to quantify a distance. However, the main difference is that while a conventional ranging system has only one photo sensor that detects the reflected signal of a collimated light beam, the ToF cameras have a complete array of sensors which simultaneously receive the reflected light of a non-coherent illumination system. Details in first developments of this technology can be found in [1], [2] and improvements are described in works like [3].

The ToF cameras have several advantages and disadvantages over other depth map construction techniques. On the one hand, the fast frame rates of complete depth maps acquired with ToF cameras allow real-time applications, as well as low software complexity in comparison with stereo vision systems. On the other hand, the ToF imagers show a high amount of noise and low spatial resolution. These and other issues as well as description of alternative techniques to build depth maps are addressed in detail in this chapter.

The high amount of noise can be treated with de-noising techniques and bilateral filters as have been shown for this type of images in [4]. The main advantage of bilateral filters applied to ToF images is the preservation of edges and the low pass filtering of the rest of the areas which

may be affected by flickering pixels and other sources of noise as modeled in [5]. In this way, the quality of ToF depth maps is increased without loss of important information by applying bilateral filters or similar adaptive techniques. Furthermore, a novel approach to noise reduction is presented in [6] which switches between a median filter and a bilateral filter depending on the nature of the depth image portion that is going under filtering.

In addition to the noise reduction techniques for depth maps acquired with ToF cameras, image fusion techniques can be beneficial as well. Some of the fusion based techniques to increase spatial resolution of depth maps employ Joint Bilateral filters [7] and Markov Random fields [8] as mentioned in [9] and are focused on 3DTV applications. Moreover, newer techniques have been proposed in [9] and [10]. An optimized edge weight technique is used in [9] while an interpolation weights technique is used in [10]. All these techniques take advantage of an additional acquisition device, an RGB camera, which provides a higher spatial resolution observation than the low spatial resolution of the ToF cameras.

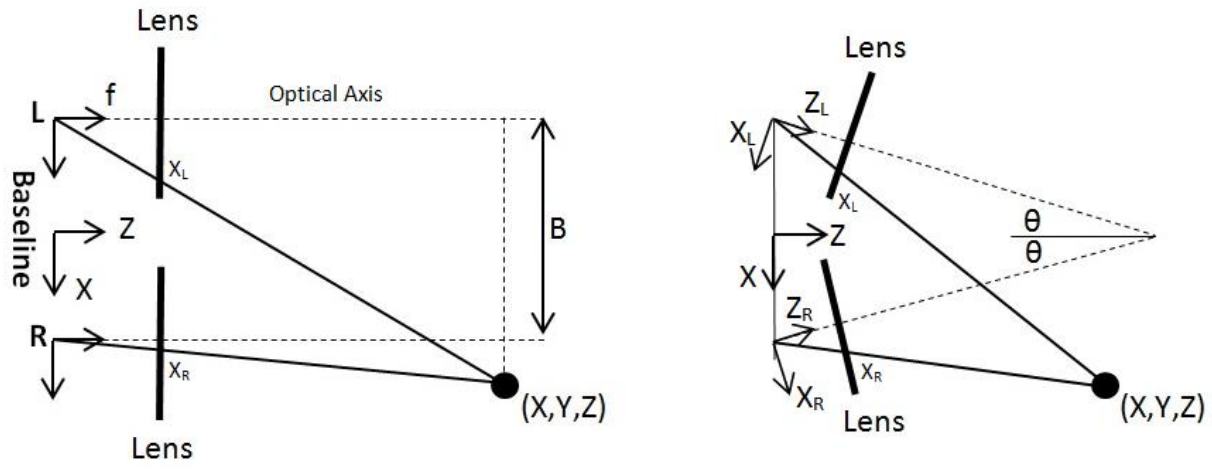
This chapter describes the three most used techniques to build 3-dimensional depth maps. A depth map is an array of values similar to an image, but the values represent depth instead of light intensity. Additionally, this section shows in detail the advantages and disadvantages as well as the reasons why the specific technique was selected to be discussed in the context of this research.

1.1 Stereo Vision (Passive Triangulation)

The stereo vision is a well know technique to compute depth of a scene and is part of the passive triangulation techniques to generate 3D maps. The principle is similar to the human vision for which this technique employs two cameras. The main process consists of developing a 3D geometry of the two cameras and the objects in front of them. The geometry of the system can have a parallel optical axis or it can have converging optical axes. Figure **1** presents both geometries with the axis normal to the baseline of the two cameras, based on diagrams shown in [11].

In this way, it is possible to obtain depth in both cases just by applying projective geometry and knowing the cameras dispositions. More complete models have been developed, regarding lens distortions and camera defects, such as the pinhole model as shown in [11] and [12]. Such models require significant amount of processing to generate a complete 3D map, thus in general stereo vision techniques are restrictive. It is difficult to obtain results for real time applications in which several 3D frames per second are required. Additionally, the depth information obtained is not uniformly accurate in regions of uniform texture, as mentioned in [13]. On the other hand, the depth maps obtained from a high-resolution stereo pair can also have the same high resolution.

The correspondence problem is one of the main disadvantages of this technique, as mentioned in [1]. This problem is generated by the need of knowing which points from one image plane corresponds to the same points in the image plane of the second camera. There are several techniques to solve the correspondence problem as listed in [12] such as Correlation-Based Methods or Feature-Base Methods. However, the generation of the pairs of points in both image planes that are projections of the same point in the world is a very complex problem and the solution is computationally expensive as stated in [1].



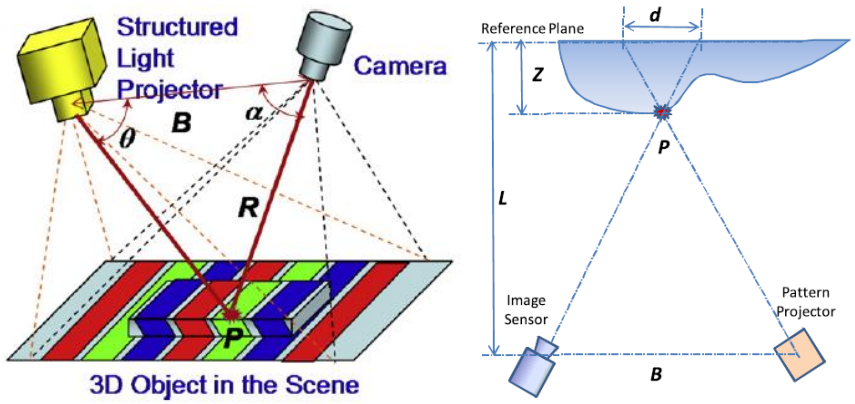
(a) (b)

Figure 1: (a) Geometry with parallel optical axis. (b) Geometry with converging optical axes. Slightly modified diagrams from: **[11]**.

1.2 Structured-light (Active Triangulation)

The structured light technique as mentioned previously is also employed to generate 3D maps. Due to its potential for high precision, several applications use this approach, such as in industrial surface inspection and in video game console interfaces (based on the Kinect I). The main principle of this technique is based on active triangulation to come up with the depth of a scene. First, a pattern needs to be projected over the target scene to be mapped, then a sensor will compare the projected pattern over the scene and the different objects in the field of view of the sensor. The system will then compare this captured image against reference patterns to perform an accurate differentiation from the other objects despite the ambient light falling on the objects as explained in [1]. Figure **2** (a) shows a basic diagram of a structured light system taken from [13]. As can be appreciated, the distance R from the point P to the Camera can be computed if the base line B is known and the reference pattern can be differentiated properly as shown in Figure **2** (b).

The projected pattern can vary depending on the application and its properties such as periodic or quasi-periodic patterns as shown in [13] and [14].



(a) (b)

Figure 2: (a) Illustration of structures light (b) Z depth calculation. Source: **[13]**.

The main disadvantage of this technique consists of the need for a high powered projected light and a well-focused pattern, which can be highly susceptible to ambient illumination. In many cases a scanning of the light through the scene may be required which would reduce the frame rate of the system [1] and would limit the standoff distance because of limited energy of the projection light. In contrast, single laser scanning systems do not have these limitations, as mentioned in [13].

1.3 Time-of-Flight Cameras

The time-of-flight principle has been used by different devices since several decades ago. The basic principle is to measure the time delay between a radiated signal and its reflection. Devices originally using this principle are LIDARs and 3D laser scanners/rangers with rotating mirrors. However, all of these devices process the time delay for a single point at a time in contrast with the processing by a ToF camera which has a matrix array of CMOS based photo-detectors. The detector responses can be processed in parallel since they have special features to sense the reflected signals at the same time. Thus, ToF cameras require a non-coherent source of light completely different to the coherent light used by LIDARs. On the other hand, ToF cameras cannot deal with long ranges as LIDARs do because the amount of required power to illuminate big areas at long distances is prohibitive. This makes ToF cameras suitable only for ranges on the order of 1 to 10 meters with current technology More detailed state of the art review can be found in the ToF camera surveys presented in [15] and [16]. Notice that the camera used by us in this research was not yet available by the publication dates of [15] and [16], nevertheless the camera details and specs are provided in the following sections.

A LIDAR in general is able to generate a 3D map but a scanning of the scene must be performed using a mechanical system. Examples of application using these systems include topographic scanning from airplanes in rural and urban areas, automotive ranging, building scanning for architectural projects, etc. On the other hand, applications of ToF cameras are in the areas of Human Machine Interface (HMI) for personal computers, automobiles and video games, as well as 3DTV capturing, manufacturing inspection and scanning for 3D printing.

The basic principle of operation of a ToF camera is more directly based on phase shift measurements than in direct quantification of time delay between radiated and reflected signal.

The circuitry to distinguish directly times on the order of picoseconds to resolve depths of millimeters is not competitive with indirect methods based on phase shift detectors by charge integration as shown in Figure 4**.** Equation (1) shows the basic relation between phase and distance, where c is the speed of light, f is the frequency of the radiated signal and $\Delta\varphi$ is the phase difference between the radiated and the reflected signals. This basic physics formulas are presented in [1], [3], [5], [6], [15], [17] and in most of the ToF research articles and books.

$$Distance = \frac{c}{2} \cdot \frac{\Delta\varphi}{2\pi f} \tag{1}$$

Figure 3 shows a generic block diagram presenting the main components of a ToF camera system. Two objects are shown here so that two different phase shifts $\Delta\varphi_1$ and $\Delta\varphi_2$ are computed for the groups of sensors that receives the corresponding reflected signal. In this way, the bigger phase shifts correspond to Object1 since the distance is longer than for Object2 which is closer to the Depth sense imager.

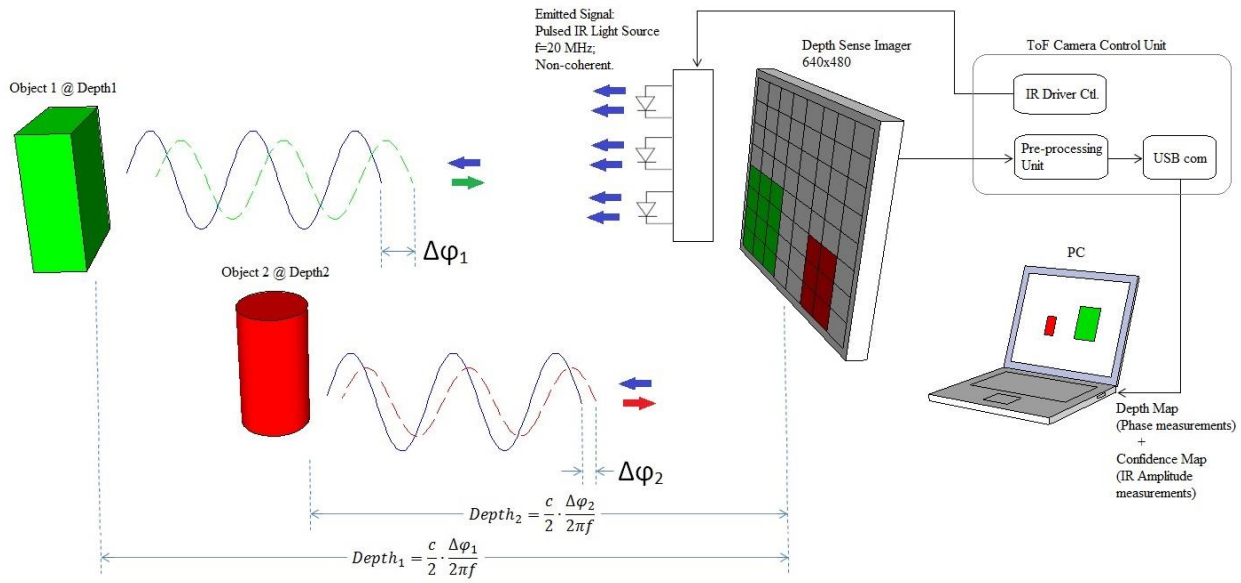


Figure 3: Time of flight principle.

The whole detector matrix can be processed fast enough so that complete image frames are delivered at rates of 25 to 60 per second, by the DS325 Softkinetic camera used for this investigation. As can be seen in Figure 3, the depth is directly proportional to phase shift, and a general description of the phase shift computation is provided by [17] in terms of charge integration from 4-phase control signals during the period of the radiated signal. This process is shown in Figure 4. In contrast with the sinusoidal waveforms of the previous figure, it is more practical to use square waveforms easily generated by digital hardware as explained in [18]. Nevertheless, the exact circuitry of each commercial ToF camera is proprietary information not disclosed in the public domain.

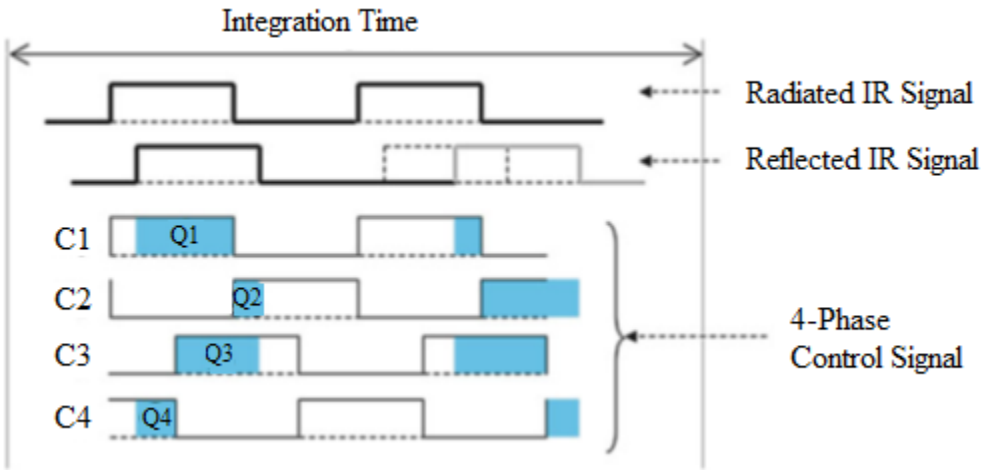


Figure 4: Four-Phase Charge Integration Process. Source: **[17]**.

The computation of the phase difference is defined by

$$\Delta\varphi = \tan^{-1}\left(\frac{Q_3 - Q_4}{Q_1 - Q_2}\right) \tag{2}$$

After presenting here the basic theory of ToF cameras, some of the main disadvantages of these devises are discussed next and more details are mentioned in [17]:

1. Low depth accuracy: the depth precision can be affected by the reflectivity of the materials, variation in illumination conditions, power of the radiated IR signal and geometry of the objects in the target scene.
2. Motion blur: scene objects in movement can cause blurring in the depth image due to the limited frame rate as consequence of long integration times. On the other hand, small integration times decrease the depth accuracy of the camera.

3. High noise level and low spatial resolution: The ToF cameras group several measurements to calculate a single pixel value; this increases the signal to noise ratio (SNR) but decreases the spatial resolution.

This work deals with the last problem by proposing methods to increase the spatial resolution and filter the still present noise in the depth measurements.

Finally, comparisons between the three different techniques presented in section 3 and other approaches for creating depth maps are presented in [1], [17] and [18].

1.4 Time of Flight Frame Acquisition

The selected cameras used for the initial ToF frame acquisition for this project were the DS311 and the DS325 from the manufacturer Softkinetic and the DepthSenze 3D which contains also the same imager sensor from Softkinetic, but it is commercialized by Creative Labs for Intel Developer applications. The Softkinetic's technology is currently used in indoor applications and automotive environments for gesture recognition applications. The manufacturer offers a PC driver to achieve communication with the camera through USB interface.

Table 1 shows the spatial resolution of each sensor in the cameras used for this project. As can be seen, the DS311 has a longer range of 7 meters, but it's spatial resolution is low. The second model, the DS325, has a higher spatial resolution but its range is only of 3 meters. While the DS311 can be used in applications to map a whole room, the DS325 is better suited for hand gesture recognition application close to the user such as right in front of a computer monitor.

Table 1: ToF Camera Specification.

Camera model	**Spatial Depth Sensor Resolution** (Pixels)	**Spatial RGB Sensor Resolution** (Pixels)	**Range** (meters)
DS311	160x120 QQVGA	320x240 QVGA	7
DS325	640x480 VGA	1280x720 HD	3
Senz3D	640x480 VGA	1280x720 HD	3

Since the manufacturer software focuses on gesture recognition, games implementation and camera evaluation, it was required to implement a custom interface to acquire depth and RGB frame sequences and to store these sequences in a proper data/file format to allow algorithm development and testing. For this purpose, a program was developed in Visual Studio C++. The interface can recognize both models of the cameras and allows the user to configure the acquisition parameters. In addition to the *dll* functions available from the supplier, the OpenCV library was used to take advantage of its implemented functions to manage the acquired frame sequences and its optimized data memory transfers.

The main purpose of the developed interface is to store data in a compatible format for Matlab access of the depth and RGB frames and additional information provided by the camera firmware such as IR intensity (also called confidence image) frames, camera parameters, and mapped pixel coordinates from one image plane into the other. The selected format to transfer the files was PGM, which is widely used for intermediate data exchange of graphics information. The

PGM format was slightly modified from its original version that used a matrix type array, in which each value of the array is an 8-bit number representing a pixel from a single frame. The data storage approach was modified to avoid generation of a single PGM file per frame. Instead, multiple frames of a video sequence are inserted in a single file, in which every row corresponds to a complete frame that will be rearranged later in the Matlab script. In the case of the depth frames, additional modifications were required due to the fact that depth pixel values are provided as 16 -bit numbers. Those values are split into two 8 bits number to be inserted into the PGM files and then assembled in the proper form using a Matlab script.

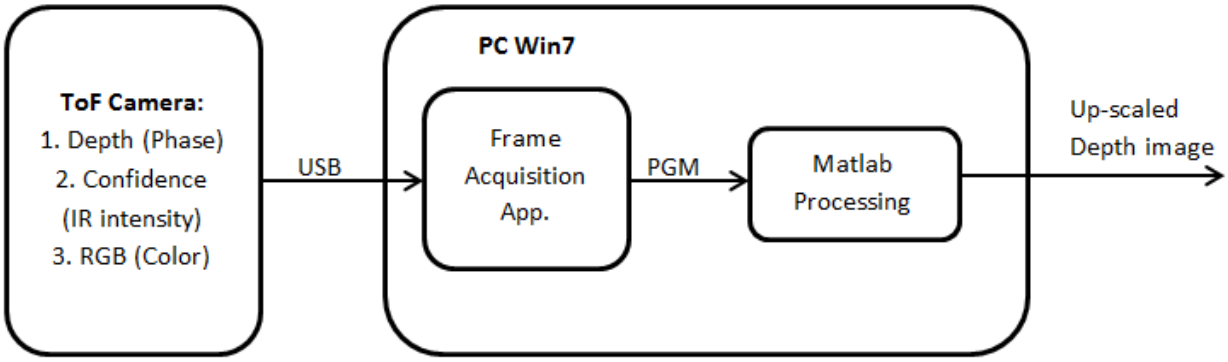


Figure 5: Frame Acquisition Block Diagram before Matlab Processing and its role in the complete processing system.

Once the frames are arranged in a Matrix format, a noise reduction is performed over the Depth information to eliminate as much as possible the flickering pixels and other sources of noise. The filter selected for initial experimentation is the standard bilateral filter based on properties shown in [6] and the actual Matlab filter implementation was taken from [19] which follows the theoretical approach of [4]. Another two approaches for up-scaling are proposed based on wavelet

fusion and multiple frame super-resolution techniques; these two additional approaches are described in the sections further ahead.

The frame acquisition process of the Intel Senz3D camera was simpler since there was available already Matlab code to acquire depth maps.

1.5 ToF Depth Image Dataset Survey

Additional to the acquired images with our own ToF cameras a survey of data set was conducted to identify ToF depth image Datasets available for research as well as stereoscopic depth map Datasets were used for the algorithm performance for validation purposes. It is important to mention that stereoscopic depth maps have a High-resolution that can be used as ground truth while down sampled versions of that high-resolution frames are used to simulate noisy x-y shifted and rotated observations.

Table 2 shows several found dataset with appropriate content for this investigation since they include consecutive frame sequences for multi-frame super-resolution applications.

The Middlebury dataset [20] is commonly used to validate up-sampling techniques of depth maps like in the work developed by [21]. Although this dataset doesn't have video sequences of depth maps, its high-resolution disparity results are very well suited for performance evaluation of the up-sampling techniques and for comparison against other investigations that have used the same depth map files from this dataset.

Table 2: ToF Camera Specification.

Dataset	Author Reference	Sensor / Technique	Features	Frames		
				Depth	IR	RGB
Middlebury	D. Scharstein and R. Szeliski [20].	Stereoscopic	High Resolution Depth Maps with Multiple objects and scenes.	Yes		Yes
CoRBS	Oliver Wasenmmuller, et al. [22]	**Kinet V2 (ToF)** & others	Video sequences with high fps proper for multi-frame SR	Yes	Yes	Yes
FAU	Thomas Kohler, et al. [23]	Kinet V1 & **Simulated ToF**	Proper frames for validation.	Yes		Yes
TLC	Samuele Gasparrini, et al. [24]	**Kinet V2 (ToF)** & others	Video sequences of people falling and walking.	Yes	Yes	Yes

The Comprehensive RGB-D Benchmark for SLAM data set (CoRBS) [22] is a very complete source of ToF information that contains video sequences of different trajectories in 4 scenes. The dataset not just provide depth, infrared and RBG date, it also records the ground truth trajectory of the ToF camera from special markers that were installed on it. The recorded trajectories offer translation and rotation average speed which can provide an idea of the limitations of multi-frame super-resolution real application and valuable data for image registration and camera calibration.

The Dataset collected by [23] present a novel application of ToF super-resolution in the medical area. This data set contains endoscopic simulated depth maps with a noise model characteristic of the ToF cameras. Although the real data collected with the camera prototype is not available at the dataset, the fidelity of the simulated data is still of interest for this investigation as well as it is a good source of test cases for the up-sampling techniques.

Finally, [24] collected several depth frame sequences of people falling and lying for fall detection, however the motion of persons from frame to frame is relatively high and the

displacement only of the person body with the rest of the background scene remaining in the same frame location represent a challenge for partial registration and makes clear that is not the best type of sequences for multi-frame super-resolution.

Chapter 2: Up-sampling and Image fusion techniques for Depth Images

This chapter presents two representative image up-scaling techniques as background relevant to our work as well as to provide examples of state-of-the-art techniques. Two techniques discussed below are: Markov Random Field based up-scaling [8] and Edge Weighted Optimization based image up-scaling [9].

2.1 Markov random fields

Diebel and Thrun present a novel technique which uses Markov Random Fields (MRFs) in [8] to generate a high-resolution depth image taking advantage of the fact that discontinuities in depth images and in the color-texture tend to co-align. The work presented by Soh in [25] presents a pertinent introduction to explain the framework used in [8] as well as an analysis of its limitations.

The main idea of the MRF is to generate a probability of each pixel in the high-resolution version of the depth image based on conditional probability and following the theory of generating a model of the image based on this type of fields as shown in [26]. In this way, if **L** is defined as a vector of the available depth pixels and **Z** as the vector of a set of random variables corresponding to each pixel, the conditional probability between them will have the following expression according to the well-known Bayes' rule:

$$P(\boldsymbol{Z}|\boldsymbol{L}) = P(\boldsymbol{L}|\boldsymbol{Z}) \cdot P(\boldsymbol{Z})/\mathrm{P(L)}$$

Namely that an optimal high-resolution depth image can be obtained with the correct configuration of the set of random variables in **Z** such that P(**Z**|**L**) is maximized. For more details

on the penalty functions and the smoothness function, as well as increased resolution depth image results see [8] and [25].

According to [25], the main disadvantage of the MRF technique is the edge smoothing, which occurs when color image doesn't yield the same edges as the depth image. In other words, the smoothness happens when the color pixels have nearly the same values at the corresponding depth image edge pixel locations.

2.2 Weighted Optimization Approach for ToF Sensor Fusion

One of the recent novel approaches to increase the low spatial resolution of ToF cameras is the technique developed by Schwarz in [9]. This section pretends to provide a brief overview of this technique as part of the background on state-of-the-art techniques related to the work of this investigation.

The main idea of this technique is to regard the depth up-scaling problem as an energy minimization problem. In this way, the low-resolution information is mapped into a sparse image in a high-resolution coordinate domain which is that of the RGB high-resolution texture image. This is done with proper image registration between both camera image planes, compensating mapping errors in image alignment by using texture data as ground-truth to align the depth map during the up-scaling process. It is important to mention that the camera calibration is performed using tools from [27] which is used also in this investigation and presented in section 5.

The empty spaces of the sparse image are filled by employing error energy minimization weighted by texture edges from the high resolution (HR) image. It is called the Edge Weighted

Optimization Concept (EWOC) since edge detection is performed on the HR image to validate them with the low-resolution edges. Figure 6 shows an example of the edge weight generation W_E.

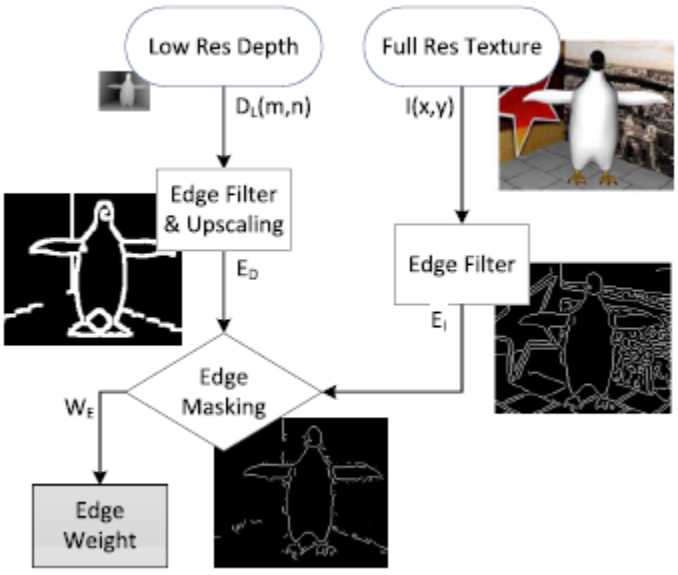


Figure 6: Edge weight generation example. Source: **[9]**.

The EWOC technique, presented by Schwarz in earlier publications and in [9] where it is improved with additional weights to address the specific ToF cameras characteristics by implementing a noise reduction strategy and a flickering reduction technique based on temporal consistency. The weight functions used in [9] perform similar tasks as bilateral filtering. They preserve edges while sharpening depth transitions between objects by relaxing the spatial constraints defined by the similarities in the pixel values of the neighborhood. These pixels are usually part of the same object surface with large areas with gradual depth transitions. The spatial constraints are actually just the difference between consecutive depth pixels in horizontal and vertical direction of computation which are described in [9] as horizontal energy Q_H, vertical

energy Q_V and the overall energy Q_S. These energies are summarized in the following group of equations:

$$Q_H = \sum_x \sum_y W_E(x,y)\epsilon_h^2(x,y)$$

$$Q_V = \sum_x \sum_y W_E(x,y)\epsilon_v^2(x,y)$$

$$Q_s = Q_H + Q_V$$

Where the horizontal and vertical error are defined as:

$$\epsilon_h(x,y) = D(x,y) - D(x+1,y)$$

$$\epsilon_v(x,y) = D(x,y) - D(x,y+1)$$

And the weighted function as generated in Figure 6 is:

$$W_E(x,y) = 1 - E_I(x,y) \cdot E_D(x,y)$$

Note that E_I is the binary edge map of the high-resolution image and E_D is the binary edge mad of the corresponding depth image.

Three strategies are investigated in [9] for the proposed ToF Super-Resolution (TSR) technique. These techniques consist of: using only a single weighted TSR (S-TSR), using a double weighted TSR (D-TSR) or using a triple weighted TSR (T-TSR). The overall block diagram of their technique is shown in Figure 7. More details about the computation of the different components of the algorithm can be found in [9].

Finally, Table 2 shows the results of the Single ToF Super-Resolution algorithm performance as well as comparison with other up-scaling techniques such as: Joint Bilateral Up-scaling (JBU) presented in [7], the 3D-JBU from [28] with Multi-scale color measure extension (MCM) and the Weighted Mode Filtering (WMF) algorithm from [29]. The processing time is normalized with respect to Bilinear up-scaling. The results show that the S-TSR has a better performance in objective quality although the 2D JBU+MCM algorithm has the minimum processing time. The objective quality is the evaluation of the percentage of bad matching pixels, taking as ground true high-resolution depth maps generated using Middlebury dataset stereo image pairs [20], which were down sampled to be used as standard test images for up-sampling algorithms.

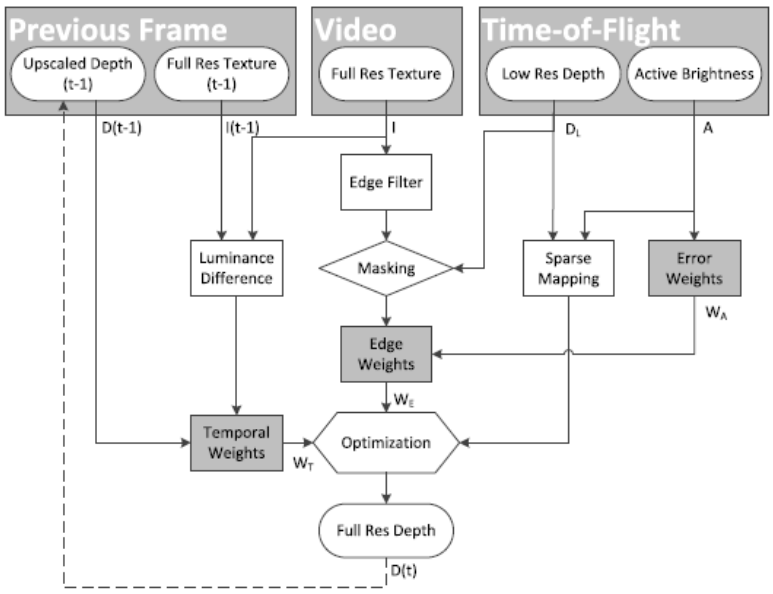


Figure 7: Overview of ToF Super-Resolution algorithm. Source: **[9]**.

Table 2: Middlebury Evaluation by a factor of 8 of the S-TSR performed in **[9]** with previous data from **[29]**.

Algorithm	**Tsukuba**		**Venus**		**Teddy**		**Cones**	
	all [%]	**time [s]**	**all [%]**	**time [s]**	**all [%]**	**time [s]**	**all [%]**	**time [s]**
Bilinear upscaling	10.40		3.26		11.90		14.70	
2D JBU	9.04	0.61	2.04	0.94	14.00	0.95	14.70	0.95
2D JBU + MCM	9.71	**0.23**	2.01	**0.33**	16.10	**0.34**	16.70	**0.34**
3D JBU	7.89	143.00	1.67	215.50	10.70	220.60	12.10	219.20
3D JBU + MCM	4.46	43.30	0.66	65.20	8.88	65.80	10.60	65.60
WMF	4.35	0.36	0.61	0.54	9.51	0.55	9.43	0.55
S-TSR	**3.29**	0.30	**0.42**	0.63	**6.08**	0.57	**4.81**	0.63

2.3 Image guided Depth Up-sampling using TGV

Another novel technique to up-sample ToF Depth maps with the help of a RGB higher resolution image was developed by David Ferstl in the framework presented in [21]. Ferstl presents a convex optimization problem formulation using an anisotropic diffusion tensor to perform the low-resolution frames up-sampling. The following section presents an overview of the Ferstl algorithm and subsequent section present an improvement in the algorithm by doing minor change in the tensor.

2.3.1 Anisotropic Tensor TGV Up-sampling using Primal Dual Optimization Publication Overview

The principal idea behind the technique from [21] is to add information from a high-resolution intensity image acquired via a conventional RGB camera in the context of a variational optimization framework. The additional information will contribute with more image details or high frequency spatial content to the poor detailed depth image acquired by the ToF Camera. This

contribution is provided mainly from the textural edges that most of the times also occur in the object borders, which generates depth discontinuities in the depth map. However, this is not necessarily the case in several scenarios and type of images as will be shown in the section 2.4.2. Despite this last remark, the technique generates higher density maps with increases spatial resolution in terms of objective metrics and it improves the visual quality. The minimization problem formulation proposed by Ferstl is provided by the following equation:

$$D_{HR} = \underset{u,v}{\operatorname{argmin}}\{\alpha F(u,v) + G(u, D_{LR})\}$$

Where the first term is the regularization term times a constant α. It is defined as the Total Generalized Variation (TGV) functional that incorporates the prior knowledge of the smoothens of the image and also introduces an image tensor. Then the next term in the equation is the data term $G(u, D_{LR})$ that quantifies the fidelity between the low-resolution measurements D_{LR} and the optimization variable u, eventually the optimum solution of the problem. Thus, rewriting the previous equation, the following expression is obtained:

$$D_{HR} = \underset{u,v}{\operatorname{argmin}} \left\{ \left[\alpha_1 \int_{\Omega_{HR}} \left| T^{\frac{1}{2}} (\nabla u - v) \right| dx + \alpha_0 \int_{\Omega_{HR}} |(\nabla v)| dx \right] + \left[\int_{\Omega_{HR}} w |(u - D_{LR})|^2 dx \right] \right\}$$

The role of the tensor T is to carry the spatial high-frequency details into the TGV functional, it is defined as:

$$T = e^{(-\beta |\nabla I_{HR}|^{\gamma})} n n^T + n^{\perp} n^{\perp T} \quad (3)$$

Where the gradient of the high-resolution image is $n = \frac{\nabla I_{HR}}{|\nabla I_{HR}|}$ and $n^{\perp}$ is the normal vector to n, and β and γ adjust the magnitude and sharpness of the tensor as stated in [21].

Additionally, it is required to mention that Equation (3) originally defined the square root of the tensor in [21], however, the implemented Matlab code by Ferstl uses the actual tensor value and not its root. This error in [21] is in part justified by the fact that tensors roots are used in norms, but not if they are part of the argument of an inner product like in the gradient descent used in [21].

The case of a regularization term with $\boldsymbol{l}^2$ norm is:

$$\|S\nabla u\|_2 = \sqrt{\langle \mathrm{S}\ \nabla u\ , S\ \nabla u\rangle}$$

$$= \sqrt{\langle\ \nabla u\ , S^T \mathrm{S}\ \nabla u\rangle}$$

$$= \sqrt{\langle\ \nabla u\ , \mathrm{T}\ \nabla u\rangle}$$

$$= \sqrt{\nabla u^T\ \mathrm{T}\ \nabla u}$$

where $T = S^T \cdot S;\ S = L^{\frac{1}{2}} \cdot Q^T;\ S = L^{\frac{1}{2}} \cdot Q;$ and L contains the eigenvalues of T in its diagonal and Q contains the normalized eigenvectors of T.

Then it is not necessary to perform a spectral decomposition and the complete tensor might be used as shown in the Equation (4) without the need to compute the tensor root:

$$\min_{u\in R^{MN},v R^{2MN}} \max_{p\in P,q\epsilon Q} \alpha_1\langle T(\nabla u - v), p\rangle + \alpha_o\langle \nabla v, q\rangle + \sum_{i,j\epsilon\Omega} w_{i,j}\left(u_{i,j} - D_{S_{i,j}}\right)^2 \quad (4)$$

Moreover, [30], a posterior work from Ferstl, uses the same tensor, but this time it is defined not as the root but as done here in (3) as well as its gradient proceeds from a different source, which is a high-resolution depth patch instead of the RGB high resolution image as required for the corresponding application.

In general, one of the main advantages of the TGV is the reduction of the staircasing phenomenon from the up-scaled image solution of the optimization process. Furthermore, the primal dual technique employed in [21] facilitates the TGV super-resolution implementation avoiding complex and large terms involved in the computations. Unfortunately, all this is at the expense of required number of iterations of the order of 10^4 to reach low mean square errors.

More information about the tensor properties and the benefits of including them in the proposed technique will be addressed in detail in subsequent sections.

2.3.2 RGB High-resolution Texture-Less Tensor Improvement for Depth Up-sampling Fusion Anisotropic Tensor - TGV

As it was initially mentioned in the previous section, not in all of the cases the textural edges from RGB image have a correspondence with all the edges present in a depth map. This is specialty evident in cases in which a flat surface that is normal to the focal axis has certain decorative painting, i.e. a checkerboard type of pattern, that will create edges which are not going to be present in the edge map produced from a ToF image, in such a case only the contour of the

actual physical surface will correspond with the contour from the high-resolution intensity image. Although it was not part of the main objective of this investigation, it is pertinent to include the implementation of a simple improvement in the method proposed by Ferstl in [21]. It consists of validating automatically which edges form the high-resolution frames have a correspondence with edges from the ToF low-resolution frames, a similar kind of validation as that performed in the masking edge step of Schwarz [9].

Thus, the proposed modification to the previous technique is to define a new tensor as follows:

$$T = e^{(-\beta|\nabla I_{HR} \circ P|^{\gamma})} nn^{T} + n^{\perp}n^{\perp T} \tag{5}$$

where:

$$P = Threshold\left(\nabla D_{HR_{Bilinear}}\right) \tag{6}$$

Equation (6) defines the matrix P that contains the $Threshold(\cdot)$ component wise operator that generates a binary array in which every pixel location will be assigned with a 1 if its value is above of a given level L_{Th} or a 0 if its value it is below as defined by Equation (7).

$$Threshold_{i,j}(x) = \begin{cases} 0 & if\ x_{i,j} < L_{Th} \\ 1 & if\ x_{i,j} > L_{Th} \end{cases} \tag{7}$$

Then the Threshold operator is applied to $\nabla D_{HR\_Bilinear}$, which is the gradient of an up-scaled version of the low-resolution depth map obtained using a conventional up-scaling technique, in this case the bilinear method.

Thus Equation (5) differs from the original Equation (3) in the exponential term, which is now the Hadamar product between the gradient ∇I_{HR} and P ending in a new masked gradient to guide the up-scaling process.

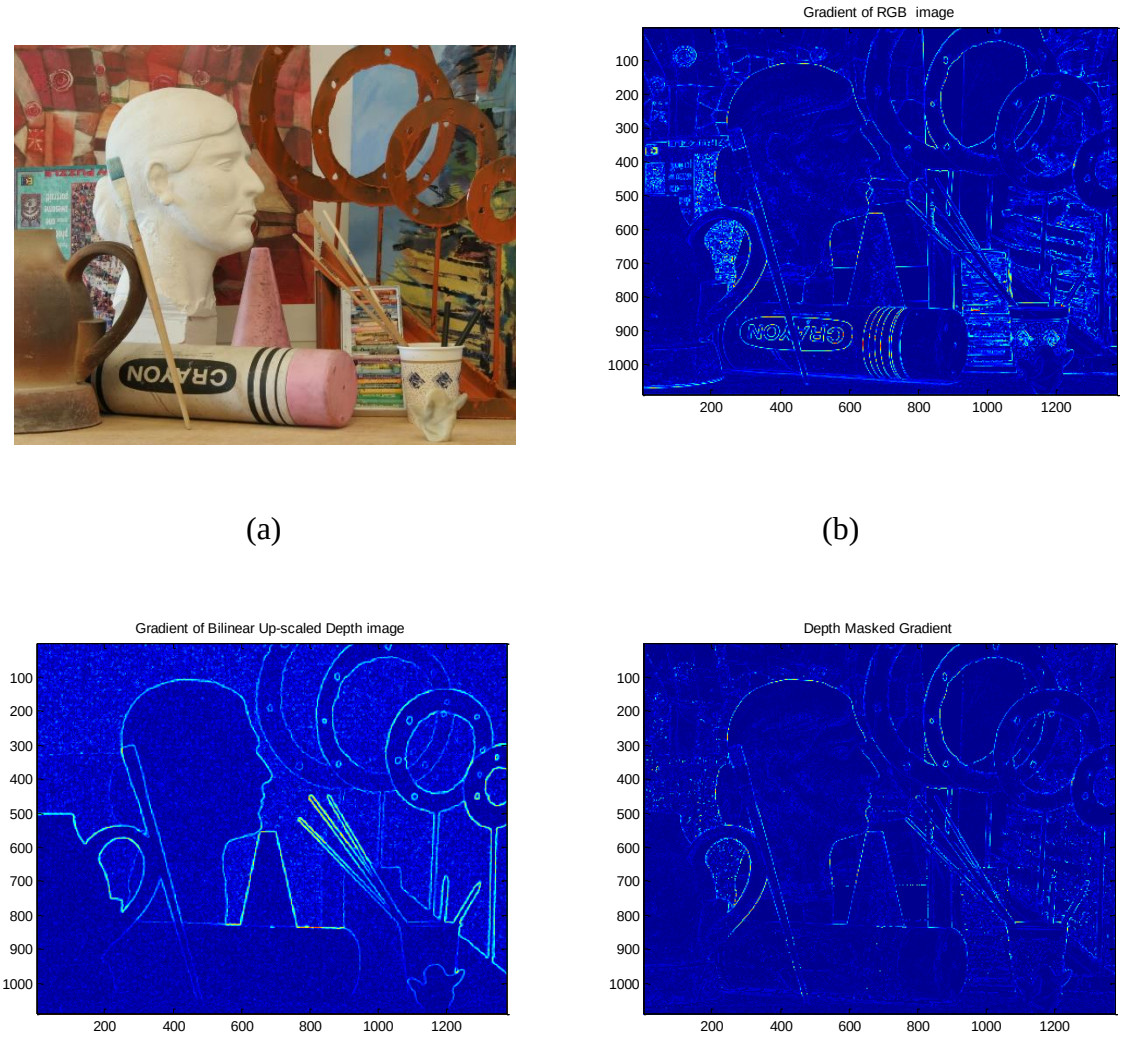
Table 3 shows the comparison between the original method from Ferstl and the proposed masked gradient improvement. As it can be seen, there is an objective improvement of the mean square error (MSE), over all other approaches for the figures Art and Books, and a reduced improvement for the third image, Moebius.

Table 3: Anisotropic Tensor TGV upscaling Performance Comparison, only 10^3 iterations.

Method	**Middlebury Images**		
	Art	**Books**	**Moebius**
Bicubic [MSE]:	4.1572	3.8207	4.2234
Bilinear [MSE]:	3.5958	3.1295	3.4557
nearest [MSE]:	5.0147	4.6848	5.1979
TGV L2 [MSE] [21]:	1.5513	1.1928	1.0823
TGV L2 - Masked Gradient [MSE]:	1.3924	0.8560	1.0284
Masked Gradient improvement %:	**10.2403**	**28.2336**	**4.9774**

The pots from Figure 8 through Figure 10 show the original gradient of the comparison between the original gradient and the masked gradient. The original gradients that are used in the TGV algorithm are displayed as well as the proposed masked gradient. Edges generated by decorative details that actually don't pertain to real object edges are eliminated. Figure 8 shows how overall the "Crayon" brand is removed from the original gradient, which causes an

improvement of 10.24% at the end of 10^3 iteration. Although [21] uses 10^4 iterations, only 10^3 are enough to show the improvement regarding the time demanding requirements of the Matlab code which for this test was to perform in a multiple core and GPU system as was the case in the original work [21].

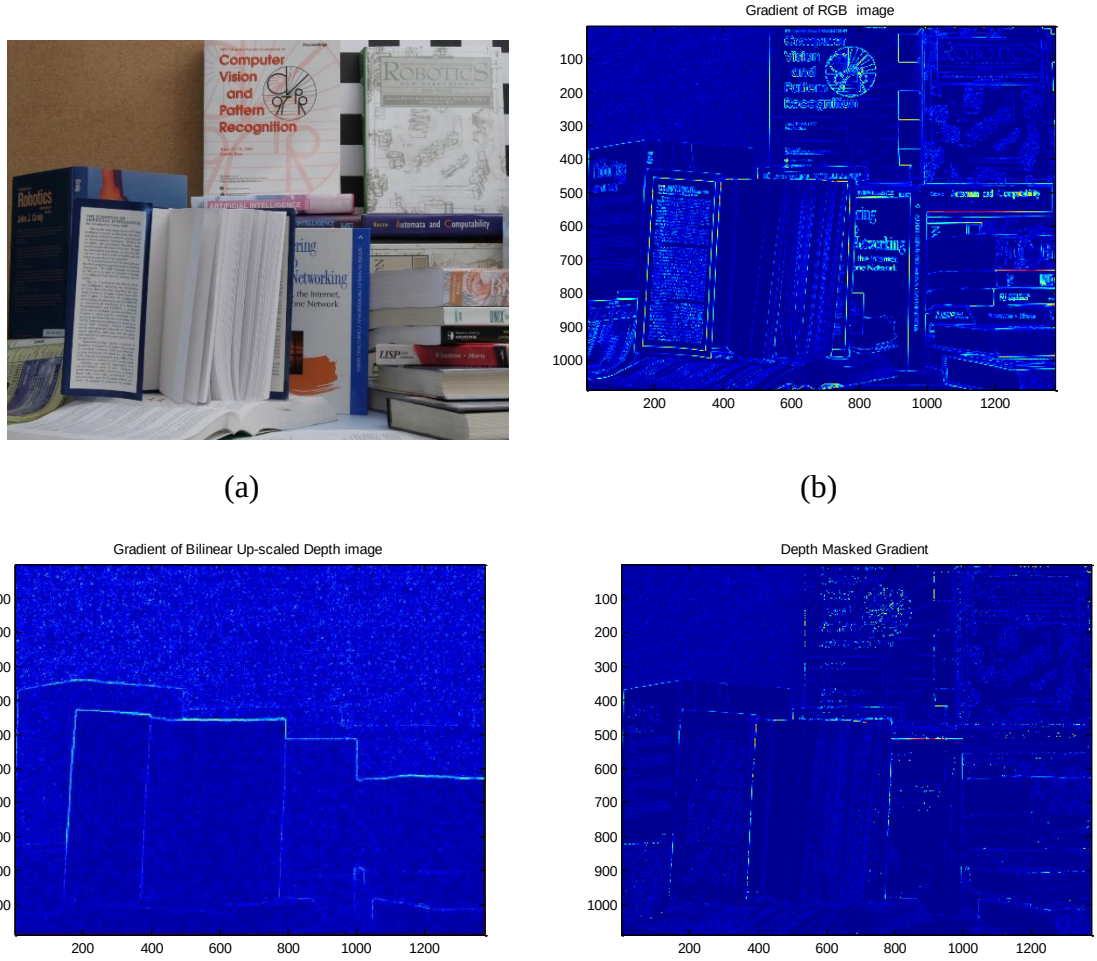


(a) (b)

(c) (d)

Figure 8: Gradient improvement image Art from **[20]** (a) High Resolution Image (b) Original Gradient (c) Gradient from bilinear upscaling of low-resolution depth map (d) Improved Masked Gradient.

Figure 9 shows the test case of the Middlebury image Books, in this case the text and diagrams from the cover of the books are removed in the masked gradient and the final MSE improvement is of 28.23 %.



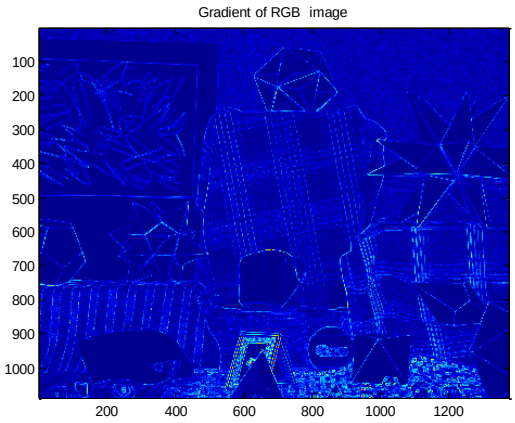
(a) (b)

(c) (d)

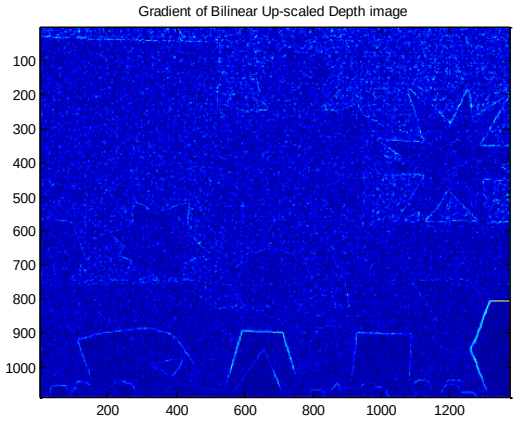
Figure 9: Gradient improvement image Books from **[20]**. (a) High Resolution Image (b) Original Gradient (c) Gradient from bilinear upscaling low-resolution depth map (d) Improved Masked Gradient.

The last gradient improvement is shown in Figure 10. The smallest improvement is shown in this case in which the decorative pattern of the fabrics is not well removed from the RGB image since the noise is affecting the gradient mask.

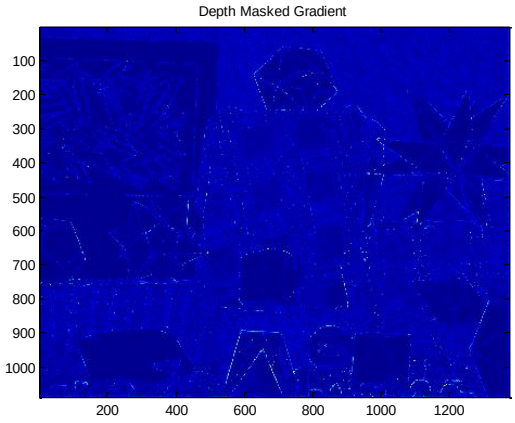
(a)



(b)



(c)



(d)

Figure 10: Gradient improvement image Moebius from **[20]**. (a) High Resolution Image (b) Original Gradient (c) Gradient from bilinear upscaling low-resolution depth map (d) Improved Masked Gradient.

Finally, Figure 11, Figure 12 and Figure 13 show zoomed regions of interest in which the non-object contours that were causing problems in the original algorithm. The artifacts are not

visible in the final solution of the proposed modification to the tensor, the masked gradient solution.

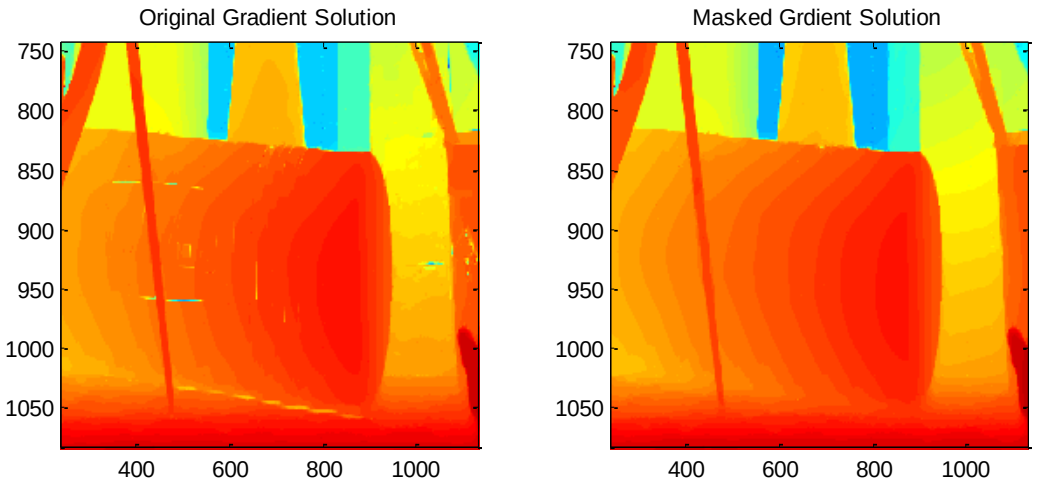


Figure 11: Art image, zoom at the bottom region.

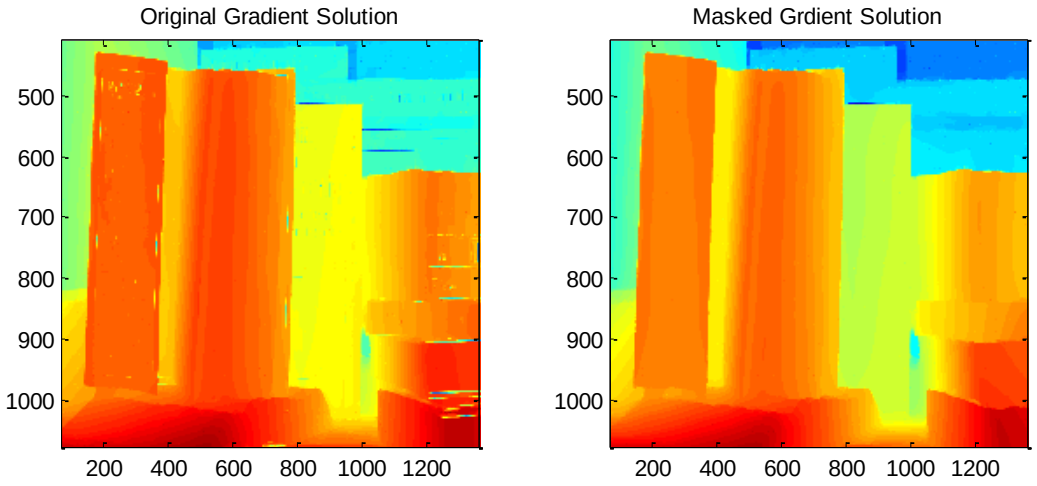


Figure 12: Books image, zoom at the low central region.

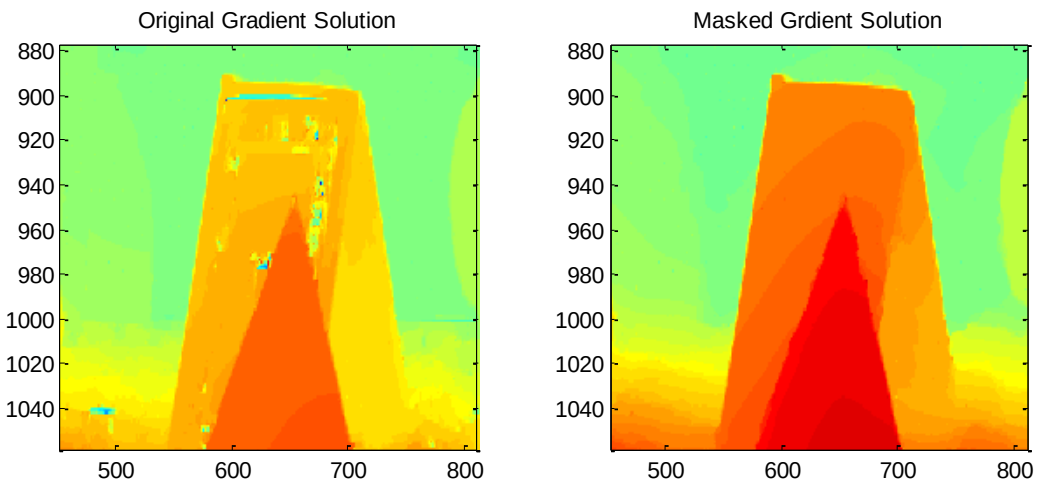


Figure 13: Moebius image, zoom at the low central region.

Chapter 3: Image Registration

The registration is the process to align different frames that capture the same scene but are located in a different point in the world coordinate space or even from different cameras or imager sensors. The registration process can also align frames of scenes in movement while the sensor remains in the same position.

The type of technique that is used for super-resolution application will determine the type of registration process that is required. The up-sampling process that fusions low resolutions frames from a ToF sensor with RGB high resolution images requires the type of registration that is presented in sections 3.1 through 3.3. This kind of registration to match both image planes uses the pin hole models of the cameras, their intrinsic and extrinsic parameter models, and a stereo calibration. The main advantage of this type of process is that once the system is calibrated the operations to register two frames is reduced to a matrix multiplication with a rotation matrix and the addition of a translation vector. The main disadvantage relies directly in the accuracy of the camera parameters, level of distortions and the precision of the stereo calibration.

On the other hand, methods that perform super-resolution using multiple frames from the same sensor can achieve the alignment using iterative techniques like the presented in section 3.4 based on the Lucas-Kanade algorithm [31]. The main disadvantage of this techniques is the required throughput to find the closest rotation and translation coefficients during an iterative process.

3.1 Manual Image Registration for Depth and RGB images

The image registration is one of the main steps in the up-scaling process since the precision of the results will affect directly the performance of the final part of the algorithm. A good registration will minimize the wrong assigned pixel from one image plane into the other image plane. Motivation for our developments is the result ofbad results obtained using Softkinetic co-registration parameters.

Basically in this work two registration methods are described. The first method consists of selecting several pairs of points from each image as shown in Figure 14. Even though this selection was performed manually, there are several methods to perform this automatically such as correlation-based methods and Feature-based methods as described in [12]. Once the registration is performed, a transformation matrix is generated which maps the low resolution depth image pixels into the high resolution RGB image plane coordinate system. The resulting sparse image is then applied to the region growing technique presented here.

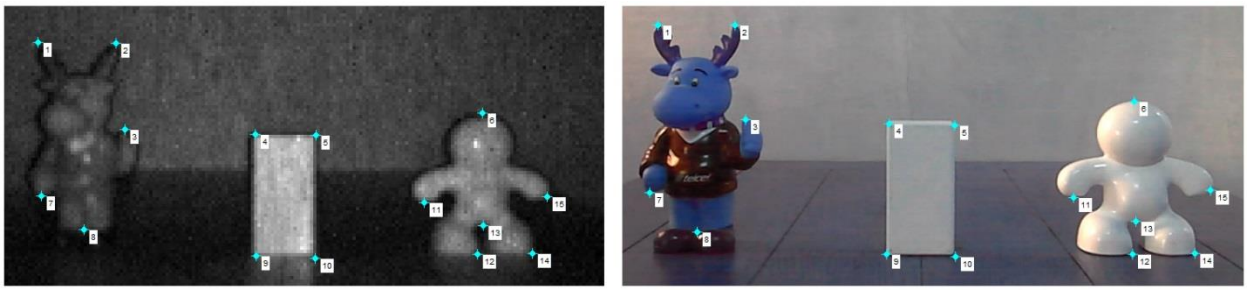


Figure 14: Manual Image registration. (a) Depth IR image selected points. (b) RGB image corresponding points.

It is important to mention that the registration was performed using the IR intensity image a. k. a. the Confidence image since it is easier to manually identify specific features on it and since it has exactly the same resolution and field-of-view as the depth image. The tranformation matrix

found can be used to create a sparse image of either the depth map or the confidence image. Generation of a sparse image from a low-resolution image is a generalization of integer upsampling. In sparse image creation, the image upscaling uses a pixel location determined by the co-registration procedure instead of simply using an integer multiple of the input pixel location. The remaining sparse image pixel values are then assigned the value zero. The transformation matrices that result saved are analyzed in the examples that follow.

3.2 Camera Model Registration for Depth and RGB images

The second method to register the depth image with the RGB image employs the camera model and its accuracy depends on how well the camera parameters are extracted and calibrated. The calibration method to extract the *intrinsic* and *extrinsic* camera parameters was performed using the tools from [27] which include tools for stereo calibration where the two cameras do not need to be the same. In our case, the ToF camera and the RGB camera are of very different resolutions, of course. The main advantage of using this method is that once the camera is calibrated, the mapping process is reduced to a set of equations that can be computed efficiently for each frame in real time applications.

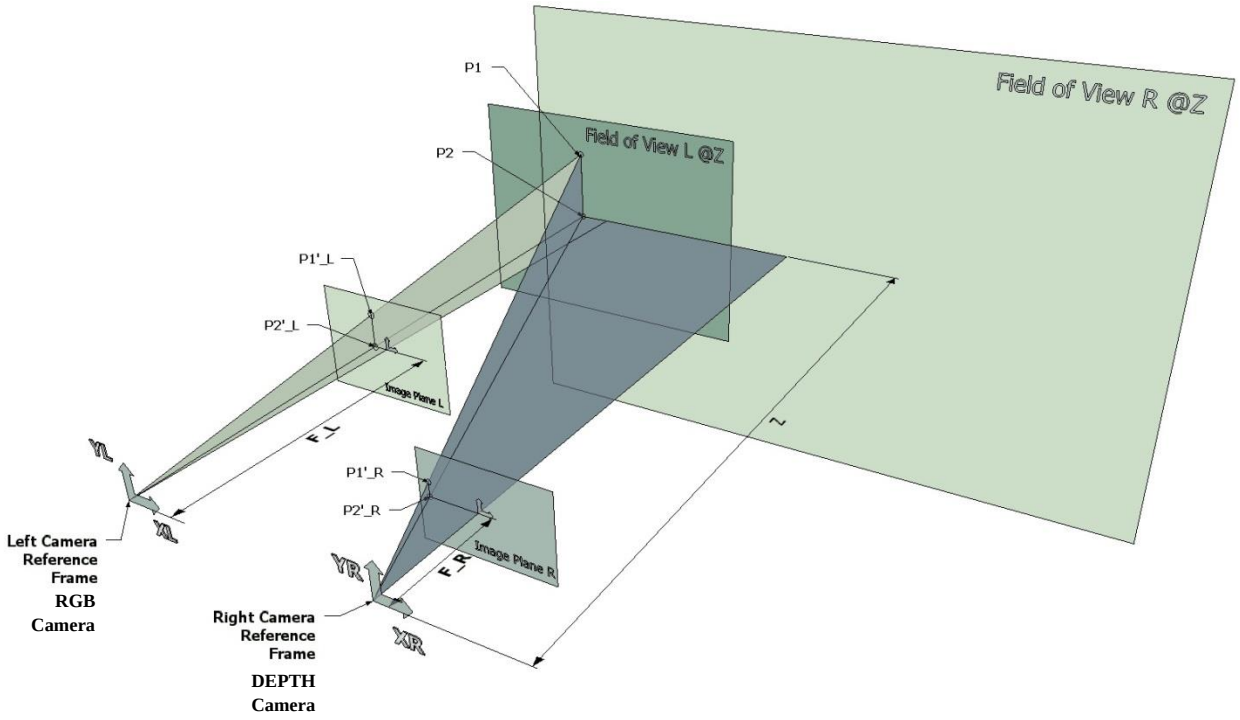


Figure 15: Stereo Camera System Geometry with parallel optical axis.

The camera model that was employed to map coordinates from one image plane into a second image plane is illustrated and described in the layout shown in Figure 15. For our purposes, the coordinate mapping problem also uses the confidence image from the ToF camera for easier matching of features with the RGB color/texture high-resolution image from the conventional co-located camera, see Figure 9.

Consider a point in the 3D world P1 with coordinates $[X_L \quad Y_L \quad Z_L]^T$ in the left camera reference frame and $[X_R \quad Y_R \quad Z_R]^T$ for the right camera reference frame. The relation between the image plane coordinates (u_L, v_L) and (u_R, v_R) and the 3D coordinates can be considered independently. For each camera, reference frames are defined using the perspective camera model in [12] chapters 2 and 6, which match with the equations provided with the SoftKinetic camera calibration user manual [32] as well as with the equations in [27].

RGB\Left camera:

$(u_L, v_L) \leftrightarrow [X_L \quad Y_L \quad Z_L]^T$

$u_L = \frac{f_L}{s_{xL}} \cdot \frac{X_L}{Z_L} + o_{xL}$

$v_L = \frac{f_L}{s_y} \cdot \frac{Y_L}{Z_L} + o_{yL}$

Depth\Right camera:

$(u_R, v_R) \leftrightarrow [X_R \quad Y_R \quad Z_R]^T$

$u_R = \frac{f_R}{s_{xR}} \cdot \frac{X_R}{Z_R} + o_{xR}$

$v_R = \frac{f_R}{s_{xR}} \cdot \frac{X_R}{Z_R} + o_{xR}$

It is worth pointing out that Z_L and Z_R are equal for Softkinetic camera system since the ToF and the conventional cameras are aligned right next to each other, as shown in Fig. 9.

Where f is focal length; s_x and s_y are the horizontal and vertical effective pixel sizes; and o_x and o_y are the coordinates of the image center. Since camera parameters are provided by [27] and [32] using the focal lengths in effective horizontal pixel units, the previous equations can be rewritten as:

RGB\Left camera:

$$u_L = f_{xL} \cdot \frac{X_L}{Z_L} + o_{xL} \quad (8)$$

$$v_L = f_{yL} \cdot \frac{Y_L}{Z_L} + o_{yL} \quad (10)$$

Depth\Right camera:

$$u_R = f_{xR} \cdot \frac{X_R}{Z_R} + o_{xR} \quad (9)$$

$$u_R = f_{xR} \cdot \frac{X_R}{Z_R} + o_{xR} \quad (11)$$

The camera calibration tools provided in [27] can generate a rotation matrix R and a translation vector T that can translate left camera reference frame coordinates into right camera reference frame coordinates. The specific tool to perform this task is the Stereo Camera Calibration

Toolbox. In this way, Equation (12) can be written to translate coordinates between the two reference frames.

$$\boldsymbol{X}_R = \boldsymbol{R} \cdot \boldsymbol{X}_L + \boldsymbol{T} \tag{12}$$

Since it is desired to link color image pixel coordinates with depth image coordinates, it will be possible to locate each depth image point in the high-resolution color image coordinate system. In our case, the color camera is the one located on the left side, thus it is necessary to solve for X_L from Equation (12). It is assumed that R is symmetric, orthonormal and invertible [Ref].

$$\boldsymbol{R}^{-1}(\boldsymbol{R} \cdot \boldsymbol{X}_L) = \boldsymbol{R}^{-1}(\boldsymbol{X}_R - \boldsymbol{T})$$

$$\boldsymbol{R}^{T}(\boldsymbol{R} \cdot \boldsymbol{X}_L) = \boldsymbol{R}^{T}(\boldsymbol{X}_R - \boldsymbol{T})$$

$$\boldsymbol{I} \cdot \boldsymbol{X}_L = \boldsymbol{R}^{T}\boldsymbol{X}_R - \boldsymbol{R}^{T}\boldsymbol{T}$$

$$\boldsymbol{X}_L = \boldsymbol{R}^{T}\boldsymbol{X}_R - \widehat{\boldsymbol{T}} \qquad \text{Where: } \widehat{\boldsymbol{T}} = \boldsymbol{R}^{T}\boldsymbol{T}$$

The next equation expands the vectors and rotation matrix to see their internal components:

$$\begin{bmatrix} X_L \\ Y_L \\ Z_L \end{bmatrix} = \begin{bmatrix} 1 & R_{1,2}^T & R_{1,3}^T \\ R_{1,2}^T & 1 & R_{2,3}^T \\ R_{3,1}^T & R_{3,2}^T & 1 \end{bmatrix} \begin{bmatrix} X_R \\ Y_R \\ Z_R \end{bmatrix} - \begin{bmatrix} \hat{T}_1 \\ \hat{T}_2 \\ \hat{T}_3 \end{bmatrix} \tag{13}$$

The reference frame mapping is described by the following:

(**RGB**\Left Ref. frame coordinates) = $R^T \cdot$ (**Depth**\Right Ref. frame coordinates) - $\widehat{\boldsymbol{T}}$

Combining Equations (8), (9), (10), (**11**) and (13) gives:

$$\begin{bmatrix} (u_L - o_{xL}) \cdot \frac{Z_L}{f_{xL}} \\ (v_L - o_{yL}) \cdot \frac{Z_L}{f_{yL}} \\ Z_L \end{bmatrix} = \begin{bmatrix} 1 & R_{1,2}^T & R_{1,3}^T \\ R_{1,2}^T & 1 & R_{2,3}^T \\ R_{3,1}^T & R_{3,2}^T & 1 \end{bmatrix} \begin{bmatrix} (u_R - o_{xR}) \cdot \frac{Z_R}{f_{xR}} \\ (v_R - o_{yR}) \cdot \frac{Z_R}{f_{yR}} \\ Z_R \end{bmatrix} - \begin{bmatrix} \hat{T}_1 \\ \hat{T}_2 \\ \hat{T}_3 \end{bmatrix} \quad (14)$$

Which gives the desired image coordinate mapping:

(**RGB**\Left image plane coord.) = R^T · (**Depth**\Right image plane coord.) - $\hat{\boldsymbol{T}}$

Finally, Equation (14) establishes the mapping relationship so that a pixel from the low-resolution depth image plane (subscript R) can be mapped into the high resolution RGB image plane (subscript L). A similar result is obtained in [33] though the level of detail is reduced.

3.3 Camera Calibration

The Softkinetic DS325 and DS311 camera drivers and firmware have available different services which provide intrinsic and external camera parameters for depth and color cameras. Furthermore, one of those services provide mapping of depth pixel into the high-resolution color camera coordinate system.

Unfortunately, using their tools the resulting sparse image is not correctly aligned with the high-resolution image. This is the main motivation to perform a camera calibration for each of the cameras so that correct camera parameters can be used to perform a correct image registration. It is beyond the scope of this research to implement an extraction technique for the intrinsic and extrinsic camera parameters, instead, the well-known and widely used Matlab toolbox [11] created

by Jean-Yves Bouguet was used. The following sections describe the procedure to calibrate both cameras separately and the final stereo calibration to get rotation matrix and translation vector to transform their coordinates form one image plane into the other.

3.3.1 Camera calibration with confidence (IR intensity) images.

The first step in the calibration process is to capture several pictures of a predefined grid in different positions with respect to the camera. The main purpose of this task is to track all the different vertices in the grid and in this way to quantify the optical performance of the camera lens, distortion and parameters. The higher the number of pictures in different positions that are used for the calibration, the better the result of the extracted parameters.

Figure 5.1 shows all the collected confidence images used for depth camera calibration. It is important to mention that for the extraction of depth camera parameters; the IR intensity or confidence image was used in the calibration process instead of the depth or phase values. This is valid since both values are collected form same sensor subject to the same physical layout of the lens, camera barrel, etc. Thus, the extracted parameters are totally valid for the depth image. Additionally, it is not possible to distinguish texture in the depth camera and the depth images only show the grid plane without the necessary square corners.

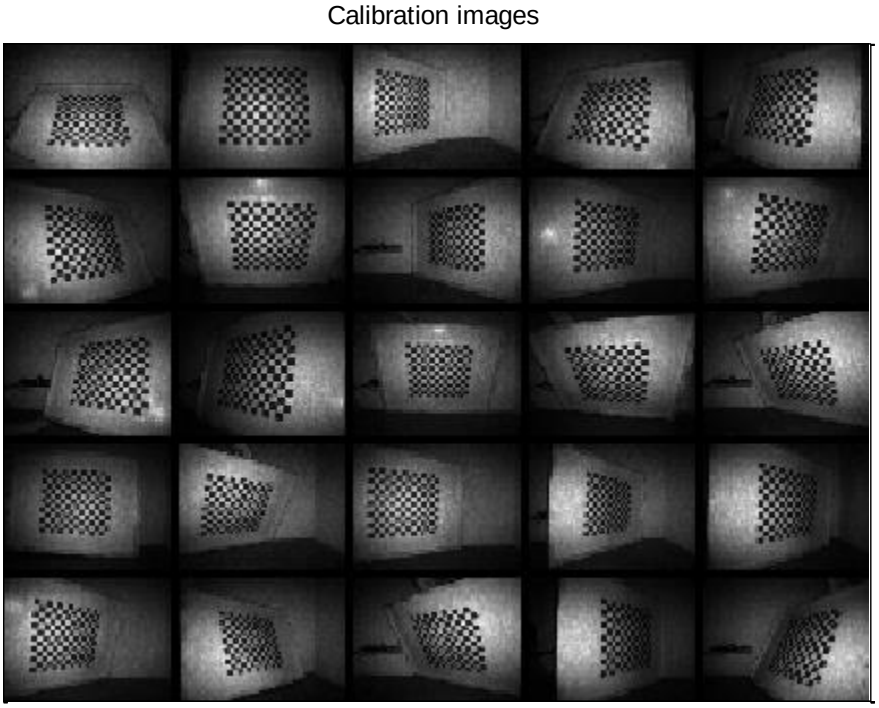


Figure 16: Confidence images of pattern grid in different orientations acquired with DS325.

After all images are collected and uploaded in memory to be ready for the tool usage, the next step is to identify all of the vertices in the grid on all the images. This process is semi-automatic since it is only required that one provide external corners and number of squares so that proportional division is applied as shown in Figure 17. In addition to this alignment, a corner finding algorithm is used so that the exact corners are located in the rage of a predefined window of size 7 by 7 pixels for the confidence images and of size 21 by 21 pixels for the high-resolution images as described further in the next section. However, if even after these processing steps there are still misaligned corners not corresponding to real corners in the image, manual adjustment can be performed. It is necessary to mention that the real square dimension in millimeters is employed by the calibration tool in [27] so that real dimensions in millimeters are reported for the found parameters. The width and height dimensions for the used pattern were of 27.4mm by 27.4mm;

and the total number of black and with squares is exactly the same number of squares used in [27] which is 14 by 13 squares.

Once all the corners are properly located, the calibration takes place using optimization techniques to solve the problem presented here. Namely to obtain the focal length in x and y, the center points, distortion parameters, and others parameters from the implicit information in the corner location across the different pattern orientations. Table 4 shows the obtained values for each parameter after the optimization process as well as a comparison with manufacturer values provided by the camera driver.

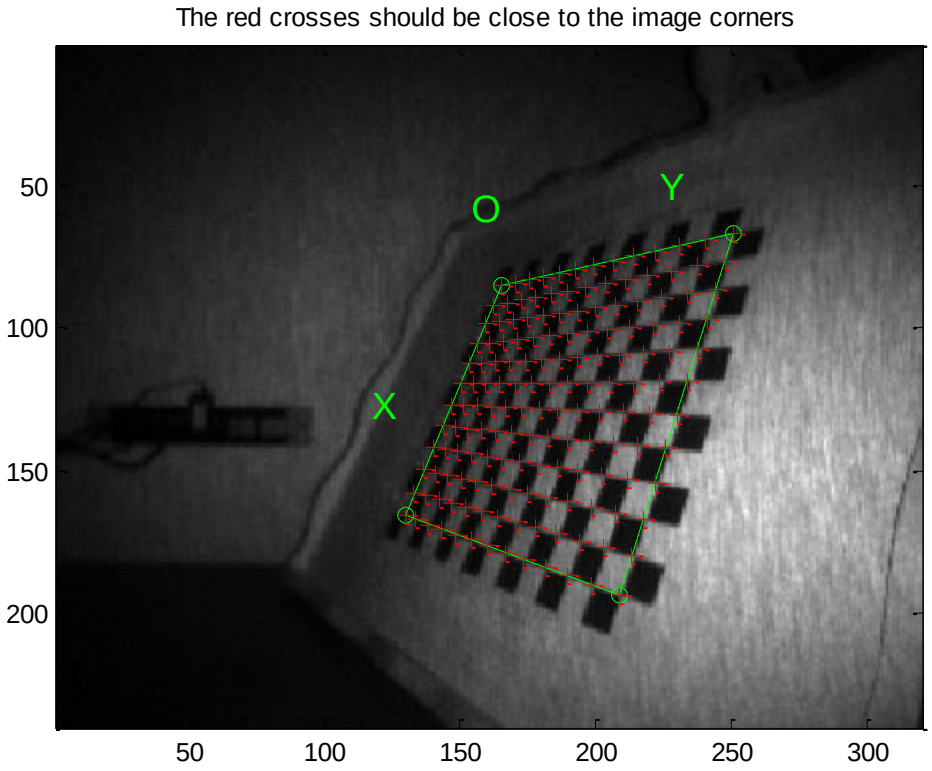


Figure 17: Example of initial corner detection in confidence image. Squares of 27.4 mm by 27.4 mm.

Table 4: Intrinsic Parameters: Depth Camera.

Intrinsic Parameters: Depth Right Camera							
Source	Focal length		Principal Point		Distortion Parameters		
	fc_x	fc_y	cc_x	cc_y	kc1	kc2	kc3
Calibration tool	228.858	226.873	156.997	115.687	-0.198	0.183	-0.003
Softkinetic	224.502	230.494	160.000	120.000	-0.170	0.144	-0.048

Finally, the extrinsic parameters were also computed. These parameters are useful in case it is necessary to map real world coordinates to the depth camera reference frame coordinates. Even though the translation vector and rotation matrix from the individual camera calibration are not directly used in this work, it is highly illustrative to look at the 25 checkerboard orientations based on the corner detections, as shown in Figure 18.

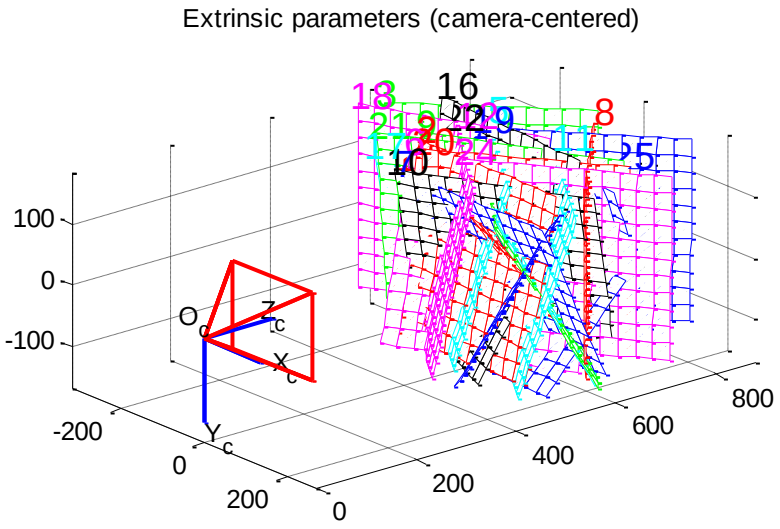


Figure 18: Check board orientations for depth right camera.

3.3.2 Color (RGB in gray scale) camera calibration

Exactly the same calibration was performed for the color left camera with the corresponding 25 images of the checkerboard pattern grid. The color images were first converted to gray scale (luminance) format. Figures 5.13 through 15 show the corresponding calibration process for the high-resolution camera. In the same way Table 5 presents the parameter values for the color camera in comparison with the parameters provided by the camera manufacturer.

Table 5: Intrinsic Parameters: Depth Camera.

Intrinsic Parameters: Color Left Camera							
Source	Focal length		Principal Point		Distortion Parameters		
	fc_x	fc_y	cc_x	cc_y	kc1	kc2	kc3
Calibration tool	988.816	980.327	608.426	355.133	-0.003	-0.043	-0.001
Softkinetic	963.394	986.950	640.000	360.000	0.027	-0.218	0.250

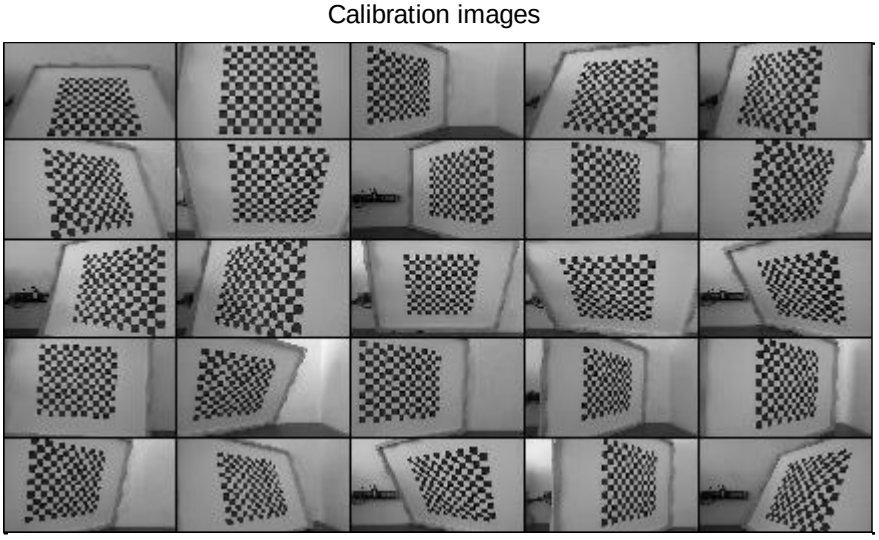


Figure 19: Color images of pattern grid in different orientations acquired with DS325.

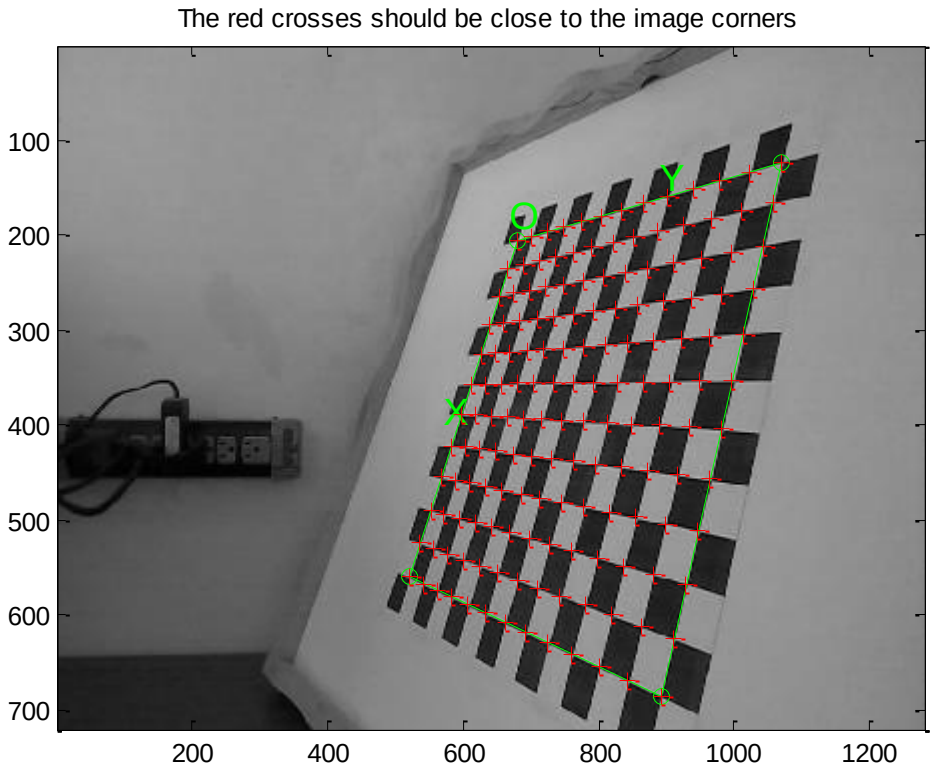


Figure 20: Example of initial corner detection in high resolution RGB image. Squares of 27.4 mm by 27.4 mm.

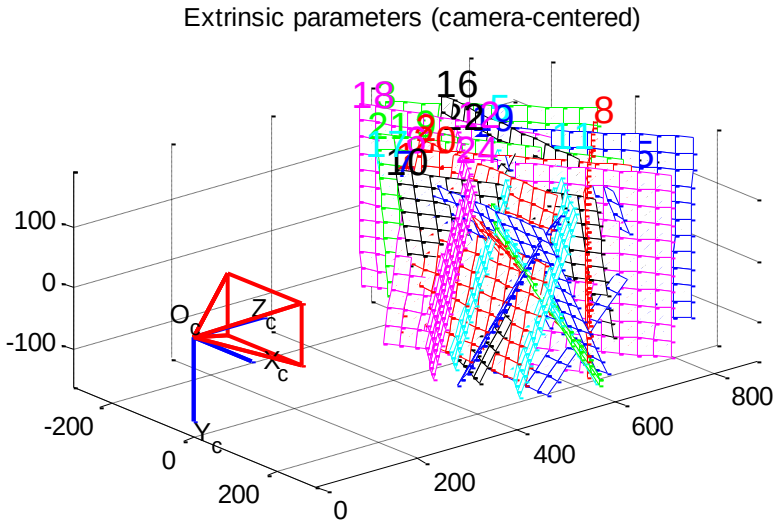


Figure 21: Check board orientations for high resolution RGB left camera.

3.3.3 Stereo calibration

The last step in the whole calibration process is to compute all the required values by Equation (12) so that it can be possible to map pixels from one image plane into the other image plane. The Matlab tool implemented in [27] also provides a stereo calibration option. The inputs for this tool are the intrinsic and extrinsic parameters of each camera presented in the last two sections. Figure 22 shows all the checkerboard orientations of each camera in the common real-world space coordinate system and how each camera location is identified by the process.

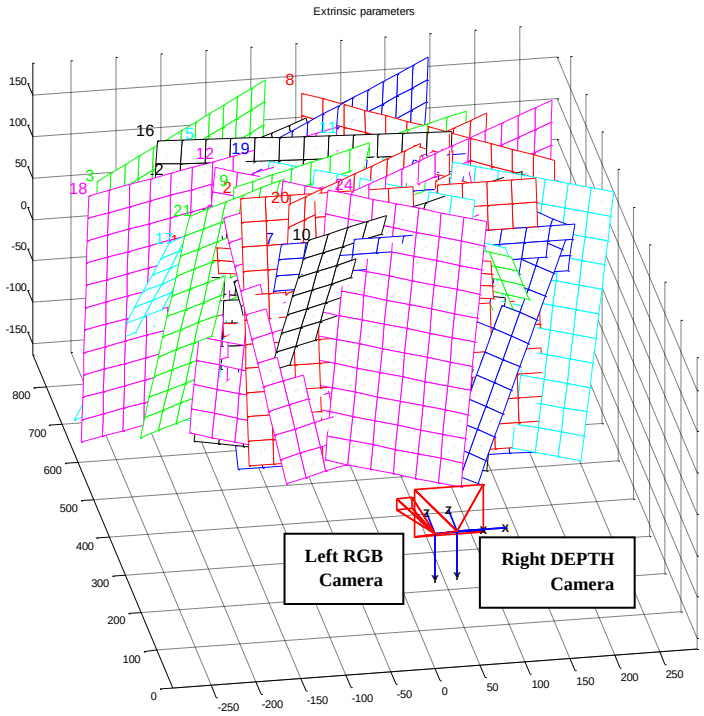


Figure 22: Checkerboard orientations from stereo calibration between color and depth cameras.

As result of this stereo calibration all the necessary information to compute the rotation matrix and translation vector shown in Equation (5.6) is generated and presented in Table 6.

Notice that the Rodrigues 3D rotation formula will take the om rotation vector values and build the rotation matrix as defined as:

$$R = rodrigues(om) = \begin{bmatrix} 1 & -om(3) & om(2) \\ om(3) & 1 & -om(1) \\ -om(2) & om(1) & 1 \end{bmatrix}$$

Table 6: Stereo Extrinsic parameters.

Vector	**Values**		
Translation	**T1**	**T2**	**T3**
	-24.720	0.045	-1.789
om	om(1)	om(2)	om(3)
	-0.00972	0.00084	-0.00030

Additionally, the intrinsic parameters of each camera are updated during the stereo optimization process; Table 7 shows the final results, which are slightly different to the ones presented in Table 4 and Table 5.

Table 7: Updated parameters from Stereo calibration.

Stereo Calibration; Updated Intrinsic Parameters: Color & Depth Cameras							
Source	Focal length		Principal Point		Distortion Parameters		
	fc_x	fc_y	cc_x	cc_y	kc1	kc2	kc3
Depth Camera	228.633	226.649	156.671	115.942	-0.199	0.184	-0.002
Color Camera	988.837	980.335	608.578	355.002	-0.003	-0.042	-0.001

An improved method has been developed by Ferstl in [34]. This technique is a modification of the previous shown method in this section with an add on installed into the Caltech tool box [27] in which the Chess pattern is replaced by an array of black spots by which the issues finding

corners in the low-resolution frames is solved and the accuracy of the calibration is improved. Figure 23 shows the proposed pattern and present the system overview and process.

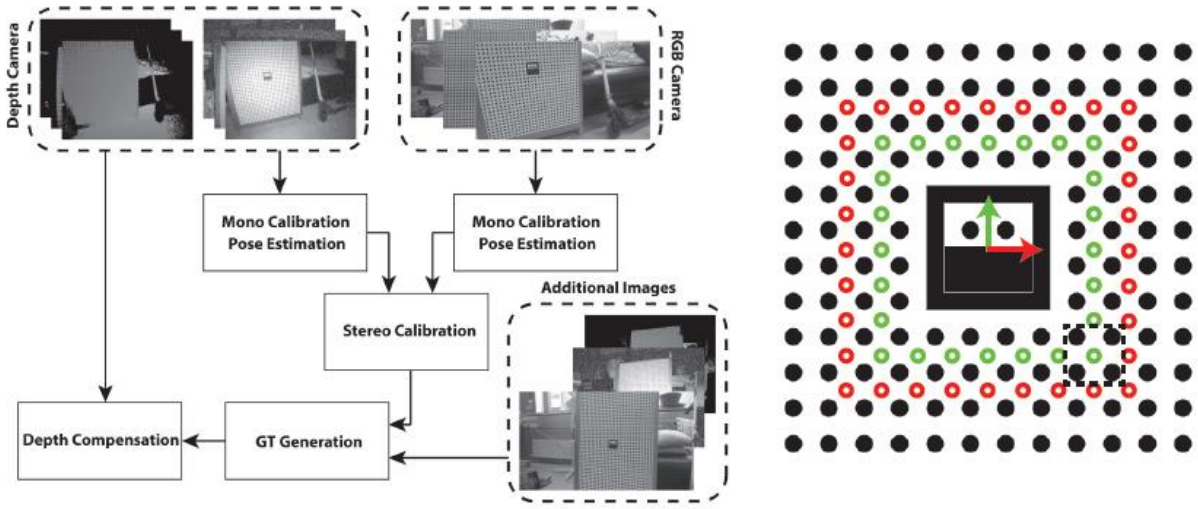


Figure 23: Alternative ToF Camera calibration system overview and proposed pattern by [34].

3.4 Multi-frame Registration for Depth images

The Multi-frame registration is a key step in the super-resolution process. In contrast with the registration from different sensors described in the previous sections, the input frames to this process are usually images from the same sensor but collected at different times and slightly different locations. That is, from a slightly shifted and rotated point in the world space with respect to previous frame. Typically, the source of frames comes from a video sequence.

The technique used to perform the Multi-frame registration step is based on the work presented in [35] and [36] with slight modifications and taking advantage of the original idea from Bruce Lucas and Takeo Kanade [31] for the initial estimation of displacements and rotation from the low resolution observations.

On the one hand, the registration algorithm developed in [31] doesn't provide accurate estimation if the shifts and rotations are too big unless the observations have certain level of blurring at the cost of a good match of small details which are suppressed. Thus, it is suggested from the authors of this method to perform two stages, one initial estimation with blurred observations to get an approximate match and then a second stage with better resolution to refine the match gotten initially. This behavior in the Lucas-Kanade method is intrinsic to its basic formulation in which it is assumed a small enough disparity **h** so that the one-dimensional approximation, shown in following simplified equation, is adequate. This requirement can be achieved to an extended range of h by smoothing the images.

$$h \approx \frac{G(x) - F(x)}{F'(x)}$$

Where functions G(x) and F(x) provide pixel value at location x. Then the algorithm will minimize the error defined as:

$$E = \sum_x [F(x+h) - G(x)]^2$$

Solving for the disparity h:

$$h \approx \frac{\sum_x F'(x)[G(x) - F(x)]}{\sum_x F'(x)}$$

The iterative disparity formula improved with a weighting factor to minimize estimation error in the regions where $F''(x)$ is high is given by:

$$h_{k+1} = h_k + \frac{\sum_x w(x)F'(x+h_k)[G(x) - F(x+h_k)]}{\sum_x w(x)F'(x+h_k)^2}$$

Where the weighting factor is:

$$w(x) = \frac{1}{|G'(x) - F'(x)|}$$

Finally, the estimation error generalization for two dimensions incorporating x-y shifts and rotation between two images is provided in [31] as:

$$E = \sum_{x} [F(xA + h) - (\alpha G(x) + \beta)]^2$$

Where the matrix A represents the linear spatial transformation between F(x) and G(x) that contains the rotation, scaling and shearing, and α and β are contrast and brightness adjustments respectively.

3.4.1 Registration Process Implementation

This method is employed in the proposed iterative framework to perform a first registration over low-resolution bicubic interpolated up-scaled frames previously filtered to remove impulsive noise, characteristic of flickering pixels caused by a ToF device. Once this initial registration is completed, the rest of the algorithm will use these parameters to start the subsequent registrations in the upscaling process. The initial registration as previously mentioned is not accurate but provides the first set of shift and rotation parameters for the proposed framework.

The registration will be executed again in every noise detection loop, but in every iteration the registration will be applied over the high-resolution image improved by the up-scaling method

such as the ADMM TGV, which is described in the following sections. The overall algorithm is as follows:

Algorithm 1 Registration and Upscaling Method

Require: LR images

1. Set initial conditions for shift and rotation estimations: sx = 0; sy=0; θ=0.
2. If S&P impulsive noise then Filter LR images with amf filter.
   Else Filter LR images with acwf2.
3. **Do Registration against initial image u_0 & update: sx, sy & θ.**
4. Do Initial Impulsive Noise Detection
5. k=0;
6. While k < max_iter do
   a. Up-scale image u using ADMM TGV method
   b. Detect Remaining Impulsive noise.
   c. **Do Registration against u_k & and update: sx, sy & θ.**
   d. k=k+1
7. Return final solution u.

As it can be in the above algorithm overview, the registration process is part of the upscaling algorithm and its accuracy it is improved in every new iteration.

Figure 24 shows the case of 10 aligned frames generated form the depth map Art of the Middlebury dataset [20]. Deeper description of the alternating methodologies for registration estimation and super-resolution can be found in [37] and [38].

The 10 low-resolution frames were initially corrupted with 10% of the pixels with Salt and Pepper impulsive noise, Gaussian noise with sigma=3 and with maximum absolute shifts of sx = 1.3135 pixels, sy = 1.2246 pixels; theta = 0.0273 rad. The alignment results are shown in the up-scaled image after 1 and 20 iterations. The estimation error norm against known ground truth parameters is computed for every frame and the average is recorded to present the evolution across the algorithm iterations as shown in Equation (15).

$$MSE_{Reg}(iter) = \frac{1}{L}\sum_{i=1}^{L} \|\hat{p}_i - p_i\|^2 \quad (15)$$

Where L is the number of frames, $\hat{p}_i$ are the estimated parameters and p_i are the ground truth parameters of each image.

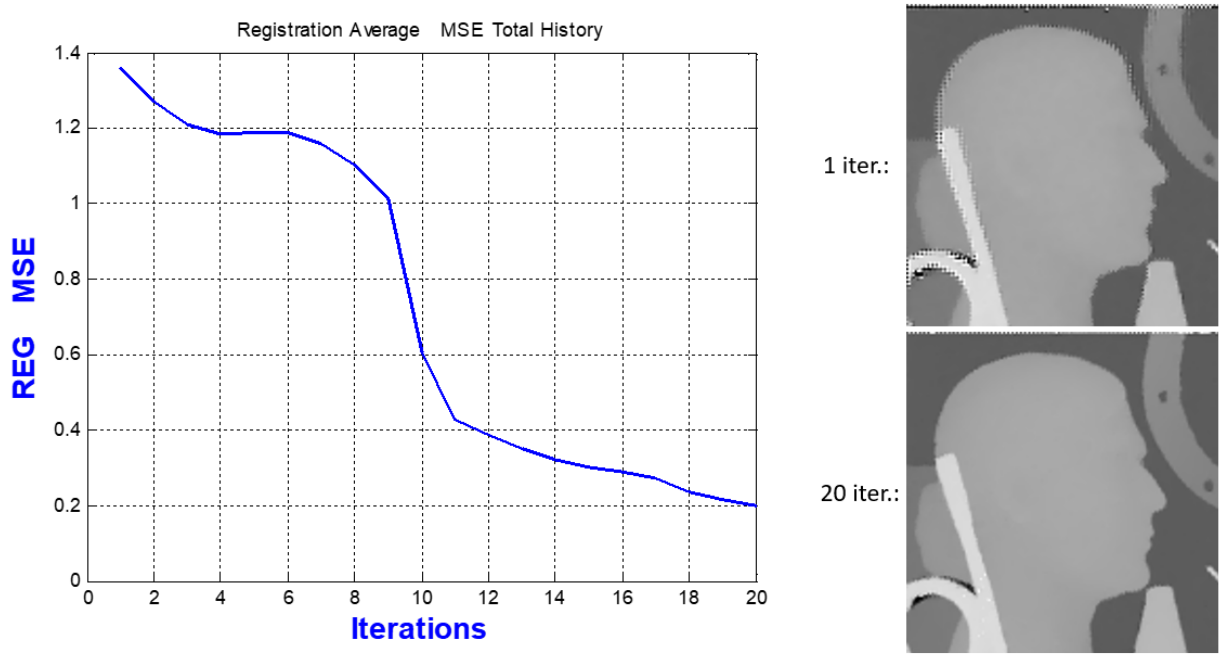


Figure 24: 10-Depth Frames Registration Mean Square Error example using Lucas-Kanade algorithm in an iterative process in conjunction with ADMM-TGV.

3.4.2 Registration Characterization

The Lukas-Kanade registration in conjunction with ADMM-TGV iterative process was characterized across several magnitudes of shifts and rotations to provide a margin of the maximum spatial change between the reference and other frames that can be properly registered. This amount of motion between frames is a key factor to regard a minimum frame per second (fps) rate of the required equipment for a given system as well as to consider the maximum speed of an object moving in a scene in a real application.

The simulated motion is generated with the original code developed by [36]. The x and y shifts are computed with the following Equations (16) and (17) respectively while the rotations are computed in radians with Equation (18). These displacements are in the high resolution image plane pixel units since a ground truth image in high resolution is transformed with these parameters and then down-sampled in a simulation example.

$$Shift_x = MaxShift \cdot r_1 - MaxShift \cdot r_2 \tag{16}$$

$$Shift_y = MaxShift \cdot r_3 - MaxShift \cdot r_4 \tag{17}$$

$$Rotation = \sqrt{MaxRot} \cdot r_5 - \sqrt{MaxRot} \cdot r_6 \tag{18}$$

In Equations (16)-(18) MaxShit and MaxRot are parameters to define maximum possible shift and rotation and r_1-r_6 are random numbers in a range of [0,1] with a normal distribution and zero mean. Table 8 defines 6 different motion configuration M1-M6 that simulates movements of the acquisition device and it presents the final performance of the registration scheme using PSNR

and SSIM metrics. Additionally, every experiment processes 10 low resolution frames contaminated with Gaussian noise with sigma=3 and 10% of pixels with Salt and Pepper impulsive noise.

It is important to clarify that the random numbers r_1-r_6 were generated in each experiment with the same seed so that a more logical comparison between experiments can be done.

Additionally, Figure 25 plots the evolution of the 6 motion simulations across a total of 20 iterations. It can be inferred that the registration algorithm has only acceptable levels of PSNR and SSIM in simulations M1 and M2. Unfortunately, this is one of weaknesses of the current framework when the shift and rotation parameters are unknown, which is the scenario of real applications in which the acquisition device is under random motion that needs to be estimated.

Table 8: Motion Simulation Results.

Exp.	Max-Shift	Max-Rot	Displacements					PSNR	SSIM
			Max. x [HR pixel]	Max. y [HR pixel]	Max. x [LR pixel]	Max. y [LR pixel]	Max Rot. [degrees]	dB	%
M1	1	1	0.4280	0.8878	0.2140	0.4439	0.6326	39.4246	0.9879
M2	2	3	0.8560	1.7756	0.4280	0.8878	1.0957	35.7867	0.9813
M3	3	4	1.2840	2.6635	0.6420	1.3318	1.2652	24.3298	0.8488
M4	4	6	1.7120	3.5513	0.8560	1.7757	1.5495	22.8876	0.8154
M5	5	7	2.1400	4.4391	1.0700	2.2196	1.6737	22.4151	0.7913
M6	6	9	2.5680	5.3269	1.2840	2.6635	1.8978	21.9544	0.7608

The work presented in [36] provides to the Lucas-Kanade algorithm initial conditions extracted directly from the random values computed by equations (16) through (18). These values help dramatically to the algorithm to perform a much better estimation, accepting even maximum shifts and rotations in the order of Maxshift=20 and MaxRot=30. Nevertheless, these values are not available in a real application, thus in the present framework such initial conditions are set to

0. Although this performance could discourage the real application of the developed algorithm, the real natural images acquired from videos in [35] and [36] as well the sequences tested in the following sections of this document show good objective results since the amount of motion is in the range of the simulation like in M2.

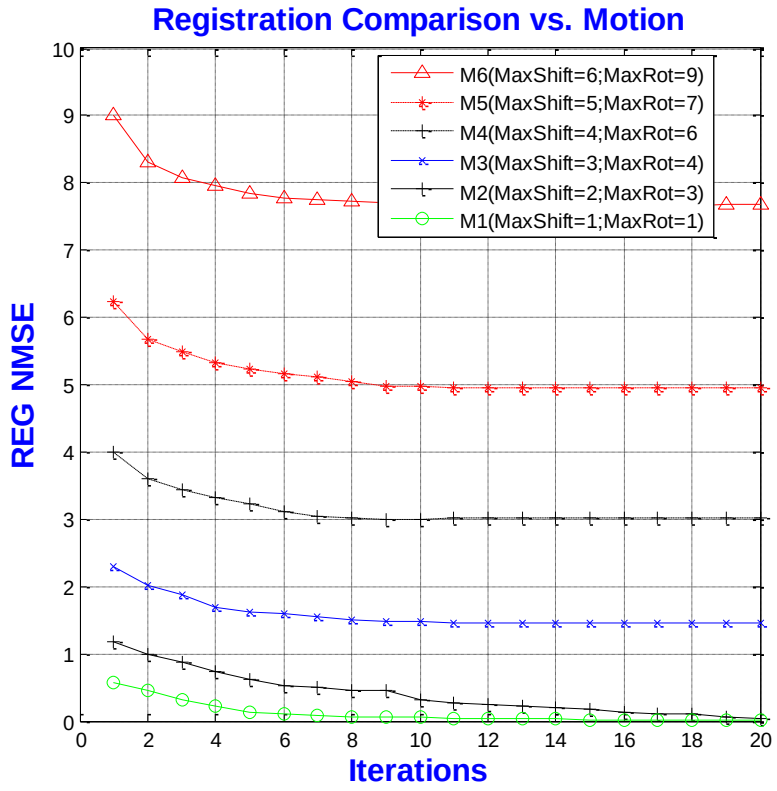


Figure 25: Registration error evolution under 6 Motion Simulations.

Chapter 4: Super-Resolution (SR) and Inpainting for Depth Images

The general proposed frame work to up-scale depth images is described in this chapter. The first part describes the Super resolution model as well as the stage of the algorithm that corresponds to outlier detection or impulsive noise addressed as an inpainting problem of the pixels with missing information. The second section describes the regularization stage and the functional that was evaluated.

4.1 Multi-frame Super-Resolution Model

4.1.1 Image Observation Model and Fidelity term

The Multi-Frame Super-Resolution (MFSR) Model is described in this section, which is based in the model used in [35]. First of all, regard a group of L number of low resolution images Y_i, i=1…L, and a set of reconstruction matrices H_i i=1…L, as follows:

$$\boldsymbol{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_L \end{bmatrix}; \qquad \boldsymbol{H} = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_L \end{bmatrix}$$

Consider the size of the low-resolution observation image Y_i to be n by n pixels, which in column vector form is just the n^2 total number of pixels, $n^2 \times 1$, for implementation purposes.

Additionally, it is necessary to regard the product HX between the reconstruction matrix H and the unknown high-resolution square image X, which has a total number of pixels $N^2 = r^2n^2$. X is actually magnified by an up-sampling factor of r^2, in the vector form too, respect to the low-resolutions observations size n^2. In the same way, the vectorized size of X is $r^2n^2 \times 1$.

$$\boldsymbol{HX} = \begin{bmatrix} H_1X \\ H_2X \\ \vdots \\ H_LX \end{bmatrix}$$

The image observation model used in [35] is a well-known scheme that defines the L reconstruction matrices H_i in terms of linear operators: down-sampling matrices D_i, blurring matrices B_i, and rotation and shift matrices S_i, in the following way:

$$H_i = D_i B_i S_i, \qquad i = 1 \cdots L.$$

Note that D_i is of size $n^2 \times r^2 n^2$, B_i is of size $r^2 n^2 \times r^2 n^2$ and S_i is of size $r^2 n^2 \times r^2 n^2$, which provokes sizes of the reconstruction matrices H_i of $n^2 \times r^2 n^2$. These dimensions require an efficient use of memory as well as conversion of some variables to sparse types to avoid memory out issues in the Matlab algorithm implementation.

It is necessary to establish that the observation model assumes the *a priori* knowledge of an amount of blur in the images as well as shifts in x and y and rotation in the image spatial domain. The blur amount is neglected in this work but it can be studied for applications that deal with low acquisition rates or low frames per second (fps) systems in which the amount of blur due to fast motion is present. The shift and rotation estimation from frame to frame is estimated using the technique described in section 3.4 Multi-frame Registration for Depth images, which is an essential part of the algorithm and of the iterative process.

In this way, the low-resolution observations can be related mathematically by the model with the unknown high-resolution image X with the following expression:

$$Y_i = D_i B_i S_i \boldsymbol{X} + V_{\boldsymbol{i}}, \qquad i = 1 \cdots L.$$

where: $V_{\boldsymbol{i}}$ is an additive noise term of size $n^2 \times 1$.

Then a fidelity term respect to the measured data of the target functional can be built using the norm 2 in the following way:

$$\|HX - Y\|_2^2 = \sum_{i=1}^{L} \|H_i X - Y_i\|_2^2 \tag{19}$$

This fidelity term doesn't include the impulsive noise part yet. Continuing with the construction of the fidelity term some terms that will be necessary in the next derivations, lets compute the first and second derivatives, this is the Gradient and Hessian of Equation (19):

$$\nabla \|HX - Y\|_2^2 = \sum_{i=1}^{L} {H_i}^T (H_i X - Y_i) \tag{20}$$

$$\nabla^2 (\|HX - Y\|_2^2) = \sum_{i=1}^{L} {H_i}^T H_i = H^T H \tag{21}$$

4.1.2 Regularization Term

The next step is to define the regularization term. The regularization type of term proposed for this research is the Total Generalized Variation (TGV) semi-norm proposed by Bredies in [39]:

$$TGV_{\alpha}^{k}(u) = sup\left\{\int_{\Omega} u\ div^{k} v\ d\mathbf{x} \mid v\ \epsilon C_{c}^{k}\left(\Omega, Sym^{k}(\mathbf{R}^{d})\right),\ \|\text{div}^{l} v\|_{\infty} \leq \alpha_{l}, l = 0, \dots, k-1\right\}$$

As mentioned in [21], the second order TGV is enough for the regularization term of the depth up-scaling applications, since most object piecewise affine surfaces and shapes can be well-approximated. Thus, the previous expression for k=2 defines the second order TGV expression:

$$TGV_{\alpha}^{2}(u) = sup\left\{\int_{\Omega} u\ div^{2} v\ d\mathbf{x} \mid v\ \epsilon C_{c}^{2}\left(\Omega, Sym^{2}(\mathbf{R}^{d})\right), \|v\|_{\infty} \leq \alpha_{0}, \|\text{div}\ v\|_{\infty} \leq \alpha_{1}\right\}$$

where:

$Sym^{2}(\mathbf{R}^{d}) = S^{dxd}$, equivalent to the set of symmetric matrices;

$$(\text{div}\ v)_{i} = \sum_{j=1}^{d} \frac{\partial v_{ij}}{\partial x_{j}},\ 1 \leq i \leq d; \qquad \text{div}^{2}\ v = \sum_{i,j=1}^{d} \frac{\partial^{2} v_{ii}}{\partial {x_{j}}^{2}}$$

$$\|v\|_{\infty} = sup\left(\sum_{i,j=1}^{d} |v_{ii}(x)|^{2}\right)^{\frac{1}{2}} \qquad \|\text{div}\ v\|_{\infty} = sup\left(\sum_{i=1}^{d} |v_{ii}(x)|^{2}\right)^{\frac{1}{2}}$$

The Alternating Direction Method of Multipliers (ADMM) numeric method was selected to solve the optimization problem that contains this functional. This ADMM implementation is based in the development presented by Chuan He in [40] for denoising purposes. Furthermore, the problem in [40] is defined in an equivalent way based on the Lagrange duality. This transformation makes possible the implementation of the numerical method via the ADMM to solve this problem.

First of all, He departs from the discrete version of the 2nd order TGV:

$$TGV_{\alpha}^{2}(u) = \min_{\mathbf{p}}\{\langle u, div\ v\rangle |\ div\ v = d,\ \|v\|_{\infty} \leq \alpha_0, \|d\|_{\infty} \leq \alpha_1\}$$

where:

$$\|v\|_{\infty} = \max_{i,j}\left(v_{i,j,1}^{2} + v_{i,j,2}^{2} + v_{i,j,3}^{2}\right)^{\frac{1}{2}}$$

$$\|d\|_{\infty} = \max_{i,j}\left(d_{i,j,1}^{2} + d_{i,j,2}^{2}\right)^{\frac{1}{2}}$$

Then, the following expression for the functional is obtained (a detailed derivation can be found in the appendix A of [40]):

$$TGV_{\alpha}^{2}(u) = \min_{\mathbf{p}} \alpha_0\|\varepsilon(\mathbf{p})\|_1 + \alpha_1\|\nabla u - \mathbf{p}\|_1 \tag{22}$$

where: $\mathbf{p} \in \mathbf{R}^{mn} \times \mathbf{R}^{mn}$ belongs to the two-dimensional 1-tensor field. The $p_{i,j}$ denotes the (i,j)th components of p and $\mathbf{p}_{i,j} = [p_{i,j,1}, p_{i,j,2}]$. Next is defined the (i,j)th component of the matrix $\varepsilon(\mathbf{p})$ as:

$$\varepsilon(\mathbf{p}) = \begin{bmatrix} \varepsilon(\mathbf{p})_{i,j,1} & \varepsilon(\mathbf{p})_{i,j,3} \\ \varepsilon(\mathbf{p})_{i,j,3} & \varepsilon(\mathbf{p})_{i,j,2} \end{bmatrix} = \begin{bmatrix} \nabla_{x_1} p_{i,j,1} & \dfrac{\nabla_{x_1} p_{i,j,1} + \nabla_{x_1} p_{i,j,2}}{2} \\ \dfrac{\nabla_{x_1} p_{i,j,1} + \nabla_{x_1} p_{i,j,2}}{2} & \nabla_{x_2} p_{i,j,2} \end{bmatrix}$$

where: $\varepsilon(\mathbf{p})$ is a symmetric 2 tensor and the operators ∇_{x_1} and ∇_{x_2} compute differences in the x_1 and x_2 directions respectively.

In addition to the regularization term defined by Equation (22), a variant will be evaluated by introducing a Tensor T in the regularization term which will be discussed in the next sections. At this point it is just defined as the Tensorial TGV (TTGV):

$$TTGV_{\alpha}^{2}(u) = \min_{\mathbf{p}} \alpha_0 \|\varepsilon(\mathbf{p})\|_1 + \alpha_1 \|\mathrm{T}(\nabla u - \mathbf{p})\|_1 \tag{23}$$

where T is a structure tensor defined in the next sections.

The norm 1 operator $\|\cdot\|_1$ used in equations (22) and (23) is defined as follows. First consider a given vector x:

$$\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$$

If the vector components are scalars, then the norm 1 is:

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|$$

If the x_i components of x are vectors, then:

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{n} \|x_i\|_2$$

If the x_i components of x are matrices, then:

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{n} \|x_i\|_F$$

where the $\|x_i\|_F$ extends the norm 2 of vectors, the Frobenius norm.

4.1.3 Super-Resolution Functional and Bayesian Inference

Once the regularization term was initially defined, it can be introduced into the proposed functional to be minimized:

$$X^* = \underset{X}{\operatorname{argmin}} F(X) = \frac{\lambda}{2}\|HX - Y\|_2^2 + TTGV_\alpha^2(X) \tag{24}$$

Where λ is the regularization parameter that provides the balance between the fitting term and the smoothing term. This is between the fidelity term and regularization term. The stated problem by Equation (24) can be seen from a Bayesian philosophy point of view, this is employing the Bayesian inference formulation of the solution. This is treating the high-resolution image X as samples of random fields with prior distributions that describes the knowledge of it and considering the components of the transformation matrices H_i as known parameters previously estimated. The low-resolutions observations Y_i are also considered as samples of a random field that is related to the process of obtaining X via the conditional distribution. This well-known approach is detailly described in [37] and [38], but with the difference that the motion vectors contained in the matrices H_i are not included. Now, here is just entitled the basic posterior modeling which defines the probability of obtaining a high-resolution image X given certain observations Y:

$$\mathbf{P}(\boldsymbol{X}|\boldsymbol{Y}) = \frac{\mathbf{P}(\boldsymbol{X})\mathbf{P}(\boldsymbol{Y}|\boldsymbol{X})}{\mathbf{P}(\boldsymbol{Y})}$$

Where: $\mathbf{P}(\boldsymbol{Y}|\boldsymbol{X})$ is the likelihood of the observation.

$\mathbf{P}(\boldsymbol{X})$ is the prior.

The simplest form of the Bayesian inference that provides the maximum a posteriori (MAP) solution maximizes the posterior probability distribution described above with the following expression:

$$\hat{X} = \underset{X}{\operatorname{argmax}}\, \mathbf{P}(\boldsymbol{X}|\boldsymbol{Y})$$

$$\hat{X} = \underset{X}{\operatorname{argmax}} \left\{ \frac{\mathbf{P}(\boldsymbol{Y}|\boldsymbol{X})\mathbf{P}(\boldsymbol{X})}{\mathbf{P}(\boldsymbol{Y})} \right\}$$

Since the probability of the low-resolution observations P(Y) is independent of the maximization variable X, then the above expression can be rewritten as:

$$\hat{X} = \underset{X}{\operatorname{argmin}} \{ \mathbf{P}(\boldsymbol{Y}|\boldsymbol{X})\mathbf{P}(\boldsymbol{X}) \} \tag{25}$$

$$\hat{X} = \underset{X}{\operatorname{argmin}} \{ -\log \mathbf{P}(\boldsymbol{Y}|\boldsymbol{X}) - \log \mathbf{P}(\boldsymbol{X}) \} \tag{26}$$

Now, the likelihood P(Y|X) and the prior P(X) can be defined as:

$$\mathbf{P}(\boldsymbol{Y}|\boldsymbol{X}) = e^{-\lambda \|HX - Y\|_2^2}$$

$$\mathbf{P}(\boldsymbol{X}) = e^{-TTGV_\alpha^2(X)}$$

Finally, by substituting these two last definitions in Equation (25) it is obtained:

$$\hat{X} = \underset{X}{\operatorname{argmin}} \left\{ -\log e^{-\lambda \|HX - Y\|_2^2} - \log e^{-TTGV_\alpha^2(X)} \right\}$$

$$\hat{X} = \underset{X}{\operatorname{argmin}} \{ \lambda \|HX - Y\|_2^2 + TTGV_\alpha^2(X) \}$$

For practical purposes during the minimization, a constant ½ is added without affecting the overall process, getting the Equation (24) originally proposed:

$$\hat{X} = \underset{X}{\operatorname{argmin}} \left\{ \frac{\lambda}{2} \|HX - Y\|_2^2 + TTGV_{\alpha}^2(X) \right\}$$

Since close forms to solve the above expression can be analytically difficult, numerical methods employing an alternating minimization can be employed as initially described in the registration section first estimating the shifts and rotations contained in H and then estimating the high-resolution image X. Once H is estimated in every iteration, then these updated values can be used to estimate the next high-resolution image approximation.

4.2 Impulsive Noise Detection and Inpainting with SR

The depth images have intrinsic defects by the nature of their acquisition process in addition to those mentioned in Section 1.3 Time-of-Flight Cameras. Such defects can be generated by occlusion and geometry of the objects in the captured scene, by low reflected signal amplitude or low confidence in the sensing, by concave cavities, as well as by saturation of reflected signals from objects very close to the camera lens. All of these defects are processed by the implemented software as 0 level values, which are displayed in the gray scale depth images as black pixels for most to the camera drivers.

Argáez, Sanchez and Ramirez present a novel ℓ_1-algorithm for undetermined systems and its applications in [41], in which they implemented the Path-Following Signal Recovery (PFSR) method which takes advantage of sparse representation of an image as well as wavelet orthogonal properties to restores images as shown by them in [41]. Here, the same inpainting technique is

applied to the depth images to populate missing pixels in a 128 by 128 pixels window as it was performed in [42].

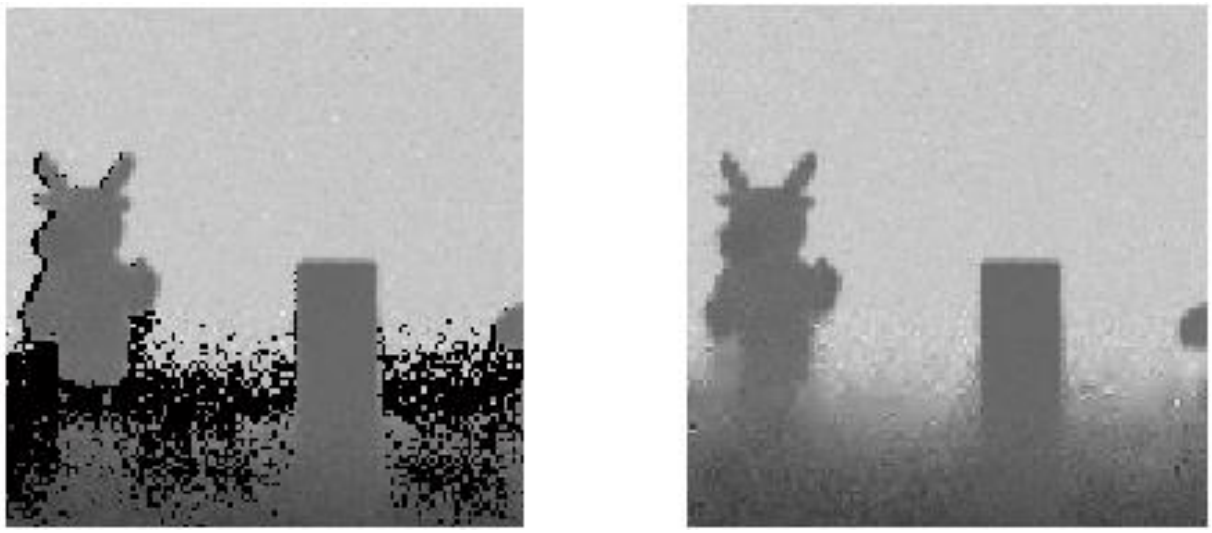


Figure 26: Depth image correction by PFSR. (a) Original depth image with black missing pixels (b) Restored depth image.

An initial research to upscale ToF images was evaluated using this image restoration and inpainting techniques over a sparse depth image with twice the dimensions for a single frame, however the amount of missing information for this case was in the order of 75%, which made this technique not really useful for the upscaling process, but it is a viable alternative for inpainting process of ToF image missing pixels as a preprocessing stage with proper cautions about the creation of artifacts for critical applications.

On the other hand, a different framework that combines both: inpainting and image up-scaling was used for this investigation. The proposed technique is based in the outlier detection method and super-resolution scheme developed by Silva, *et al*. in [35]. This approach is mainly applied to natural images captured with RGB imager sensors and to ToF depth images without displacements. The outlier detection method is preserved as it was originally developed in [35],

however the Total Variation (TV) based regularization scheme is replaced by the Total Generalized Variation functional as well as the numerical method is changed from using Split-Bregman technique to the Alternating Direction Method of Multipliers (ADMM) which will be described in the next section. Additionally, the multi-frame registration technique is reused from [36], except that the registration initialization is assumed totally unknown as described in the Algorithm 1 for synthetic test cases. It is important to establish the close relation between the level of noise, the missing or flickering pixels and the amount of motion between frames and the difficulties to reach a proper registration. That is, more clean images with reduced number of missing pixels will produce better registration results as it was shown in Section 3.4 Multi-frame Registration for Depth images. All these factors make critical the noise removal to boost the registration stage and the up-scaling interpolation process.

The depth image model noise employed for this investigation regards a mixture of Gaussian noise and impulsive noise for the synthetic test cases. The impulsive noise models two different types of phenomena present in the ToF camera caused by the source of errors described in Section 1.3 Time-of-Flight Cameras while the Gaussian noise models the typical variation of the system performance and their electronic components due to manufacturing tolerances.

On the one hand, Random Noise (RV) type of impulsive noise addresses the erroneous depth measurements caused by problems in reflectivity, concave surfaces issues and in general low IR reflected signals that causes wrong phase estimation and at the end an inexact depth estimation. On the other hand, the Salt-and-Pepper (SP) impulsive noise mimics the type of systems that are configured to assign specific control values to the low confidence measurements, such that the Softkinetic camera employed to acquire the depth map shown in Figure 5. In other words, some camera drivers are already configured to set values outside of the valid depth range

for measurements whose IR amplitude is below certain value or when the amount of reflected signal saturates the IR sensor as it was previously explained. Usually in either case, a common practice is to assign zeros to these spots, regarding an equivalent range of gray levels of 16 bits.

The next two following sections describe the theoretical foundation of the noise detection scheme and provide an easy example to illustrate the method.

4.2.1 Theoretical derivation of the noise detection scheme

The depth image restoration problem is actually addressed as an inpainting problem in which there is a missing part Ω_1 of the image domain Ω. This problem is formulated for the discrete case by using a binary matrix Λ:

$$\frac{\lambda}{2}\|\Lambda HX - \Lambda Y\|_2^2 + TTGV_\alpha^2(X) \tag{27}$$

Thus, this is just a modification of the fidelity term of the original functional defined in Equation (24) in which the matrix Λ is acting like a mask with the following form:

$$\Lambda = \begin{bmatrix} \Lambda_1 & & & \\ & \Lambda_2 & & \\ & & \ddots & \\ & & & \Lambda_L \end{bmatrix}$$

Note that in this section the index k is replacing the previous designator used for the low-resolution observations i. This is to avoid confusion with the internal element indexes i and j. Every diagonal component Λ_k corresponds to the lexicographical order of the low-resolution images Y_k.

Thus, regard each Λ_k component as follows:

$$\Lambda_k = \begin{bmatrix} \Lambda_{k_{1,1}} & 0 & \cdots & 0 \\ 0 & \Lambda_{k_{1,2}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Lambda_{k_{rn,rn}} \end{bmatrix}, \quad k = 1, \cdots, L$$

Moreover, it is necessary to consider a vectoral representation of the low-resolution images for the next definition, in which for a given position (i,j) in a Y_k image corresponds the vectoral position $(i-1)n+j$, where n is the number of columns of the image matrix. This is, an assignment to the element $\Lambda_k(i,j)$ is actually affecting the $(i-1)n+j$ diagonal element of the matrix Λ_k.

$$\Lambda_{k_{i,j}} = \begin{cases} 0 & (i,j) \in I_k \\ 1 & (i,j) \in \Omega \backslash I_k \end{cases} \tag{28}$$

Where I_k is the set of corrupted pixels. This last definition stablishes that the binary matrices Λ_k contains the information about corrupted areas with zeros in the image locations where pixels are corrupted with impulsive noise.

From the expression (27) the following matrices can be derived:

$$\Lambda Y = \begin{bmatrix} \Lambda_1 Y_1 \\ \Lambda_2 Y_2 \\ \vdots \\ \Lambda_L Y_L \end{bmatrix}; \quad \Lambda H = \begin{bmatrix} \Lambda_1 H_1 \\ \Lambda_2 H_2 \\ \vdots \\ \Lambda_L H_L \end{bmatrix}$$

The next part of the formulation to be defined are the masking matrices Λ_k. These matrices need to be determined so that they map the location of the pixels contaminated with the type of noise described previously, which in general can be modeled as impulsive noise, this is more specifically Salt and Pepper (S&P) noise for the type of flickering pixels and random noise for

reflectance issues or in general low confidence from the IR amplitude. The Gaussian noise will be removed by the regularization itself, like in the original application of denoising presented in [40]. In this regard, the following functional is defined departing from Equation (27) plus an additional term with the purpose of estimating the number of corrupted pixels with impulsive noise in the low-resolution images Y^l via the l_0-quasi-norm that counts the number of non-zero elements of a vector:

$$F(X,\Lambda) \equiv \frac{\lambda}{2}\|\Lambda HX - \Lambda Y\|_2^2 + TTGV_\alpha^2(X) + \sum_l \eta_l \|\mathrm{I} - \Lambda_l\|_0$$

Where: I is the identity matrix and η_l is threshold to decide if a pixel is corrupted or not, which will be described in more detail in this section. Then this new functional need to be minimized with respect to X and Λ as follows:

$$\min_{X,\Lambda} F(X,\Lambda) \equiv \frac{\lambda}{2}\|\Lambda HX - \Lambda Y\|_2^2 + TTGV_\alpha^2(X) + \sum_l \eta_l \|I - \Lambda_l\|_0$$

But this problem can be separated into two different branches regarding each variable of the minimization, this is one problem for the X minimization and another for the Λ minimization as follows:

$$X^k = \underset{X}{\operatorname{argmin}}\, F(X,\Lambda^k) \equiv \frac{\lambda}{2}\|\Lambda^k HX - \Lambda^k Y\|_2^2 + TTGV_\alpha^2(X) \tag{29}$$

$$\Lambda^k = \underset{\Lambda}{\operatorname{argmin}}\, F(X^k,\Lambda) \equiv \frac{\lambda}{2}\|\Lambda HX^k - \Lambda Y\|_2^2 + \sum_l \eta_l \|I - \Lambda_l\|_0 \tag{30}$$

On the one hand, the problem stated in Equation (29) is practically the same as the one already defined in (24) but with the slight modification in the fidelity term. This problem will be solved with the ADMM technique as it was already established. On the other hand, the problem stated in Equation (30) has fortunately a closed form solution. This second problem can be solved in each iteration with the last computed X^k that minimizes (29) and the X^{k+1} will use the matrix Λ^k just updated. However, the following formulations will assume a fixed X for a given computation of Λ and for simplicity in the notation, the supra index k is eliminated:

$$\underset{\Lambda}{\operatorname{argmin}} \sum_{l} \sum_{i,j} \Lambda_{l_{i,j}} \left((H_l X)_{i,j} - Y_{l_{i,j}} \right)^2 + \sum_{l} \eta_l \sum_{i,j} \left(1 - \Lambda_{l_{i,j}} \right) \tag{31}$$

Recall that index $l = 1, \cdots L$ refers to the low-resolution observations. If it is assumed that the total number of pixels corrupted by impulsive noise is bounded by a constant Γ^l, then the previous problem can be posed as a problem with the restrictions as follows:

$$\begin{aligned} &\underset{\Lambda}{\operatorname{argmin}} \sum_{l} \sum_{i,j} \Lambda_{l_{i,j}} \left((H_l X)_{i,j} - Y_{l_{i,j}} \right)^2 \\ &subject\ to \sum_{i,j} \left(1 - \Lambda_{l_{i,j}} \right) \leq \Gamma^l, l = 1, \cdots, L. \end{aligned} \tag{32}$$

Note that the bound Γ^l is part of the *a priori* knowledge based on the experience with the type of images that are processed. Equation (32) is separable in that there is a total of L minimization arguments for L corresponding constraints, which leads to L problems of the type:

$$\underset{\Lambda}{\operatorname{argmin}} \sum_{i,j} \Lambda_{l_{i,j}} \left((H_l X)_{i,j} - Y_{l_{i,j}} \right)^2$$

$$subject\ to \sum_{i,j} \left(1 - \Lambda_{l_{i,j}} \right) \leq \Gamma^l \tag{33}$$

Problems (31), (32) and (33) are equivalents for some η_l in relation with the bounds Γ^l. This is, for each specific η_l there exists a Γ^l such that problems (31), (32) and (33) are equivalent and vice versa.

In order to determine the values of the masking matrix $\Lambda_{l_{i,j}}$, problem (33) is rearranged as:

$$\underset{\Lambda}{\operatorname{argmin}} \sum_{i,j} \Lambda_{l_{i,j}} \left((H_l X)_{i,j} - Y_{l_{i,j}} \right)^2$$

$$subject\ to \sum_{i,j} 1 \leq \Gamma^l + \sum_{i,j} \Lambda_{l_{i,j}}$$

Subsequently:

$$\underset{\Lambda}{\operatorname{argmin}} \sum_{i,j} \Lambda_{l_{i,j}} \left((H_l X)_{i,j} - Y_{l_{i,j}} \right)^2$$

$$subject\ to\ n^2 - \Gamma^l \leq \sum_{i,j} \Lambda_{l_{i,j}} \tag{34}$$

Where n^2 is the total number of pixels of the low-resolution observations as defined in the previous sections. Problem (34) imposes the constraint that there is a minimum $(n^2 - \Gamma^l)$ of $\Lambda_{l_{i,j}}$

equal to 1s, which denote non-corrupted pixel locations in the binary matrix mask Λ_l. Recall that according to Equation (28) the zeros in the binary matrix Λ_l are assigned to impulsive noise corrupted pixel locations.

In other words, the minimization of the objective function in problem (34) implies that the $\Lambda_{l_{i,j}}$ components will be equal to 1 for the $(n^2 - \Gamma^l)$ smallest $\left\{\left((H^l X)_{i,j} - Y_{i,j}^l\right)^2\right\}_{i,j}$ elements or equivalent the $\Lambda_{l_{i,j}}$components will be equal to 0 for the biggest $\left\{\left((H^l X)_{i,j} - Y_{i,j}^l\right)^2\right\}_{i,j}$ elements.

Finally, an expression for the $\Lambda_{l_{i,j}}$ is defined as:

$$\Lambda_{l_{i,j}} = \begin{cases} 0 & \left((H_l X)_{i,j} - Y_{l_{i,j}}\right)^2 > \tau_l \\ 1 & \left((H_l X)_{i,j} - Y_{l_{i,j}}\right)^2 \leq \tau_l \end{cases}$$

where: τ_l is the Γ^l largest element of $\left\{\left((H_l X)_{i,j} - Y_{l_{i,j}}\right)^2\right\}_{i,j}$.

In this way, now it is possible to introduced the Algorithm 2.

Algorithm 2 Registration Upscaling Method with impulsive noise detection

Require: LR images

1. Set initial conditions: sx = 0; sy=0; rot=0;
2. Compute estimated high-res. image $\hat{X} = upscaleBicubic(Y_1)$;
3. $Y_{CL} = ACMF(Y)$;
4. [sx, sy, rot] = $F(\hat{X}, Y, sx, sy, rot)$;
5. Build H ← D · B · S;
6. k = 1;
7. Compute an estimative of inpainting matrix Λ_0;
8. H ← Λ_0H;
9. Y ← Λ_0Y;
10. $X_{init} = \hat{X}$;
11. $\Lambda = \Lambda_0$;
12. **while** $k \leq$ numIter **do**
    a. $T \leftarrow Compute\_Tensor(Y_{CL}, sx, sy, rot)$
    b. $X \leftarrow \underset{X}{\operatorname{argmin}}\, F(X, \Lambda, X_{init}, H, Y, T)$ according to Algorithm **3**;
    c. [sx, sy, rot] = $F(X, Y, sx, sy, rot)$;
    d. Build H ← D · B · S;
    e. For $l \leq$ numLRimgs do
        i. $\tau_l \leftarrow L^l$ largest $\left((H_l X)_{i,j} - Y_{l_{i,j}}\right)$
        ii. $\Lambda \leftarrow \left[\left((H_l X)_{i,j} - Y_{l_{i,j}}\right)^2 < \tau_l\right]$
    f. end **For**;
    g. H ← ΛH;
    h. Y ← ΛY;
    i. $X_{init} \leftarrow X$;
    j. $k \leftarrow k + 1$;
13. end of **while**;
14. return X, Λ

A numeric example is presented in the next section to sketch a hypothetical image case to describe the employed noise detection scheme.

4.2.2 Noise Detection Dummy Example

Consider the following 3×3 matrices corresponding to the terms $H^l X$ and $Y_{i,j}^l$ of a given observation l, in which the pixel locations contain hypothetical normalized values in a range [0,1] as the reused implemented code does.

$$H^l X = \begin{bmatrix} 0.2 & 0.2 & 0.3 \\ 0.2 & 0.2 & 0.3 \\ 0.4 & 0.4 & 0.4 \end{bmatrix}; \qquad Y^l = \begin{bmatrix} 0.2 & 0.2 & 0.5 \\ 0.2 & 0.2 & 0 \\ 0.2 & 0.4 & 0.1 \end{bmatrix}$$

Note that the elements which are different in both matrices are in red, where $H^l X^k$ is simulating a clean version of the observation Y^l. Then the quantification of how far some of the pixels are from the clean case is computed with the difference of these two matrices is:

$$H^l X - Y^l = \begin{bmatrix} 0 & 0 & -0.2 \\ 0 & 0 & 0.3 \\ 0.2 & 0 & 0.3 \end{bmatrix}$$

For this given example, assume an *a priori* knowledge that indicates that the Target images have 40% of the pixels with noise. Then the 40% of pixels in this small matrix size of a total of 9 pixels is: $9 \times \left(\frac{40\%}{100\%}\right) = 3.6$. However, consider only the integer part since indexes are not fractional. This means that the threshold value τ_l will be extracted around the 3 first positions of the descendent ordered version of the absolute value of the difference $(H^l X - Y^l)$.

Figure 27 shows the vector format version of the absolute value of the elements of $H^l X - Y^l$, its position indexes and the descent ordered version of it as well as the localization of the threshold value to build the inpainting matrix.

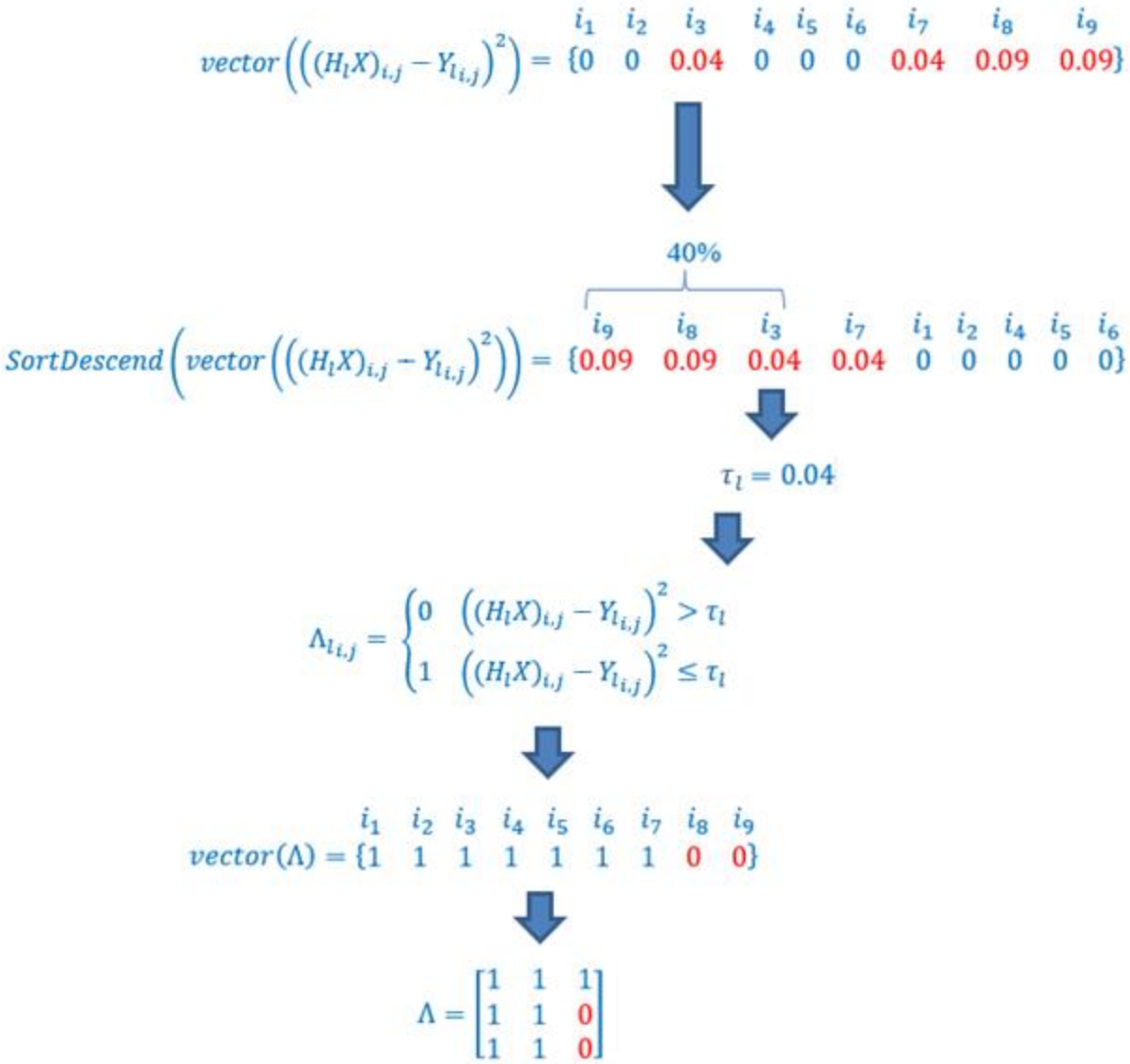


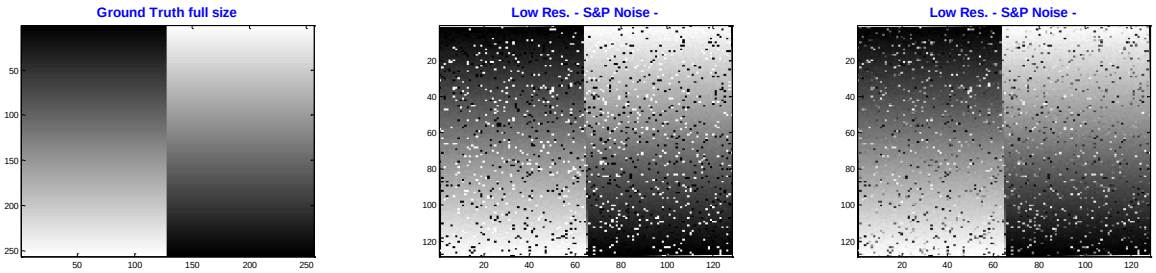
Figure 27: Impulsive Noise Detection Numeric Example.

It is important to mention that the inpainting matrix will be evolving across the iteration as shown in Algorithm 2. Thus, some pixels could be oscillating but at the end the matrix will convergence to a noise free image since it will be computed and operating updated matrices improved in every iteration respect to the noise observations.

4.3 Regularization Terms Characterization.

This section presents the performance of the two regularization terms that were implemented for this work, the Total Variation (TV) and the Total Generalized Variation (TGV). It is well known that the first order TV regularization term introduces artefacts of the staircasing type, while the second order TGV [39] eliminates those artefacts.

A set of 10 synthetic images with simple geometric shapes, gradual changes in pixel intensity, contaminated with 10% of salt and pepper noise as well as 3dB-sigma Gaussian noise were evaluated and studied to show the evident elimination of the stair-casing artefacts from the reconstructed image. Furthermore, a second set of 10 images were contaminated with random value impulsive noise and they were evaluated with the two regularization parameters. Figure 28 shows the original image and representative low-resolution frames for both types of noise.



(a) (b) (c)

Figure 28: Double_slope image: (a) Ground truth Image 256x256 pixels; (b) Salt & Pepper Low-resolution frames 128x128; (c) Random noise Low-resolution frames 128x128.

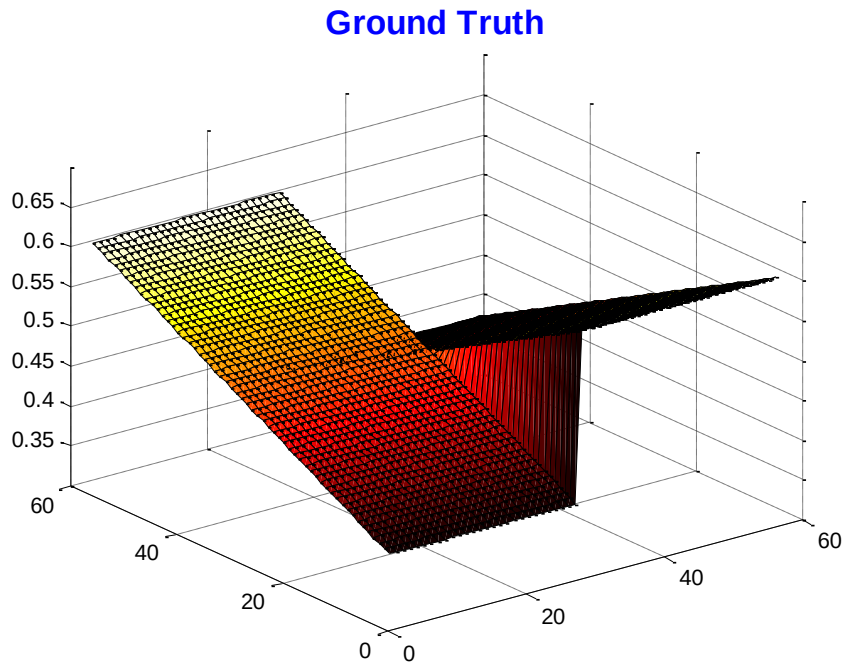


Figure 29: Ground Truth surface close up plot of the central area of the Double_slope image.

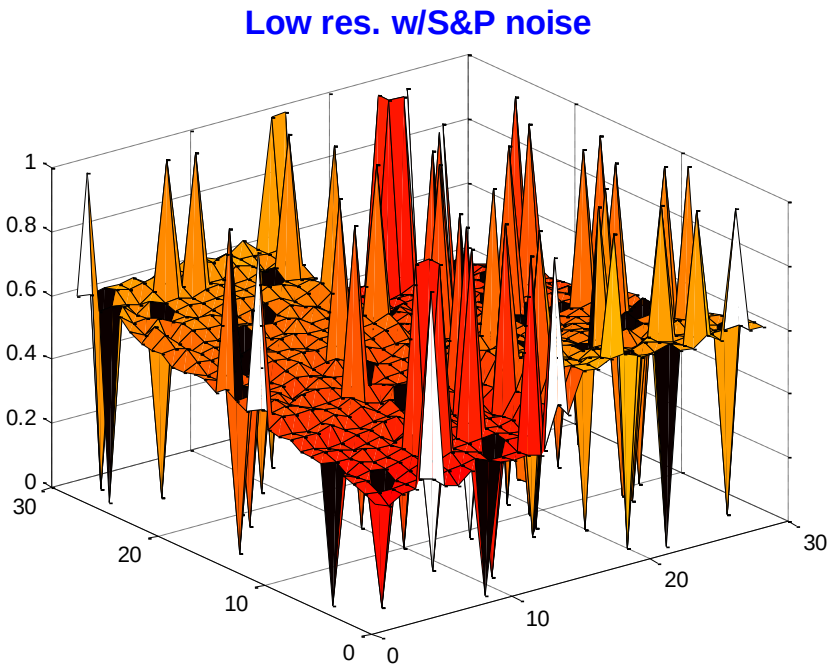


Figure 30: Low Resolution close up plot of the central area of the Double_slope image contaminated with S&P noise.

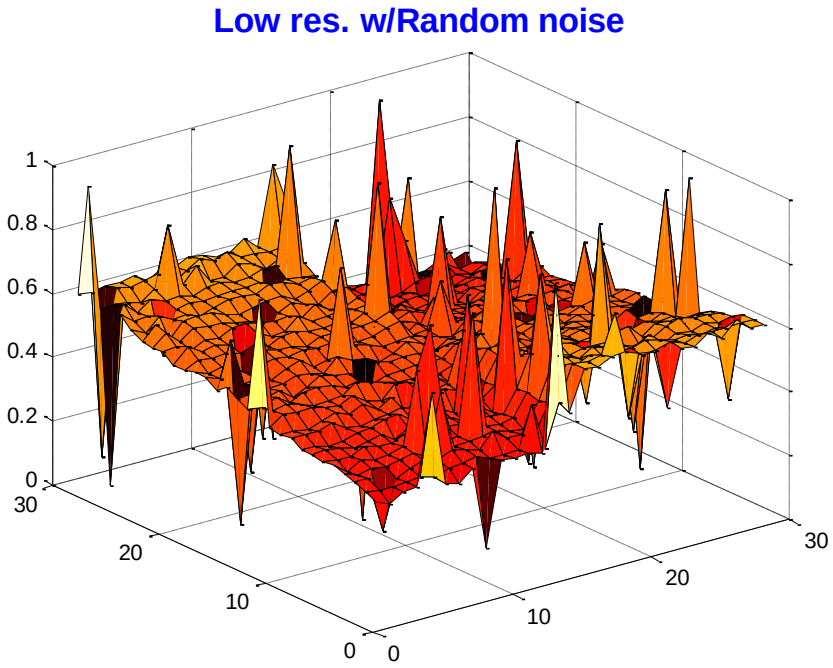


Figure 31: Low Resolution close up plot of the central area of the Double_slope image contaminated with random noise.

Table 9 contain the objective evaluation of the up-scaled and reconstructed images for the two cases of noise contamination. Note that the figures for PSNR and SSIM do not behave in the same way.

Table 9: Double_slope synthetic image numeric results for the S&P noise case.

Method	S&P Noise				Random Value Noise			
	Iter	**PSNR**	**Iter**	**SSIM**	**Iter**	**PSNR**	**Iter**	**SSIM**
ADMM-TV	500	29.364	8	0.9795	456	29.133	476	0.9779
ADMM-TGV	482	29.673	485	0.9828	206	28.615	226	0.9847
ADMM-TTGV	286	26.946	500	0.9848	488	26.780	475	0.9858

The Figure 32 and Figure 33 plot a selected row and a selected column of the reconstructed images using the different evaluated functionals for the two cases of noise contamination to compare against the Ground Truth image the same location, which provide an initial good idea of the staircasing performance across the two spatial dimensions of the reconstructed image.

Additionally, Figure 34 through Figure 36 present a close up of the central part of the reconstructed imaged in pseudo-color representation to provide another subjective evaluation of the staircase performance.

Finally, Figure 37 through Figure 42 generate surfaces in 3D of the reconstructed images, which are actually depth maps.

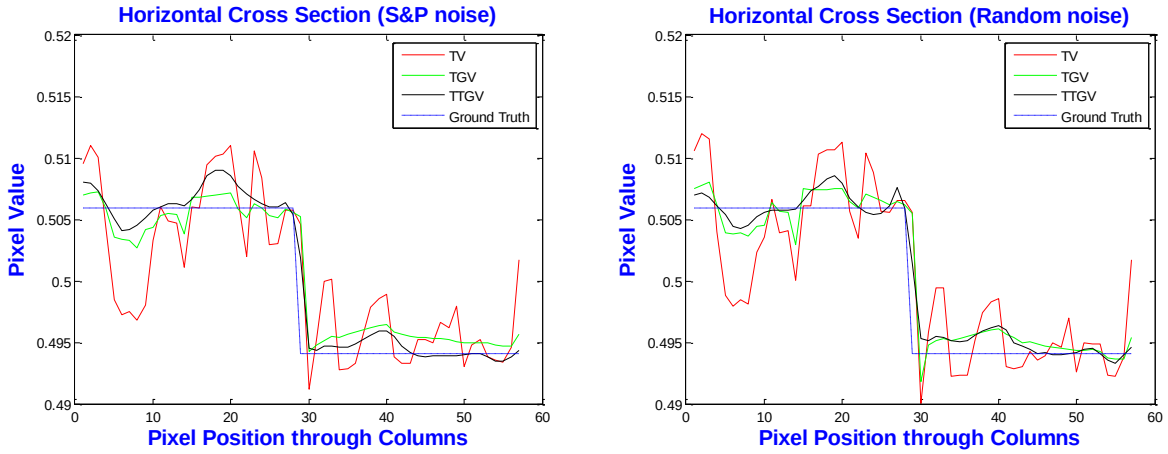


Figure 32: Comparison of regularization terms in a Horizontal cross-section of the image Double_slope.

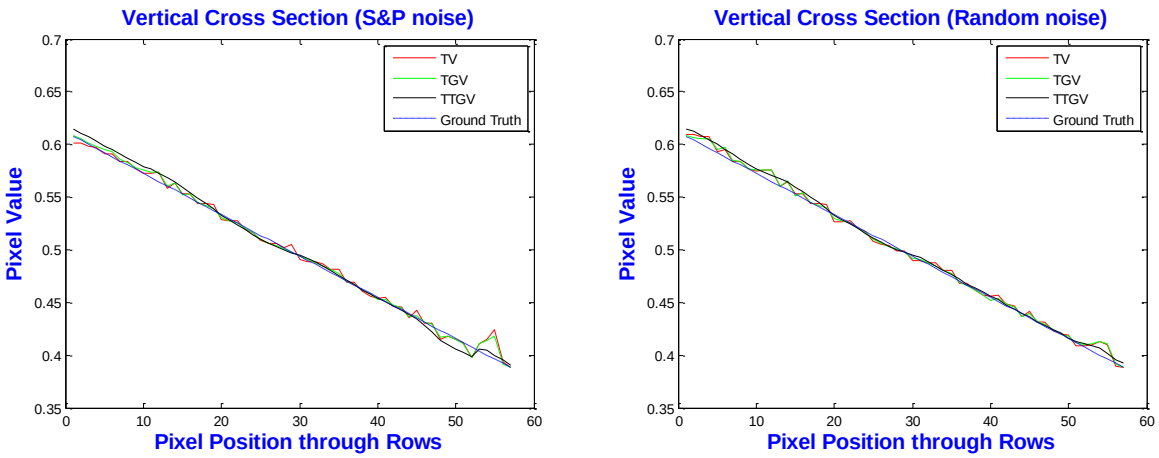


Figure 33: Comparison of regularization terms in a Vertical cross-section of the image Double_slope.

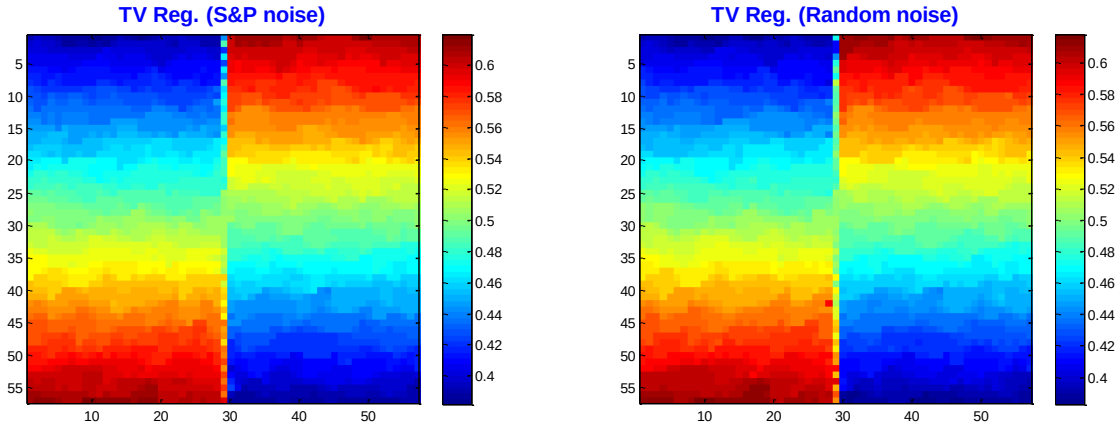


Figure 34: TV Regularization result of image Double_slope.

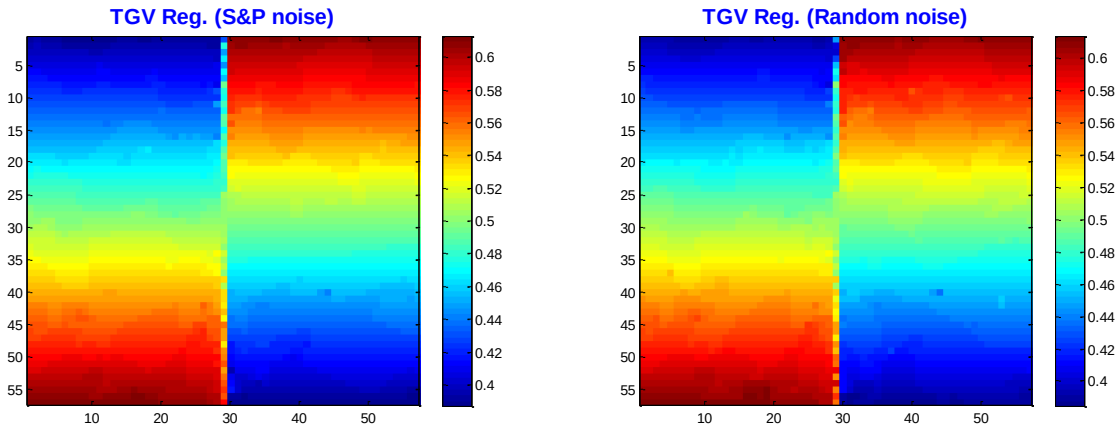


Figure 35: TGV Regularization result of image Double_slope.

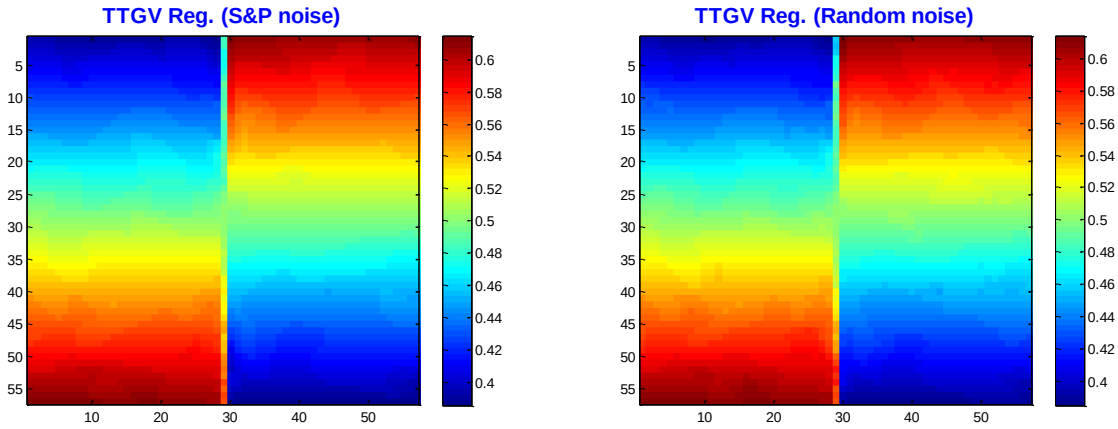


Figure 36: Tensor-TGV Regularization result of image Double_slope.

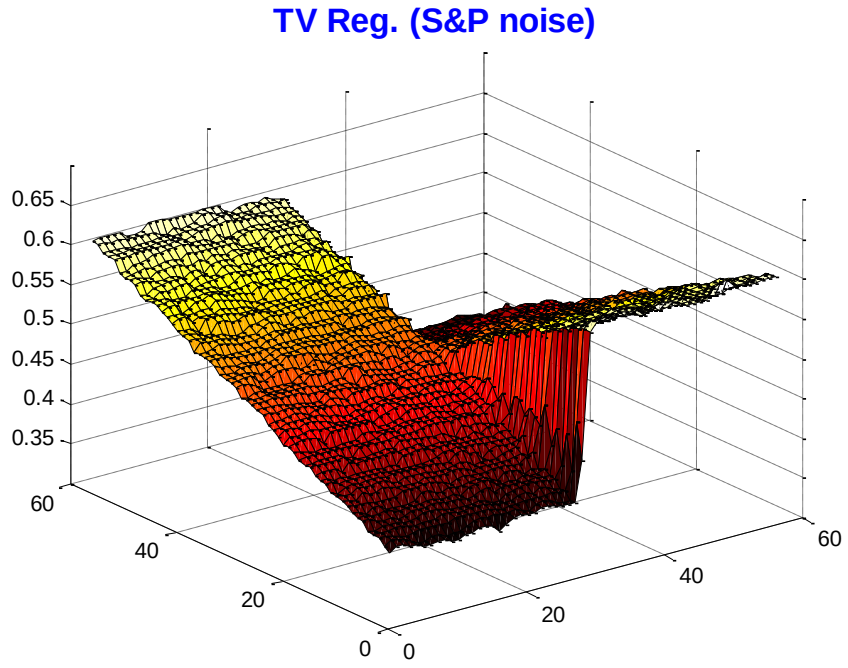


Figure 37: TV Regularization Result Surface of image Double_slope previously contaminated with S&P noise.

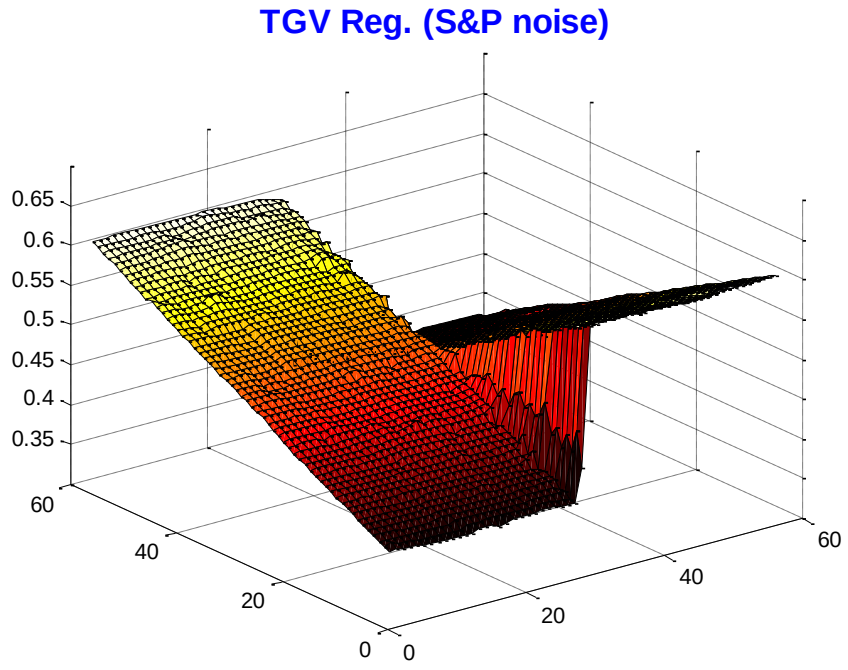


Figure 38: TGV Regularization Result Surface of image Double_slope contaminated with S&P noise.

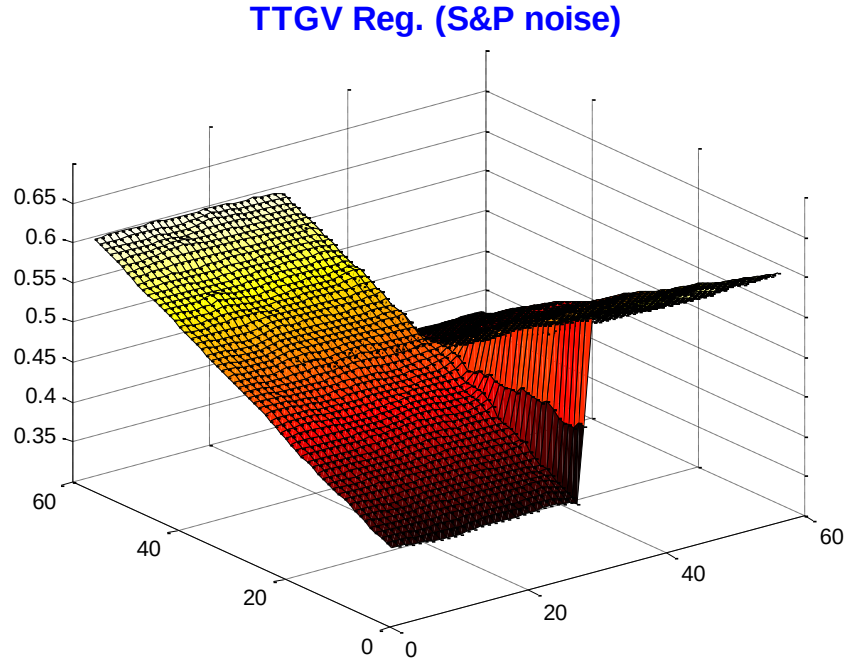


Figure 39: Tensor TGV Regularization Result Surface of image Double_slope contaminated with S&P noise.

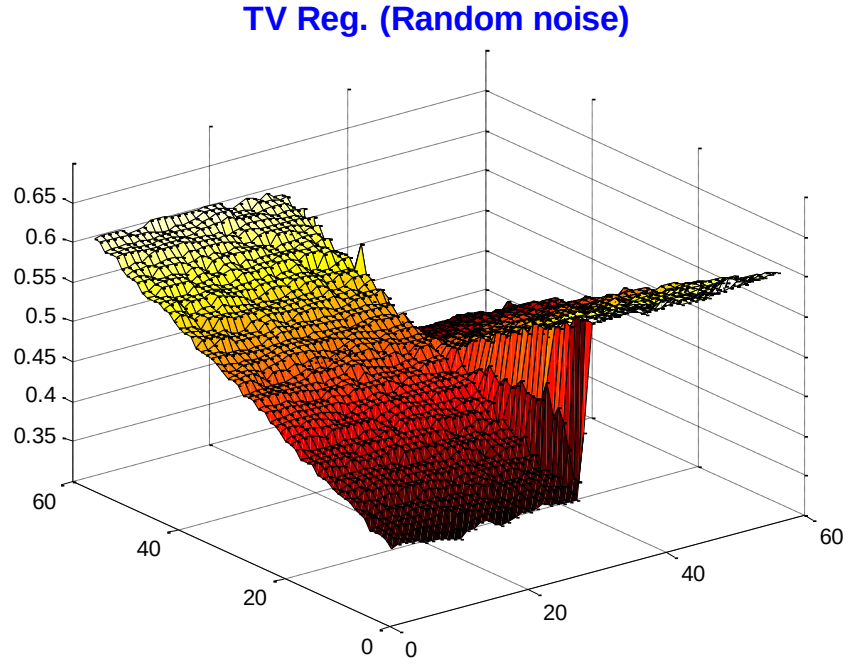


Figure 40: TV Regularization Result Surface of image Double_slope previously contaminated with Random noise.

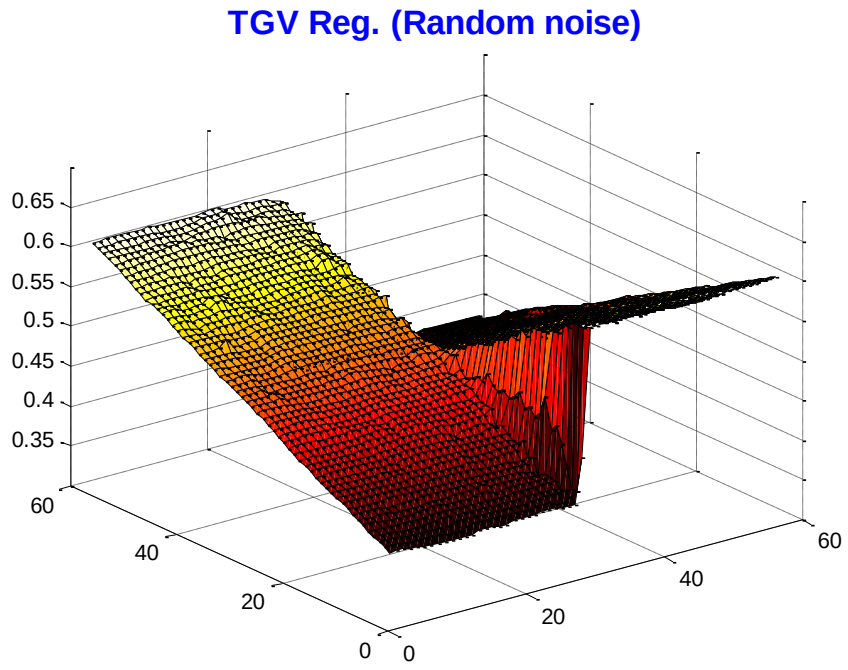


Figure 41: TGV Regularization Result Surface of image Double_slope previously contaminated with Random noise.

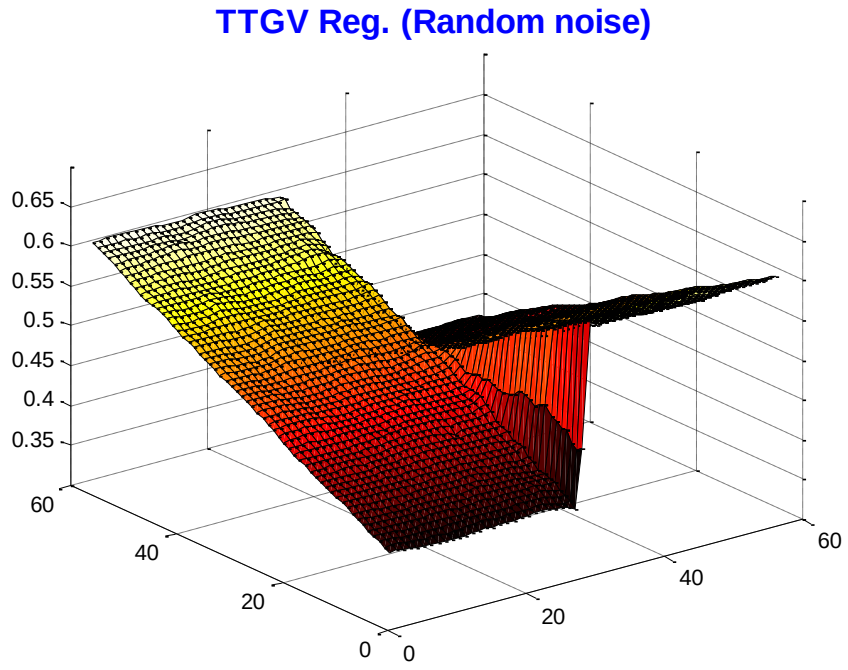
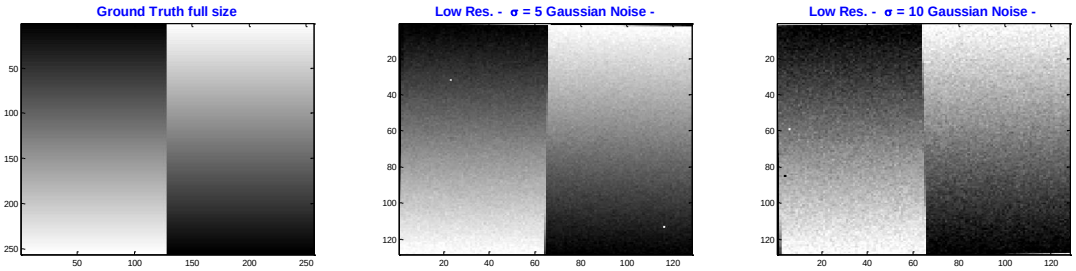


Figure 42: TTGV Regularization Result Surface of image Double_slope previously contaminated with Random noise.

In addition to the impulsive noise characterization, two cases of pure Gaussian noise contamination were tested to show the performance of the regularization terms. The Gaussian noise standard deviations for each case were $\sigma_1 = 5$ and $\sigma_2 = 10$. The impulsive noise was reduced to less than 0.01 %, a negligible amount.



(a) (b) (c)

Figure 43: Double_slope image: (a) Ground Truth Image 256x256 pixels; (b) Gaussian Noise $\boldsymbol{\sigma_1 = 5}$ Low-resolution frames 128x128; (c) Gaussian Noise $\boldsymbol{\sigma_2 = 10}$ Low-resolution frames 128x128.

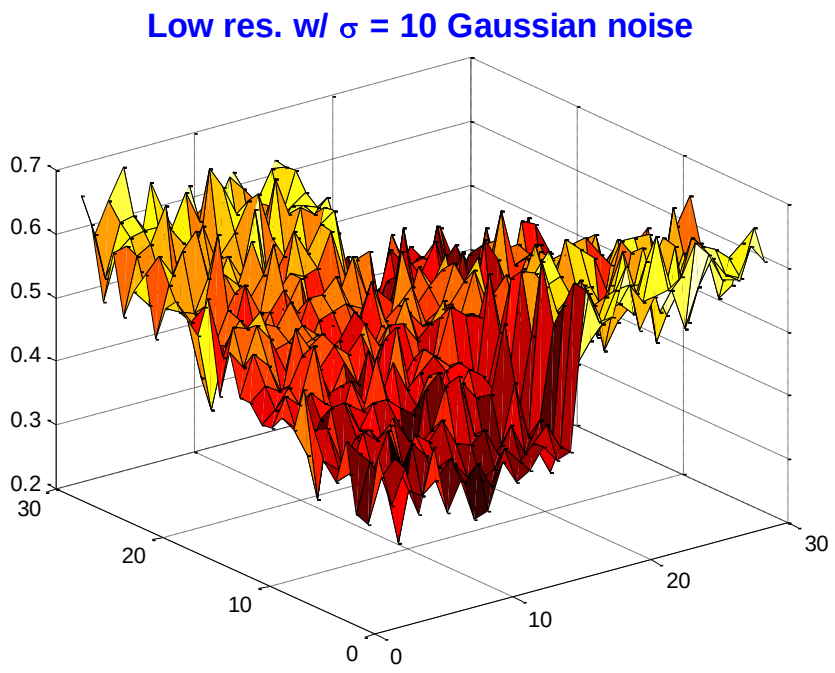


Figure 44: Low Resolution close up plot of the central area of the Double_slope image contaminated with Gaussian Noise $\boldsymbol{\sigma_1 = 5.}$

The subsequence Figure 45 - Figure 51 present the regularization results for these two cases of Gaussian noise.

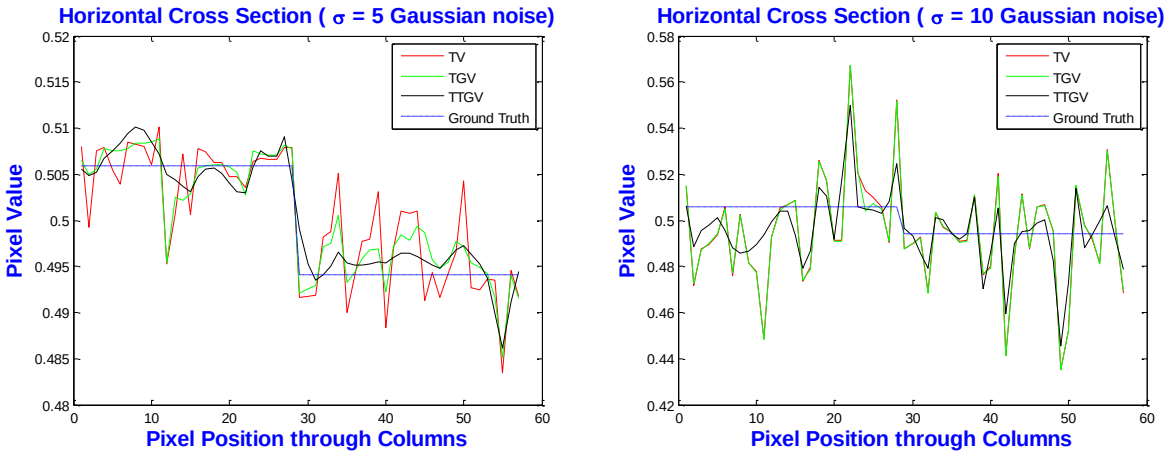


Figure 45: Comparison of regularization terms in a Horizontal cross-section of the image Double_slope contaminated with Gaussian Noise.

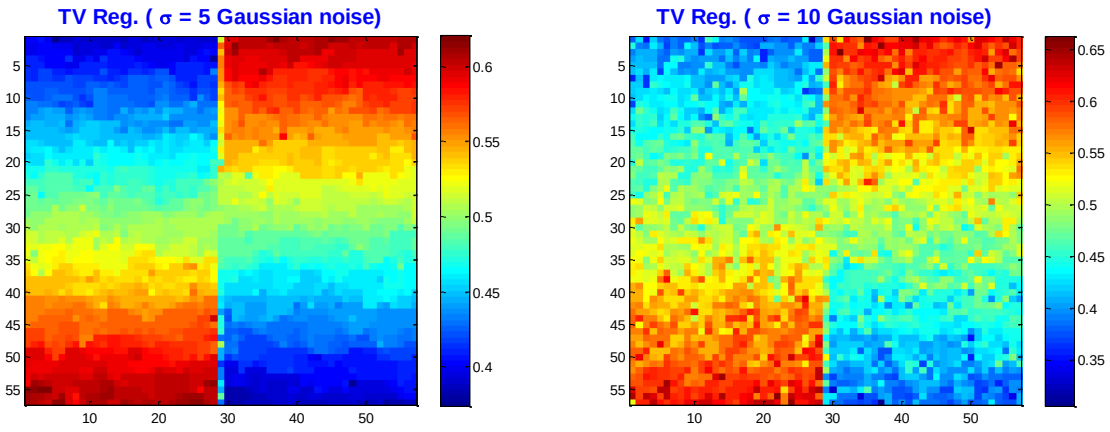


Figure 46: TV Regularization result of image Double_slope contaminated with Gaussian Noise.

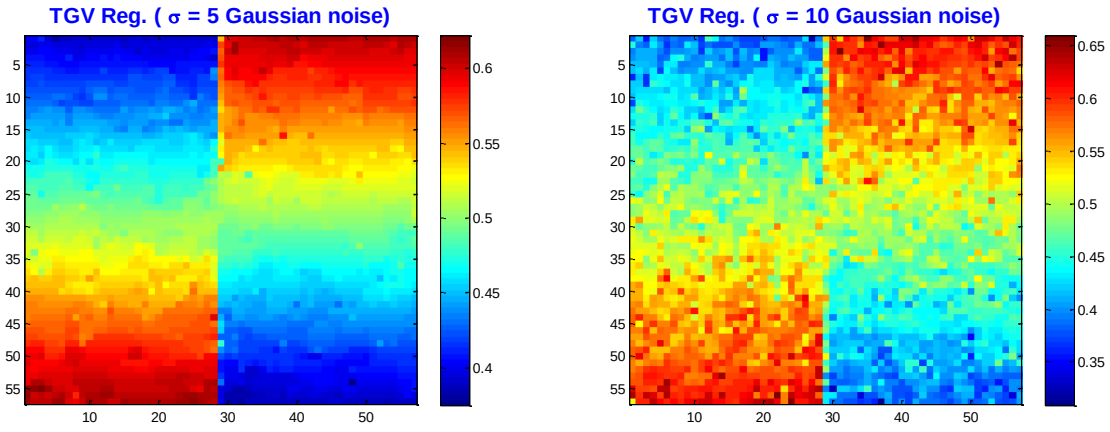


Figure 47: TGV Regularization result of image Double_slope contaminated with Gaussian Noise.

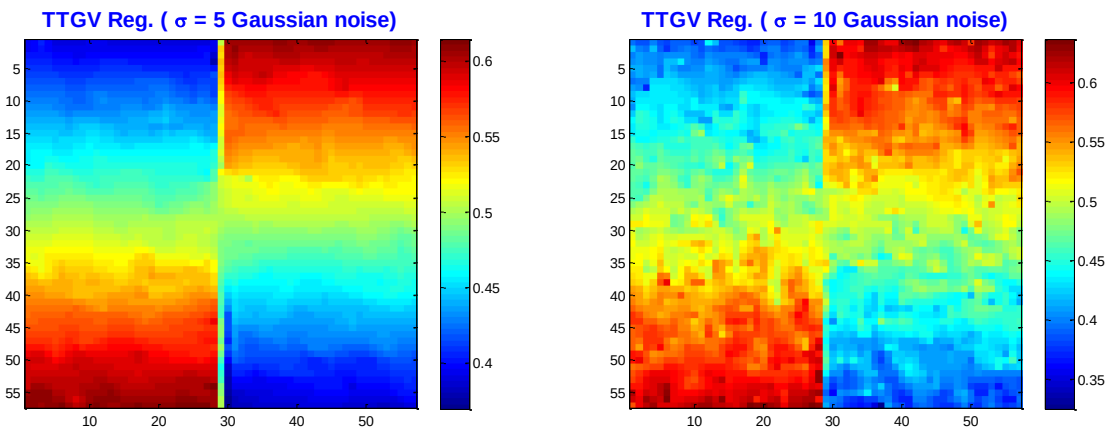


Figure 48: TTGV Regularization result of image Double_slope contaminated with Gaussian Noise.

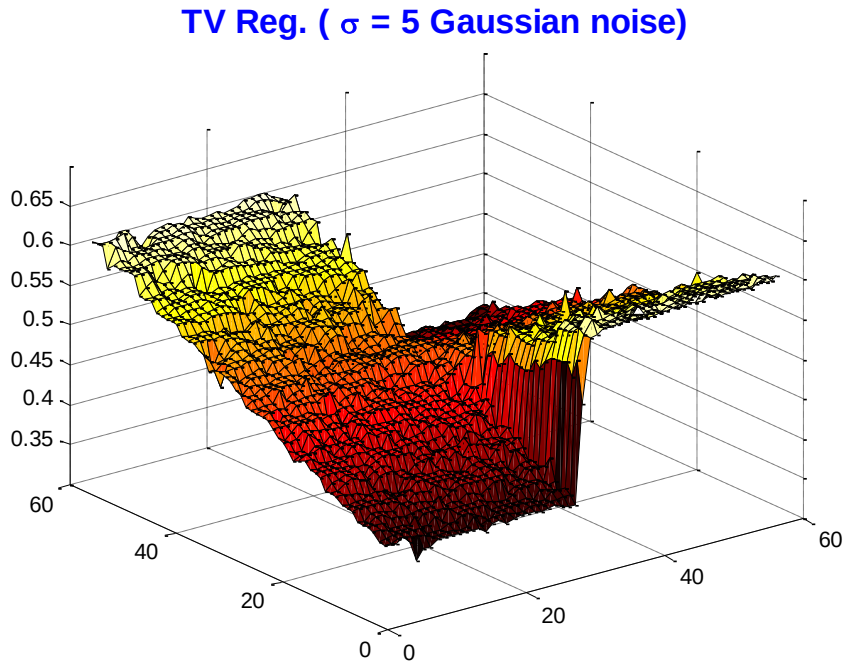


Figure 49: TV Regularization Result Surface of image Double_slope previously contaminated with Gaussian Noise.

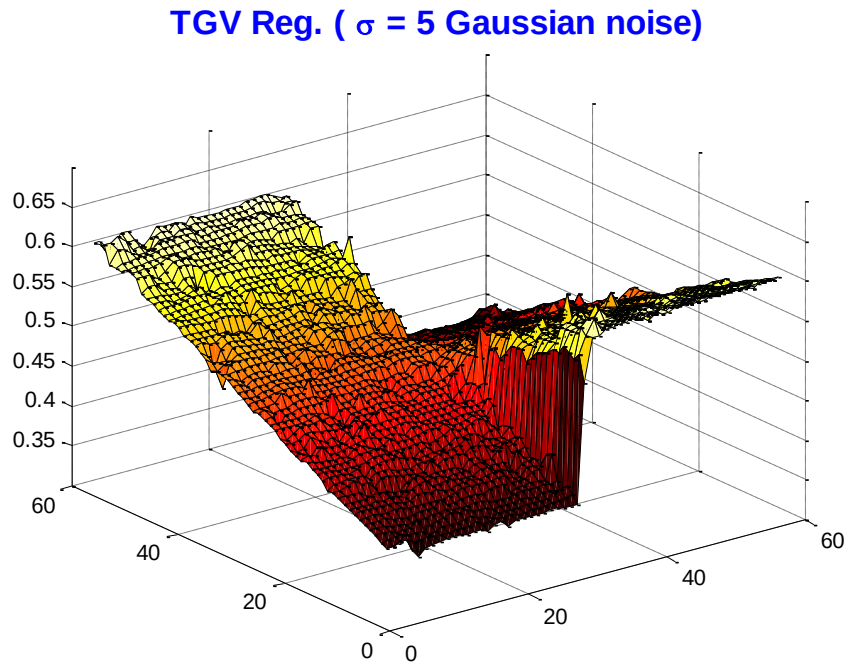


Figure 50: TGV Regularization Result Surface of image Double_slope previously contaminated with Gaussian Noise.

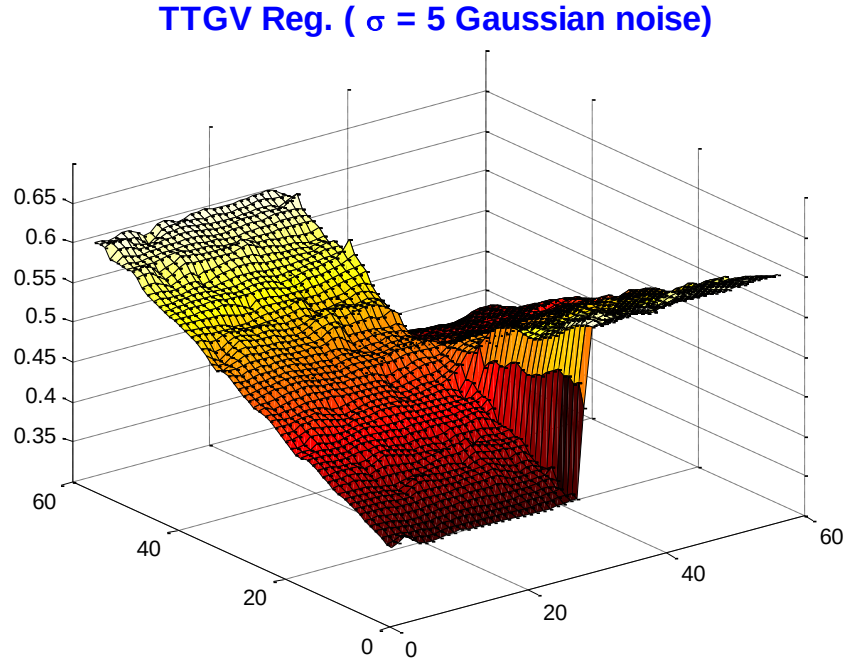


Figure 51: TTGV Regularization Result Surface of image Double_slope previously contaminated with Gaussian Noise.

Chapter 5: Numerical Method Implementation for Depth Super-Resolution

This chapter presents the implemented numerical method to solve the super-resolution problem of depth maps. The first part presents a brief introduction to the method and the motivation to use it. The following section shows the modification from a previous implementation for a different application and proceeds to introduced an additional term to improve the process via a tensor.

5.1 Numerical Optimization based on Alternating Direction Method of Multipliers

The optimization solver technique selected to perform the depth image up-sampling was based on the Alternating Direction Method of Multipliers (ADMM), which is a distributed optimization method that can be used to implement efficient and scalable algorithms [43]. The structure of the ADMM in terms of decomposing the main problem into several subproblems and its flexibility make this method very suitable for applications of decentralized algorithms and provide opportunities for parallel optimization as well [44].

A brief description of the method history and a survey of the technique are provided in [44]. Some similar ideas to the ADMM are traced back to 1955 with the work of J. Douglas and H. Rachford in a procedure that can be factored into a multi-stage process to find the solution of the linear equations in an easier way [45]. Then by 1976, the ADMM was introduced by D. Gabay and B. Mercier with their approach based on the use of an Augmented Lagrangian functional that leads to an efficient and simply implementable algorithm [46]. In general, the ADMM is easy to

implement as well as it doesn't require as many iterations as other methods. This was shown in the case of the Primal Dual method used in Section 2.3 Image guided Depth Up-sampling using TGV.

As was shown previously in Algorithm 2, every impulsive noise removal iteration needs to compute a new X, which is updated using an optimization method that employs the ADMM technique. The following section presents the optimization solver implementation of the proposed algorithm, which is a modification from the work of Chuan He in [40] that uses ADMM to solve a functional with a TGV term for denoising applications of a single image. On the other hand, this work proposes to apply the ADMM with the TGV functional term to the image up-scaling problem, also introduces and additional term with a tensor to improve the registration process and the final reconstructed image quality.

5.2 ADMM-TGV Modifications and Improvements

The first step to implement the intended ADMM-TGV modification for multi-frame super-resolution and the incorporation of a tensor requires to visit first the original constrained problem defined by [40]. This initial problem uses auxiliary variables and pretends to avoid the problems to solve the TGV functional defined in Equation (22) which suffers from the absence of linearity and differentiability. Thus, the TGV problem can be posed as an equivalent constrained problem:

$$\underset{\mathbf{u},\mathbf{p}}{\operatorname{argmin}}\{\alpha_0\|\mathbf{y}\|_1 + \alpha_1\|\mathbf{z}\|_1 + \mathrm{I}_{\Phi}(\mathbf{w})\} \tag{35}$$

$$\text{subject to } \mathbf{u} = \mathbf{w},\ \varepsilon(\mathbf{p}) = \mathrm{y},\ \ \nabla\mathbf{u} - \mathbf{p} = \mathbf{z}.$$

where $\mathrm{I}_{\Phi}(\mathbf{u})$ is an indication function of the feasibility set defined as:

$$\mathrm{I}_{\Phi}(\mathbf{u}) = \begin{cases} 0 & if\ u\ \in \Phi \triangleq \{u: \|u - u_0\|_2^2\ \leq c\} \\ +\infty & otherwise \end{cases}$$

Then [40] defines also the Lagrangian functional as:

$$\mathcal{L}_{\mathcal{A}}(\mathrm{u}, \mathrm{p}, \mathrm{w}, \mathrm{y}, \mathrm{z}; \mu, \xi, \eta)\ \triangleq\ \mathrm{I}_{\Phi}(\mathbf{w}) - \langle \mu, \mathbf{w} - \mathbf{u} \rangle + \frac{\beta_1}{2} \|\mathbf{w} - \mathbf{u}\|_2^2$$

$$+\alpha_0 \|\mathbf{y}\|_1 - \langle \xi, \mathbf{y} - \varepsilon(\mathbf{p}) \rangle + \frac{\beta_2}{2} \|\mathbf{y} - \varepsilon(\mathbf{p})\|_2^2 0$$

$$+\alpha_1 \|\mathbf{z}\|_1 - \langle \eta, \mathbf{z} - \nabla \mathbf{u} + \mathbf{p} \rangle + \frac{\beta_3}{2} \|\mathbf{z} - \nabla \mathbf{u} + \mathbf{p}\|_2^2 \qquad (36)$$

where μ, ξ and η are the Lagrange multipliers and β_1, β_2 and β_3 are penalty parameters.

However, (35) and (36) are equations to solve a denoising problem and the purpose of this investigation is to perform a super-resolution process. Therefore, the super-resolution problem requires a term of the form $\|\mathrm{H}\mathbf{u} - \mathrm{Y}\|_2^2$ instead of the indication function $\mathrm{I}_{\Phi}(\mathbf{u})$. Then, the defined problem before using auxiliary variables is as follows:

$$\underset{\mathbf{u},\mathbf{p}}{\operatorname{argmin}} \left\{ \alpha_0 \|\varepsilon(\mathbf{p})\|_1 + \alpha_1 \|\nabla \mathbf{u} - \mathbf{p}\|_1 + \frac{1}{2} \|\mathrm{H}\mathbf{u} - \mathrm{Y}\|_2^2 \right\}$$

The constrained problem using auxiliary variables to remove the restriction of a feasible set over **u** is:

$$\underset{\mathbf{u},\mathbf{p}}{\operatorname{argmin}} \left\{ \alpha_0 \|\mathrm{y}\|_1 + \alpha_1 \|\mathbf{z}\|_1 + \frac{1}{2} \|\mathrm{H}\mathbf{w} - \mathrm{Y}\|_2^2 \right\} \qquad (37)$$

subject to $\mathbf{u} = \mathbf{w}$, $\varepsilon(\mathbf{p}) = \mathrm{y}$, $\nabla \mathbf{u} - \mathbf{p} = \mathbf{z}$.

The incorporation of an additional structure tensor in (37) is proposed as previously introduced in equation (23) but with the following modified LaGrange functional that introduces an additional auxiliary variable x in the term $\alpha_1\|\mathbf{x}\|_1$ as well as an additional constraint **x** = **Tz** resulting in the new constrained problem:

$$\underset{\mathbf{u},\mathbf{p}}{\operatorname{argmin}}\left\{\alpha_0\|\mathbf{y}\|_1 + \alpha_1\|\mathbf{x}\|_1 + \frac{1}{2}\|\mathrm{H}\mathbf{w} - \mathrm{Y}\|_2^2\right\} \tag{38}$$

$$\text{subject to } \mathbf{u} = \mathbf{w},\ \varepsilon(\mathbf{p}) = \mathrm{y},\ \nabla\mathbf{u} - \mathbf{p} = \mathbf{z},\ \mathbf{x} = \mathbf{T}\mathbf{z}$$

Where T is the proposed structure tensor of a high-resolution image that will help the registration process and the super-resolution algorithm in general. The modified terms and the new ones are highlighted to facilitate the comparison with the original ones defined in [40]. In the same way, the new augmented Lagrangian functional $\mathcal{L}_{\mathcal{A}_{SRT}}$ is now defined as a modified version of (36) in the following form, taking into account the new terms:

$$\mathcal{L}_{\mathcal{A}_{SRT}}(\mathrm{u},\mathrm{p},\mathrm{w},\mathrm{x},\mathrm{y},\mathrm{z};\mu,\xi,\eta,\theta) \triangleq \frac{1}{2}\|\mathrm{H}\mathbf{w} - \mathrm{Y}\|_2^2 - \langle\mu,\mathbf{w}-\mathbf{u}\rangle + \frac{\beta_1}{2}\|\mathbf{w}-\mathbf{u}\|_2^2$$

$$+\alpha_0\|\mathbf{y}\|_1 - \langle\xi,\mathbf{y}-\varepsilon(\mathbf{p})\rangle + \frac{\beta_2}{2}\|\mathbf{y}-\varepsilon(\mathbf{p})\|_2^2$$

$$+\alpha_1\|\mathbf{x}\|_1 - \langle\theta,\mathbf{x}-\mathbf{T}\mathbf{z}\rangle + \frac{\beta_4}{2}\|\mathbf{x}-\mathbf{T}\mathbf{z}\|_2^2$$

$$-\langle\eta,\mathbf{z}-\nabla\mathbf{u}+\mathbf{p}\rangle + \frac{\beta_3}{2}\|\mathbf{z}-\nabla\mathbf{u}+\mathbf{p}\|_2^2 \tag{39}$$

where β_4 is a parameter introduced to penalize the new term $\|\mathbf{x}-\mathbf{T}\mathbf{z}\|_2^2$, and the parameter θ is now required for the relation between **x** and **Tz**.

On the one hand, the optimization subproblem **y** remains as originally defined by [40] since any term that include this variable was affected by the **x, w** or **z**. The same thing happens for the other subproblems **u**, **p** and the iterative updates for the Lagrange multipliers μ, ξ and η, although a new parameter is introduced for that to address the relation between **x** and **Tz**, which is θ. No change is required in their original formulas. Moreover, to avoid excessive indexes the iterative parameters will be assumed to be constants for the subproblem definitions and derivations, however the final algorithm will take in account the required updates.

On the other hand, the **w** subproblem now includes the super-resolution term instead of the denoising one, the **z** subproblem is also different and makes it necessary to introduce another subproblem to solve the auxiliary variable **x**. The following sections will show how these subproblems were modified.

5.2.1 ADMM-TGV for Super-Resolution Subproblem

By grouping and minimizing all terms with **w** from the new augmented Lagrangian functional defined in equation (39), the subproblem **w** can be established. Notice that the super-resolution term is part of the **w** subproblem. The derived expression from (39) to solve this subproblem is defined then by:

Subproblem w:

$$w_{k+1} = \underset{\mathbf{w}}{\operatorname{argmin}}\left\{\mathcal{L}_{\mathcal{A}_{SRT}}\left(\mathrm{u}_{k+1}, \mathrm{p}_{k+1}, \mathrm{w}_k, \mathrm{x}_k, \mathrm{y}_k, \mathrm{z}_k; \mu_k, \xi_k, \eta_{k,}\, \theta_{k,}\right)\right\}$$

For convenience in the notation, the iteration index k is omitted from some of the variables in the following equations.

$$w_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\mathrm{H}\mathbf{w} - \mathrm{Y}\|_2^2 - \langle \mu, \mathbf{w} - \mathbf{u} \rangle + \frac{\beta_1}{2} \|\mathbf{w} - \mathbf{u}\|_2^2 \right\} \tag{40}$$

Consider rearranging Equation (40) in the following way by adding and subtracting $\frac{\beta_1}{2}\left\langle \frac{\mu}{\beta_1}, \frac{\mu}{\beta_1} \right\rangle$ to introduce the internal product together with the $\|\mathbf{w} - \mathbf{u}\|_2^2$.

$$w_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\mathrm{H}\mathbf{w} - \mathrm{Y}\|_2^2 - \langle \mu, \mathbf{w} - \mathbf{u} \rangle + \frac{\beta_1}{2} \|\mathbf{w} - \mathbf{u}\|_2^2 + \frac{\beta_1}{2}\left\langle \frac{\mu}{\beta_1}, \frac{\mu}{\beta_1} \right\rangle - \frac{\beta_1}{2}\left\langle \frac{\mu}{\beta_1}, \frac{\mu}{\beta_1} \right\rangle \right\}$$

$$w_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\mathrm{H}\mathbf{w} - \mathrm{Y}\|_2^2 + \frac{\beta_1}{2}\left(\|\mathbf{w} - \mathbf{u}\|_2^2 - \frac{2}{\beta_1}\langle \mu, \mathbf{w} - \mathbf{u} \rangle + \left\langle \frac{\mu}{\beta_1}, \frac{\mu}{\beta_1} \right\rangle \right) - \frac{\beta_1}{2}\left\langle \frac{\mu}{\beta_1}, \frac{\mu}{\beta_1} \right\rangle \right\}$$

Then the notable square can be completed:

$$w_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\mathrm{H}\mathbf{w} - \mathrm{Y}\|_2^2 + \frac{\beta_1}{2}\left(\|\mathbf{w} - \mathbf{u}\|_2^2 - 2\left\langle \frac{\mu}{\beta_1}, \mathbf{w} - \mathbf{u} \right\rangle + \left\langle \frac{\mu}{\beta_1}, \frac{\mu}{\beta_1} \right\rangle \right) - \frac{\beta_1}{2}\left\langle \frac{\mu}{\beta_1}, \frac{\mu}{\beta_1} \right\rangle \right\}$$

$$w_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\mathrm{H}\mathbf{w} - \mathrm{Y}\|_2^2 + \frac{\beta_1}{2} \left\| \mathbf{w} - \mathbf{u} - \frac{\mu}{\beta_1} \right\|_2^2 - \frac{\beta_1}{2}\left\langle \frac{\mu}{\beta_1}, \frac{\mu}{\beta_1} \right\rangle \right\}$$

Since the minimization is performed with respect to **x**, then the last term can be eliminated and the subproblem **w** can be rewritten as:

$$w_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\mathrm{H}\mathbf{w} - \mathrm{Y}\|_2^2 + \frac{\beta_1}{2} \left\| \mathbf{w} - \left(\mathbf{u} + \frac{\mu}{\beta_1} \right) \right\|_2^2 \right\}$$

By performing the minimization:

$$\mathrm{H}^T(\mathrm{H}\mathbf{w} - \mathrm{Y}) + \beta_1 \left(\mathbf{w} - \left(\mathbf{u} + \frac{\mu}{\beta_1} \right) \right) = 0$$

And by grouping w the following expression is gotten:

$$(\mathrm{H}^T\mathrm{H} - \beta_1\mathrm{I})\mathbf{w} = \mathrm{H}^T Y + \beta_1\left(\mathbf{u} + \frac{\mu}{\beta_1}\right) \tag{41}$$

This is a linear system of the type Ax=b that can be solved by the method of conjugate gradient.

5.2.2 ADMM-TGV Subproblems with Tensor Modifications

Now, continuing with the modifications to the original subproblems of [40], Equations (42) and (43) show the two new problems derived from (39) to address the subproblems **z** and **x**.

Subproblem x:

$$x_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}}\left\{\mathcal{L}_{\mathcal{A}_{SRT}}\left(\mathrm{u}_k, \mathrm{p}_k, \mathrm{w}_k, \mathrm{x}_k, \mathrm{y}_k, \mathrm{z}_k; \mu_k, \xi_k, \eta_{k,}\, \theta_{k,}\right)\right\}$$

$$x_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}}\left\{\alpha_1\|\mathbf{x}\|_1 - \langle\theta, \mathbf{x} - \mathbf{Tz}\rangle + \frac{\beta_4}{2}\|\mathbf{x} - \mathbf{Tz}\|_2^2\right\} \tag{42}$$

Subproblem z:

$$z_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}}\left\{\mathcal{L}_{\mathcal{A}_{SRT}}\left(\mathrm{u}_k, \mathrm{p}_k, \mathrm{w}_k, \mathrm{x}_{k+1}, \mathrm{y}_k, \mathrm{z}_k; \mu_k, \xi_k, \eta_{k,}\, \theta_{k,}\right)\right\}$$

$$\mathbf{z}_{k+1} = \underset{\mathbf{z}}{\operatorname{argmin}} \Big\{ -\langle \theta, \mathbf{x} - \mathbf{T}\mathbf{z} \rangle + \frac{\beta_4}{2} \|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 - \langle \eta, \mathbf{z} - \nabla \mathbf{u} + \mathbf{p} \rangle + \frac{\beta_3}{2} \|\mathbf{z} - \nabla \mathbf{u} + \mathbf{p}\|_2^2 \Big\} \tag{43}$$

The first problem for $\mathbf{x}$ needs to be rewritten so that a quadratic completion can be performed and Equation (42) can be properly posed for a numerical optimization shrinkage process. Then, consider for a moment only the last two terms of (42) in the following development:

$$-\langle \theta, \mathbf{x} - \mathbf{T}\mathbf{z} \rangle + \frac{\beta_4}{2} \|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 = \left[-\frac{2}{\beta_4} \langle \theta, \mathbf{x} - \mathbf{T}\mathbf{z} \rangle + \|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 \right] \cdot \left(\frac{\beta_4}{2} \right)$$

$$-\langle \theta, \mathbf{x} - \mathbf{T}\mathbf{z} \rangle + \frac{\beta_4}{2} \|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 = \left[-2 \langle \frac{\theta}{\beta_4}, \mathbf{x} - \mathbf{T}\mathbf{z} \rangle + \|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 \right] \cdot \left(\frac{\beta_4}{2} \right)$$

Now by adding and subtracting $\left(\frac{\theta}{\beta_4}\right)^2$ the squared difference between $(\mathbf{x} - \mathbf{T}\mathbf{z})$ and $\left(\frac{\theta}{\beta_4}\right)$ can be completed:

$$-\langle \theta, \mathbf{x} - \mathbf{T}\mathbf{z} \rangle + \frac{\beta_4}{2} \|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 = \left[+ \left(\frac{\theta}{\beta_4}\right)^2 - \left(\frac{\theta}{\beta_4}\right)^2 - 2 \langle \frac{\theta}{\beta_4}, \mathbf{x} - \mathbf{T}\mathbf{z} \rangle + \|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 \right] \cdot \left(\frac{\beta_4}{2} \right)$$

$$-\langle \theta, \mathbf{x} - \mathbf{T}\mathbf{z} \rangle + \frac{\beta_4}{2} \|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 = \left[+ \left(\frac{\theta}{\beta_4}\right)^2 - 2 \langle \frac{\theta}{\beta_4}, \mathbf{x} - \mathbf{T}\mathbf{z} \rangle + \|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 - \left(\frac{\theta}{\beta_4}\right)^2 \right] \cdot \left(\frac{\beta_4}{2} \right)$$

$$-\langle \theta, \mathbf{x} - \mathbf{T}\mathbf{z} \rangle + \frac{\beta_4}{2} \|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 = \left[\left\| \frac{\theta}{\beta_4} - (\mathbf{x} - \mathbf{T}\mathbf{z}) \right\|_2^2 - \left(\frac{\theta}{\beta_4}\right)^2 \right] \cdot \left(\frac{\beta_4}{2} \right)$$

$$-\langle \theta, \mathbf{x} - \mathbf{T}\mathbf{z} \rangle + \frac{\beta_4}{2} \|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 = \left[\left\| (\mathbf{x} - \mathbf{T}\mathbf{z}) - \frac{\theta}{\beta_4} \right\|_2^2 - \left(\frac{\theta}{\beta_4}\right)^2 \right] \cdot \left(\frac{\beta_4}{2} \right)$$

Rewriting now (42) using this last result it is possible to have:

$$x_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \alpha_1 \|\mathbf{x}\|_1 + \left[\left\| (\mathbf{x} - \mathbf{Tz}) - \frac{\theta}{\beta_4} \right\|_2^2 - \left(\frac{\theta}{\beta_4} \right)^2 \right] \cdot \left(\frac{\beta_4}{2} \right) \right\}$$

Since this minimization is respect to **x**, then any term that doesn't contains a **x** can be eliminated as follows:

$$x_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \alpha_1 \|\mathbf{x}\|_1 + \frac{\beta_4}{2} \left\| (\mathbf{x} - \mathbf{Tz}) - \frac{\theta}{\beta_4} \right\|_2^2 \right\}$$

Then, rearranging this last expression, problem 1 can be redefined as:

Subproblem x:

$$x_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \|\mathbf{x}\|_1 + \frac{\beta_4}{2\alpha_1} \left\| \mathbf{x} - \left(\mathbf{Tz} + \frac{\theta}{\beta_4} \right) \right\|_2^2 \right\} \tag{44}$$

The previous formulation is actually the well-known proximal operator denoted as follows in [47]:

$$prox_{\lambda f}(v) = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ f(x) + \frac{1}{2\lambda} \|x - v\|_2^2 \right\}$$

Where $f = \|\cdot\|_1$ and $v = \mathbf{Tz} + \frac{\theta}{\beta_4}$. Thus:

$$x_{k+1} = prox_{\frac{\alpha_1}{\beta_4}\|\cdot\|_1} \left(\mathbf{Tz} + \frac{\theta}{\beta_4} \right) \tag{45}$$

The subproblem **x** can be solved by using the shrinkage operation as follows, for convenience the component-wise sub-indexes (i,j) are omitted:

$$x_{k+1} = \max\left\{\left\|\mathbf{T}\mathbf{z}_k + \frac{\theta_k}{\beta_4}\right\|_2 - \frac{\alpha_1}{\beta_4}, 0\right\} \times \frac{\mathbf{T}\mathbf{z}_k + \frac{\theta_k}{\beta_4}}{\left\|\mathbf{T}\mathbf{z}_k + \frac{\theta_k}{\beta_4}\right\|_2} \tag{46}$$

The shrinkage operation over the norm-1 proximal of Equation (45) is presented in detail in Appendix A – Norm 1 Proximal Shrinkage.

The other required change for super-resolution is the modification of the subproblem **z**. The minimization subproblem **z** can be rewritten so that it can be derived properly, thus the following modifications need to be applied to the Equation (43) terms in the following way:

$$\mathbf{z}_{k+1} = \underset{\mathbf{z}}{\operatorname{argmin}}\left\{-\langle\theta, \mathbf{x} - \mathbf{T}\mathbf{z}\rangle + \tfrac{\beta_4}{2}\|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 - \langle\eta, \mathbf{z} - \nabla\mathbf{u} + \mathbf{p}\rangle + \tfrac{\beta_3}{2}\|\mathbf{z} - \nabla\mathbf{u} + \mathbf{p}\|_2^2\right\}$$

$$\mathbf{z}_{k+1} = \underset{\mathbf{z}}{\operatorname{argmin}}\left\{-\langle\theta, \mathbf{x} - \mathbf{T}\mathbf{z}\rangle + \frac{\beta_4}{2}\|\mathbf{x} - \mathbf{T}\mathbf{z}\|_2^2 - \langle\eta, \mathbf{z} - \nabla\mathbf{u} + \mathbf{p}\rangle + \frac{\beta_3}{2}\|\mathbf{z} - (\nabla\mathbf{u} - \mathbf{p})\|_2^2\right\}$$

By expanding and rearranging some of the internal products as well as expanding the square of the differences $\|\mathbf{x} - \mathbf{T}\mathbf{z}\|$ and $\|\mathbf{z} - (\nabla\mathbf{u} - \mathbf{p})\|$, the following can be obtained:

$$\mathbf{z}_{k+1} = \underset{\mathbf{z}}{\operatorname{argmin}}\left\{-\langle\theta, \mathbf{x}\rangle + \langle\theta, \mathbf{T}\mathbf{z}\rangle + \frac{\beta_4}{2}(\mathbf{x}^T\mathbf{x} - 2\langle\mathbf{x}, \mathbf{T}\mathbf{z}\rangle + \langle\mathbf{T}\mathbf{z}, \mathbf{T}\mathbf{z}\rangle) - (\langle\eta, \mathbf{z}\rangle + \langle\eta, -\nabla\mathbf{u} + \mathbf{p}\rangle) + \frac{\beta_3}{2}\left(\|\mathbf{z}\|_2^2 - 2\langle\mathbf{z}, \nabla\mathbf{u} - \mathbf{p}\rangle + \|\nabla\mathbf{u} - \mathbf{p}\|_2^2\right)\right\}$$

Consider the following properties in the internal products:

$$\langle\theta, \mathbf{T}\mathbf{z}\rangle = (\mathbf{T}\mathbf{z})^T\theta = \mathbf{z}^T\mathbf{T}^T\theta = (\mathbf{T}^T\theta)^T\mathbf{z}$$

$$\langle\mathrm{x}, \mathbf{T}\mathbf{z}\rangle = (\mathbf{T}\mathbf{z})^T\mathrm{x} = \mathbf{z}^T\mathbf{T}^T\mathbf{x} = (\mathbf{T}^T\mathbf{x})^T\mathbf{z}$$

$$\langle \mathbf{Tz}, \mathbf{Tz} \rangle = (\mathbf{Tz})^T \mathbf{Tz} = \mathbf{z}^T \mathbf{T}^T \mathbf{Tz}$$

$$\langle \eta, \mathbf{z} \rangle = \eta^T \mathbf{z}$$

$$\|\mathbf{z}\|_2^2 = \langle \mathbf{z}, \mathbf{z} \rangle = \mathbf{z}^T \mathbf{z}$$

$$\langle \mathbf{z}, \nabla \mathbf{u} - \mathbf{p} \rangle = \langle \nabla \mathbf{u} - \mathbf{p}, \mathbf{z} \rangle = (\mathbf{z})^T (\nabla \mathbf{u} - \mathbf{p}) = (\nabla \mathbf{u} - \mathbf{p})^T \mathbf{z}$$

Thus, the expression to compute $\mathbf{z}_{k+1}$ can be now rewritten as:

$$\mathbf{z}_{k+1} = \underset{\mathbf{z}}{\operatorname{argmin}} \Big\{ -\langle \theta, \mathbf{x} \rangle + (\mathbf{T}^T \theta)^T \mathbf{z} + \frac{\beta_4}{2} (\mathbf{x}^T \mathbf{x} - 2(\mathbf{T}^T \mathbf{x})^T \mathbf{z} + \mathbf{z}^T \mathbf{T}^T \mathbf{T} \mathbf{z}) - (\eta^T \mathbf{z} + \langle \eta, -\nabla \mathbf{u} + \mathbf{p} \rangle) + \frac{\beta_3}{2} (\mathbf{z}^T \mathbf{z} - 2(\nabla \mathbf{u} - \mathbf{p})^T \mathbf{z} + \|\nabla \mathbf{u} - \mathbf{p}\|_2^2) \Big\}$$

Finally, by grouping the terms with respect to the role of **z**:

$$\mathbf{z}_{k+1} = \underset{\mathbf{z}}{\operatorname{argmin}} \Big\{ \mathbf{z}^T \left(\frac{\beta_4}{2} \mathbf{T}^T \mathbf{T} + \frac{\beta_3}{2} I \right) \mathbf{z} + \left((\mathbf{T}^T \theta)^T + \frac{\beta_4}{2} (-2(\mathbf{T}^T \mathbf{x})^T) - \eta^T + \frac{\beta_3}{2} (-2(\nabla \mathbf{u} - \mathbf{p})^T) \right) \mathbf{z} + \left(-\langle \theta, \mathbf{x} \rangle + \frac{\beta_4}{2} \mathbf{x}^T \mathbf{x} - \langle \eta, -\nabla \mathbf{u} + \mathbf{p} \rangle + \frac{\beta_3}{2} \|\nabla \mathbf{u} - \mathbf{p}\|_2^2 \right) \Big\}$$

Then, the last term can be eliminated from this minimization problem since no part of it contains the variable **z**. This leads to redefine subproblem **z** with the next expression:

Subproblem z:

$$\mathbf{z}_{k+1} = \underset{\mathbf{z}}{\operatorname{argmin}} \left\{ \mathbf{z}^T \left(\frac{\beta_4}{2} \mathbf{T}^T \mathbf{T} + \frac{\beta_3}{2} I \right) \mathbf{z} + \left((\mathbf{T}^T \theta)^T + \frac{\beta_4}{2} (-2(\mathbf{T}^T \mathbf{x})^T) - \eta^T + \frac{\beta_3}{2} (-2(\nabla \mathbf{u} - \mathbf{p})^T) \right) \mathbf{z} \right\} \tag{47}$$

The minimization subproblem **z** can be solved by simply differentiating the argument of (47) and making that result equal to zero. Then it is possible to obtain the following reduced expression which leads us directly to a formula for **z**:

$$(\beta_4 \mathbf{T}\mathbf{T} + \beta_3 I)\mathbf{z} = -\mathbf{T}\theta + \beta_4 \mathbf{T}\mathbf{x} + \eta + \beta_3 (\nabla \mathbf{u} - \mathbf{p})$$

$$\mathbf{z}_{k+1} = (\beta_4 \mathbf{T}\mathbf{T} + \beta_3 I)^{-1} \left(-\mathbf{T}\theta + \beta_4 \mathbf{T}\mathbf{x} + \eta + \beta_3 (\nabla \mathbf{u} - \mathbf{p}_k) \right) \tag{48}$$

5.2.3 ADMM-TGV Subproblems without Modifications

As previously mentioned the rest of the subproblem components remain as originally defined in [40] and only their final formulas are listed here with a few additional steps to link the original formulas to closer expressions implemented in the code, so that subsequently the overall algorithm pseudocode can be introduced. All of these problems were not affected by the change from the denoising application to the super-resolution and neither by the tensor insertion.

The first described subproblem is the one that defines the variable **u**, which is the actual reconstructed and upscaled image in each iteration.

Subproblem u:

$$\mathbf{u}_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}}\{\mathcal{L}_{\mathcal{A}_{SRT}}(\mathrm{u}_k, \mathrm{p}_k, \mathrm{w}_k, \mathrm{x}_{k+1}, \mathrm{y}_k, \mathrm{z}_{k+1}; \mu_k, \xi_k, \eta_k, \theta_{k,})\}$$

$$\mathbf{u}_{k+1} = \underset{\mathbf{u}}{\operatorname{argmin}} \left\{ \frac{\beta_1}{2} \left\| \mathbf{w}_k - \mathbf{u} - \frac{\mu_k}{\beta_1} \right\|_2^2 + \frac{\beta_3}{2} \left\| \mathbf{z}_k - \nabla \mathbf{u} + \mathbf{p}_k - \frac{\eta_k}{\beta_3} \right\|_2^2 \right\}$$

Subproblem **u** can be solved with the following equation:

$$\mathbf{u}_{k+1} = \left(\frac{\beta_1}{\beta_3} I + \nabla^T \nabla \right)^{-1} \left(\frac{\beta_1}{\beta_3} \left(\mathbf{w}_k - \frac{\mu_k}{\beta_1} \right) + \nabla^T \left(\mathbf{z}_k + \mathbf{p}_k - \frac{\eta_k}{\beta_3} \right) \right) \tag{49}$$

The actual implementation in software uses the two-dimensional Discrete Fourier Transform (DFT) and Inverse Discrete Fourier Transform (IDFT) in similar way as done in [48] but this time applied to the ADMM formula of Equation (49) as follows:

$$\mathbf{u}_{k+1} = \mathcal{F}^{-1} \left\{ \left(\frac{\beta_1}{\beta_3} + \mathcal{F}^*(\nabla_{x_1}) \circ \mathcal{F}(\nabla_{x_1}) + \mathcal{F}^*(\nabla_{x_2}) \circ \mathcal{F}(\nabla_{x_2}) \right)^{-1} \left(\frac{\beta_1}{\beta_3} \mathcal{F} \left(\boldsymbol{w}_k - \frac{\mu_k}{\beta_1} \right) \right. \right.$$
$$\left. \left. + \mathcal{F}^*(\nabla_{x_1}) \circ \mathcal{F} \left(\mathbf{z_1}_k + \mathbf{p_1}_k - \frac{\eta_k}{\beta_3} \right) + \mathcal{F}^*(\nabla_{x_2}) \circ \mathcal{F} \left(\mathbf{z_2}_k + \mathbf{p_2}_k - \frac{\eta_k}{\beta_3} \right) \right) \right\}$$

Where $\mathcal{F}$ is the DFT, "$*$" is the complex conjugate and "$\circ$" the component-wise multiplication. Note that the last operations require that the matrices be block circulants so that they can be diagonalized by the DFTs, See Appendix B for more details on this matter.

The following subproblem defines the computation of the variable **p** which was introduced originally by the TGV functional expression in Equation (22).

Subproblem p:

$$\mathbf{p}_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}}\{\mathcal{L}_{\mathcal{A}_{SRT}}(\mathrm{u}_{k+1}, \mathrm{p}_k, \mathrm{w}_k, \mathrm{x}_{k+1}, \mathrm{y}_k, \mathrm{z}_{k+1}; \mu_k, \xi_k, \eta_{k,}\, \theta_{k,})\}$$

$$\mathbf{p}_{k+1} = \underset{\mathbf{p}}{\operatorname{argmin}} \left\{ \frac{\beta_2}{2} \left\| \mathbf{y}_k - \varepsilon(\mathbf{p}) - \frac{\boldsymbol{\xi}_k}{\beta_2} \right\|_2^2 + \frac{\beta_3}{2} \left\| \mathbf{z}_k - \nabla \mathbf{u}_{\boldsymbol{k+1}} + \mathbf{p} - \frac{\eta_k}{\beta_3} \right\|_2^2 \right\}$$

The solution of **p** leads to the subproblems p_1 and p_2, which are the components of **p** and are defined as follows:

$$\begin{aligned} &\left(\beta_2 \nabla_{x_1}^T \nabla_{x_1} + \frac{\beta_2}{2} \nabla_{x_2}^T \nabla_{x_2} + \beta_3 I\right) \mathrm{p}_{1_{k+1}} \\ &\qquad = \beta_2 \left[\nabla_{x_1}^T \left(y_{1_k} - \frac{\boldsymbol{\xi}_{1_k}}{\beta_2} \right) + \nabla_{x_2}^T \left(y_{3_k} - \frac{\boldsymbol{\xi}_{3_k}}{\beta_2} - \frac{1}{2} \nabla_{x_1} \mathrm{p}_{2_k} \right) \right] \\ &\qquad + \beta_3 \left(\nabla_{x_1} \mathbf{u}_{\boldsymbol{k+1}} + \frac{\eta_{1_k}}{\beta_3} - \mathbf{z}_{1_k} \right) \end{aligned} \tag{50}$$

$$\begin{aligned} &\left(\beta_2 \nabla_{x_2}^T \nabla_{x_2} + \frac{\beta_2}{2} \nabla_{x_1}^T \nabla_{x_1} + \beta_3 I\right) \mathrm{p}_{2_{k+1}} \\ &\qquad = \beta_2 \left[\nabla_{x_2}^T \left(y_{2_k} - \frac{\boldsymbol{\xi}_{2_k}}{\beta_2} \right) + \nabla_{x_1}^T \left(y_{3_k} - \frac{\boldsymbol{\xi}_{3_k}}{\beta_2} - \frac{1}{2} \nabla_{x_2} \mathrm{p}_{1_{k+1}} \right) \right] \\ &\qquad + \beta_3 \left(\nabla_{x_2} \mathbf{u}_{\boldsymbol{k+1}} + \frac{\eta_{2_k}}{\beta_3} - \mathbf{z}_{2_k} \right) \end{aligned} \tag{51}$$

Notice that the expression for p_2 uses the new p_1, thus it is required that Equation (50) be computed first in the algorithm.

The method used by [40] to solve the last two Equations (50) and (51) employs DFTs and their inverses, assuming the circular boundary condition of the images as it was shown for problem **u** previously. Thus, these two equations can be rewritten as follows:

$$\begin{aligned} \mathrm{p}_{1_{k+1}} = \mathcal{F}^{-1}\Bigg\{ & \left(\beta_2 \mathcal{F}^*(\nabla_{x_1}) \circ \mathcal{F}(\nabla_{x_1}) + \frac{\beta_2}{2} \mathcal{F}^*(\nabla_{x_2}) \circ \mathcal{F}(\nabla_{x_2}) + \beta_3\right)^{-1} \beta_2 \Bigg[\mathcal{F}^*(\nabla_{x_1}) \mathcal{F}\left(y_{1_k} \right. \\ & \left. - \frac{\boldsymbol{\xi}_{1_k}}{\beta_2} \right) + \mathcal{F}^*(\nabla_{x_2}) \mathcal{F}\left(y_{3_k} - \frac{\boldsymbol{\xi}_{3_k}}{\beta_2} - \frac{1}{2} \nabla_{x_1} \mathrm{p}_{2_k} \right) \Bigg] \\ & + \beta_3 \left(\mathcal{F}(\nabla_{x_1}) \circ \mathcal{F}(\mathbf{u_{k+1}}) + \mathcal{F}\left(\frac{\eta_{1_k}}{\beta_3} - \mathbf{z}_{1_k} \right) \right) \Bigg\} \end{aligned}$$

$$\begin{aligned} \mathrm{p}_{2_{k+1}} = \mathcal{F}^{-1}\Bigg\{ & \left(\beta_2 \mathcal{F}^*(\nabla_{x_2}) \circ \mathcal{F}(\nabla_{x_2}) + \frac{\beta_2}{2} \mathcal{F}^*(\nabla_{x_1}) \circ \mathcal{F}(\nabla_{x_1}) + \beta_3\right)^{-1} \beta_2 \Bigg[\mathcal{F}^*(\nabla_{x_2}) \mathcal{F}\left(y_{2_k} \right. \\ & \left. - \frac{\boldsymbol{\xi}_{2_k}}{\beta_2} \right) + \mathcal{F}^*(\nabla_{x_1}) \mathcal{F}\left(y_{3_k} - \frac{\boldsymbol{\xi}_{3_k}}{\beta_2} - \frac{1}{2} \nabla_{x_2} \mathrm{p}_{1_k} \right) \Bigg] \\ & + \beta_3 \left(\mathcal{F}(\nabla_{x_2}) \circ \mathcal{F}(\mathbf{u_{k+1}}) + \mathcal{F}\left(\frac{\eta_{1_k}}{\beta_3} - \mathbf{z}_{1_k} \right) \right) \Bigg\} \end{aligned}$$

Finally, the subproblem **y** is defined and solved in the following way.

Subproblem y:

$$\mathbf{y}_{k+1} = \underset{\mathbf{y}}{\operatorname{argmin}}\left\{\mathcal{L}_{\mathcal{A}_{SRT}}\left(\mathrm{u}_k, \mathrm{p}_k, \mathrm{w}_k, \mathrm{x}_k, \mathrm{y}_k, \mathrm{z}_k; \mu_k, \xi_k, \eta_{k,}\, \theta_{k,}\right)\right\}$$

$$\mathbf{y}_{k+1} = \underset{\mathbf{y}}{\operatorname{argmin}}\left\{\|\mathbf{y}\|_1 + \frac{\beta_2}{2\alpha_0}\left\|\mathbf{y} - \left(\varepsilon(\mathbf{p}_{k+1}) + \frac{\boldsymbol{\xi}_k}{\beta_2}\right)\right\|_2^2\right\}$$

The solution for this subproblem employs the same shrinkage technique as was done in the case of the subproblem **x** in Equation (46) of the previous section and detailed derivation of the solution formula can be found in Appendix A – Norm 1 Proximal Shrinkage. For convenience the component wise sub-indexes (i,j) were omitted:

$$\mathbf{y}_{k+1} = \max\left\{\left\|\varepsilon(\mathbf{p}_{k+1}) + \frac{\boldsymbol{\xi}_k}{\beta_2}\right\|_2 - \frac{\alpha_0}{\beta_2}, 0\right\} \times \frac{\varepsilon(\mathbf{p}_{k+1}) + \frac{\boldsymbol{\xi}_k}{\beta_2}}{\left\|\varepsilon(\mathbf{p}_{k+1}) + \frac{\boldsymbol{\xi}_k}{\beta_2}\right\|_2} \tag{52}$$

Finally, the dual variables μ_{k+1}, ξ_{k+1}, η_{k+1} and θ_{k+1} are computed as follows:

$$\mu_{k+1} = \mu_k + \beta_1(\mathbf{w}_{k+1} - \mathbf{u}_{k+1}) \tag{53}$$

$$\xi_{k+1} = \xi_k + \beta_2\left(\mathbf{y}_{k+1} - \varepsilon(\mathbf{u}_{k+1})\right) \tag{54}$$

$$\eta_{k+1} = \eta_k + \beta_3(\mathbf{z}_{k+1} - \nabla\mathbf{u}_{k+1} + \mathbf{p}_{k+1}) \tag{55}$$

$$\theta_{k+1} = \theta_k + \beta_4(\mathbf{x}_{k+1} - \mathbf{T}\mathbf{z}_{k+1}) \tag{56}$$

Notice that θ is not part of the original dual variables, however it was included in this section with the rest of the parameters for convenience in the description. As mentioned in the

previous section, θ was introduced to address the relation between **x** and **Tz** just as the other parameters do with the original and auxiliary variables.

5.2.4 ADMM-TGV with Tensor Algorithm

Once there is an expression for each of the ADMM-TGV with Tensor subproblems, the pseudocode can be described in the following way which differs from the one presented in [40] in several ways due to the new introduced variables, terms and final application.

Algorithm 3 ADMM-TGV with Tensor

Require: Tensor, H, Y, Λ and u_0.

1. Initialize: u_0, p_0, w_0, x_0, y_0, z_0, μ_0, ξ_0, η_0, θ_0.
2. k=0.
3. **while** stopping criteria not satisfied **do**
    a. Compute $\mathbf{y}_{k+1}$ according to (52);
    b. Compute $\mathbf{x}_{k+1}$ according to (46);
    c. Compute $\mathbf{z}_{k+1}$ according to (48);
    d. Compute $\mathbf{u}_{k+1}$ according to (49);
    e. Compute $\mathbf{p}_{k+1}$ according to (50) and (51);
    f. Solve for $\mathbf{w}_{k+1}$ the linear system of (41):
        i. Left_side $\leftarrow (\mathrm{H}^T\mathrm{H} - \beta_1\mathrm{I})$;
        ii. Right_side $\leftarrow \mathrm{H}^T Y + \beta_1 \left(\mathbf{u} + \frac{\mu}{\beta_1}\right)$;
        iii. $\mathbf{w}_{k+1} \leftarrow$ Conjugate_Gradient (Left_side, Right_side);
    g. Update parameters:
        i. Update μ_0 according to (53);
        ii. Update ξ_0 according to (54);
        iii. Update η_0 according to (55);
        iv. Update θ_0 according to (56);
    h. k = k+1;
4. End of **while**.
5. Return $\mathbf{u}_{k+1}$.

In this way, Algorithm 3 works in conjunction with Algorithm 2, previously described in section 4.2.1 Theoretical derivation of the noise detection scheme, by providing the updated high resolution image X for every given iteration of the main loop in Algorithm 2. The next section covers the required descriptions and analysis to provide a Tensor that was introduced by this last algorithm.

Chapter 6: Tensor for Depth Super-Resolution

6.1 Multichannel Tensor Fusion

This research proposes a modification of the ADMM-TGV applied to super-resolution by using an inserted tensor in the TGV term as show in the previous section. In regards to the Tensor construction, this section presents the initial step to build the tensor departing from the set Y of low resolution observations.

Since a single Tensor T is provided to the Algorithm 3, it is necessary to fusion the L low resolution observations in a reliable way. To do this task the work performed by Silvano Di Senzo in [49] was selected.

A method to detect edges from m-channel images, based on the gradient is introduced in [49]. The proposed solution pretends to combine the m channels of an image into one single output.

In this way, the technique takes advantage of differential geometry by regarding:

1. Analytical treatment of images as vector fields.
2. Vector local properties via difference operators rather than gray levels.

Although Di Zenzo uses his technique on a color image application to compute the gradient over the three-color channels: red, green and blue, this concept is applied here to m number of channels, which are the properly registered upscaled low-resolution observations in which the gradient of each frame will reinforce the gradient of the single global tensor output components.

The problem is formulated by defining a function *F* that maps the square of the length of a vector created by a function *f* evaluated in two different points, **x** and **x'**. In this way, a small displacement between $f(\mathbf{x})$ and $f(\mathbf{x}')$ is quantified by the following equation:

$$F(\theta) = |f(\mathbf{x}') - f(\mathbf{x})|^2 \tag{57}$$

Where: $\mathbf{x} = (x_1, y_1)$; $\mathbf{x}' = (x_2, y_2)$

F depends on the angle θ, thus regard the following diagram to pose *F* in terms of this angle as depicted in [49].

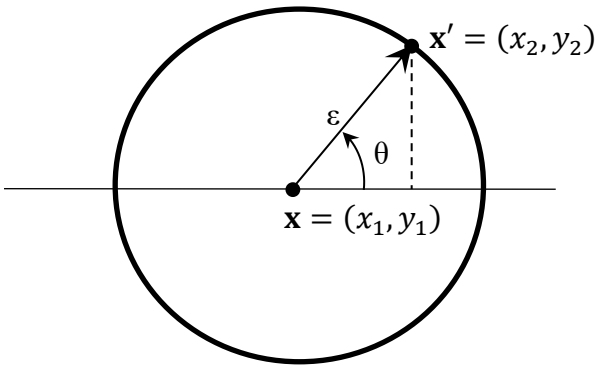


Figure 52: x and x' geometry.

Consider the following trigonometric results from the diagram of Figure 52:

$$\cos(\theta) = \frac{x_2 - x_1}{\varepsilon} = \frac{\Delta x}{\varepsilon}$$

$$\sin(\theta) = \frac{y_2 - y_1}{\varepsilon} = \frac{\Delta y}{\varepsilon}$$

Solving for the coordinates of the $\mathbf{x}'$ in terms of **x** coordinates:

$$x_2 = x_1 + \varepsilon \cdot \cos\theta$$

$$y_2 = y_1 + \varepsilon \cdot \sin\theta$$

Then Equation (57) can be rewritten as:

$$F(\theta) = |f(x_1, y_1) - f(x_1 + \varepsilon \cdot \cos\theta\,,\ y_1 + \varepsilon \cdot \sin\theta\,)|^2$$

By extending the Euclidean distance to the rest of the m image channels, then each square distance magnitude will be added to the others in the following way:

$$F(\theta) = \sum_{i=1}^{m} [f_i(x_1, y_1) - f_i(x_1 + \varepsilon \cdot \cos(\theta)\,,\ y_1 + \varepsilon \cdot \sin(\theta))]^2 \tag{58}$$

Equation (58) can be decomposed using Taylor series:

$$F(\theta) = f_i(x_1, y_1) - f_i(x_1 + h_1,\ y_1 + h_2) = \nabla f_i(x_1, y_1) \cdot (h_1, h_2) + O(\|(h_1, h_2)\|) =$$

$$F(\theta) = \frac{\partial f_i(x_1, y_1)}{\partial x_1} \cdot h_1 + \frac{\partial f_i(x_1, y_1)}{\partial y_1} \cdot h_2 + O(\|(h_1, h_2)\|)$$

Disregarding the error term O, the last expression can be rewritten as follows:

$$\begin{aligned} F(\theta) &= f_i(x_1, y_1) - f_i(x_1 + h_1,\ y_1 + h_2) \\ &\approx \frac{\partial f_i(x_1, y_1)}{\partial x_1} \cdot h_1 + \frac{\partial f_i(x_1, y_1)}{\partial y_1} \cdot h_2 \end{aligned} \tag{59}$$

Where: $h_1 = \varepsilon \cdot \cos\theta$ and $h_2 = \varepsilon \cdot \sin\theta$

Then Equation (59) can be approximated by:

$$F(\theta) \approx \sum_{i=1}^{m} \left[\frac{\partial f_i(x_1, y_1)}{\partial x_1} \cdot h_1 + \frac{\partial f_i(x_1, y_1)}{\partial y_1} \cdot h_2 \right]^2$$

$$F(\theta) \approx \sum_{i=1}^{m} \left[\frac{\partial f_i(x_1, y_1)}{\partial x_1} \cdot \varepsilon \cdot \cos\theta + \frac{\partial f_i(x_1, y_1)}{\partial y_1} \cdot \varepsilon \cdot \sin\theta \right]^2$$

Expanding the square sum:

$$F(\theta) \approx \sum_{i=1}^{m} \left[\left(\frac{\partial f_i(x_1, y_1)}{\partial x_1} \cdot \varepsilon \cdot \cos\theta \right)^2 + 2 \cdot \left(\frac{\partial f_i(x_1, y_1)}{\partial x_1} \cdot \varepsilon \cdot \cos\theta \right) \left(\frac{\partial f_i(x_1, y_1)}{\partial y_1} \cdot \varepsilon \cdot \sin\theta \right) + \left(\frac{\partial f_i(x_1, y_1)}{\partial y_1} \cdot \varepsilon \cdot \sin\theta \right)^2 \right] =$$

$$\varepsilon^2 \sum_{i=1}^{m} \left(\frac{\partial f_i(x_1, y_1)}{\partial x_1} \right)^2 \cos^2\theta + 2\varepsilon^2 \sum_{i=1}^{m} \left(\frac{\partial f_i(x_1, y_1)}{\partial x_1} \right) \left(\frac{\partial f_i(x_1, y_1)}{\partial y_1} \right) \sin\theta \cos\theta + \varepsilon^2 \sum_{i=1}^{m} \left(\frac{\partial f_i(x_1, y_1)}{\partial y_1} \right)^2 \sin^2\theta$$

And now rearranging the last approximation:

$$F(\theta) \approx \varepsilon^2 \left\{ \left[\sum_{i=1}^{m} \left(\frac{\partial f_i(x_1, y_1)}{\partial x_1} \right)^2 \right] \cos^2\theta + 2 \left[\sum_{i=1}^{m} \left(\frac{\partial f_i(x_1, y_1)}{\partial x_1} \right) \left(\frac{\partial f_i(x_1, y_1)}{\partial y_1} \right) \right] \sin\theta \cos\theta + \left[\sum_{i=1}^{m} \left(\frac{\partial f_i(x_1, y_1)}{\partial y_1} \right)^2 \right] \sin^2\theta \right\}$$

Assuming that the approximation is good enough and introducing terms g11, g12 and g22 to have a more compact expression, $F(\theta)$ can be defined as:

$$F(\theta) = g_{11}\cos^2\theta + 2g_{12}\sin\theta\cos\theta + g_{22}\sin^2\theta \tag{60}$$

Where:

$$g_{11} = \sum_{i=1}^{m}\left(\frac{\partial f_i(x_1, y_1)}{\partial x_1}\right)^2 \tag{61}$$

$$g_{12} = \sum_{i=1}^{m}\left(\frac{\partial f_i(x_1, y_1)}{\partial x_1}\right)\left(\frac{\partial f_i(x_1, y_1)}{\partial y_1}\right) \tag{62}$$

$$g_{22} = \sum_{i=1}^{m}\left(\frac{\partial f_i(x_1, y_1)}{\partial y_1}\right)^2 \tag{63}$$

The general case for these coefficients can be defined by:

$$g_{hk}(\mathbf{x}) = f_h(\mathbf{x}) \times f_k(\mathbf{x})^T$$

Where *f* is an m-tuple of partial derivatives:

$$f_h(\mathbf{x}) = \left(\frac{\partial f_1}{\partial x_h}, \cdots, \frac{\partial f_m}{\partial x_h}\right)$$

$$f_k(\mathbf{x}) = \left(\frac{\partial f_1}{\partial x_k}, \cdots, \frac{\partial f_m}{\partial x_k}\right)$$

Also consider that h, k=1,2 are the basis for the R^2 application for image processing of interest in this research. The minimization of Equation (60) by setting $\frac{dF(\theta)}{d\theta} = 0$ generates an expression useful to compute the gradient of a multichannel image which after some manipulation has the form:

$$\theta = \frac{1}{2} arctan\left(\frac{2g_{12}}{(g_{11} - g_{22})}\right) \tag{64}$$

Although Equation (60) holds the foundation to compute the gradient of a multichannel image, the most useful result from [49] for this work is the actual summation of the square derivatives in each g_{hk} term, which are the gradients in each spatial direction of the image plane. Furthermore, the terms g_{11}, $g_{12}$and g_{22} are actually the structure tensor coefficients that for the purposes of this work can be computed over multiple L low-resolution observations, this is m = L. In this way, the specific coefficients computed in [49] are not used in the formation of the tensor provided to Algorithm 3. On the other hand, the Equations (61), (62) and (63) are used to fusion all the gradients that will help to implement the structure tensor that is fed to Algorithm 3.

6.2 Structure Tensor Frame Preprocessing

The basic structure tensor that has been incorporated in this research is based on the structure tensor proposed by Köthe in [50], but with the extension of multiple frame fusion. There are three proposed structure tensor improvements in [50] which are: sampling rate increment for edge enhancement, non-linear spatial integration and integrated edge and junction detection. However, only the first feature contributed with good performance in Algorithm 3. This section

introduces the typical structure tensor and briefly descries the improvements from Köthe that were evaluated in the performance of Algorithm 3.

First let us define the set of images which will be processed to generate a structure tensor. Several combinations of interpolation techniques and low-resolution observations were tested such as: one frame upscaled using bicubic interpolation and the rest just as spare images with information from the low-resolution observation registered in a high-resolution grid, all frames upscaled for the first few iterations, then commute to feed the current high-resolution solution $\mathbf{u}_{k+1}$ generated by Algorithm 3, but the scheme with best PSNR and SSIM metrics was the following method described in Figure 53.

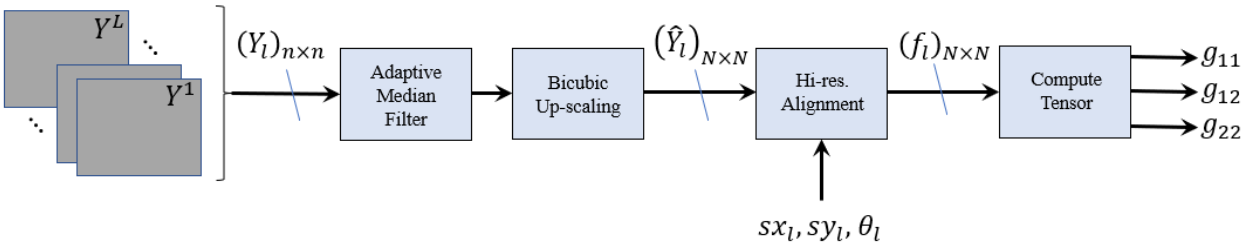


Figure 53 Multi-frame Tensor Computation Block Diagram.

Figure 53 shows the block diagram of the general scheme to compute the tensor that is used in the regularization framework of this research, having as input L low-resolution observations and as outputs the three components of the structure tensor g_{11}, g_{12} and g_{22}. All the low-resolution observation Y_l are filtered with an Adaptive Median Filter to remove some of the Gaussian noise and impulsive noise. Then the cleaned L frames are up-scaled using the bicubic interpolation method and changing in this way their initial resolution from $n \times n$ to $N \times N$, where $N = rn$ and r is the magnification factor as was previously mentioned. Once the frames $\hat{Y}_l$ are clean and at the high spatial resolution, the next preprocessing step is to align them using the last

available updated registration parameters sx_l, sy_l and θ_l. Finally, the clean, up-scales and aligned frames f_l are fed into the multi-frame block. The next part of this section describes how the structure tensor is computed.

6.3 Structure Tensor Computation

The first step in the Structure Tensor computation is to perform a pre-smoothing process (ps), thus consider the convolution of all frames $f_l(x, y)$ with a Gaussian kernel h of a given σ to generate pre-smoothed frames:

$$f_{ps_l}(x, y) = h \star f_l(x, y)$$

Where the operator $\star$ denotes a spatial convolution. Then, the gradient of the pre-smoothed observations $f_{ps_l}(x, y)$ needs to be computed in each component:

$$g_{11_l} = \left(\frac{\partial f_{ps_l}(x, y)}{\partial x}\right)^2 \tag{65}$$

$$g_{22_l} = \left(\frac{\partial f_{ps_l}(x, y)}{\partial y}\right)^2 \tag{66}$$

$$g_{12_l} = \left(\frac{\partial f_{ps_l}(x, y)}{\partial x}\right)\left(\frac{\partial f_{ps_l}(x, y)}{\partial y}\right) \tag{67}$$

The applied improvement from Köthe to the structure tensor implements a sampling process higher than the original one in which the f_l frames were received, which leads to improvements in the edges and corner detection [50]. The Köthe motivation for correct sampling

is to avoid aliasing due to spatial multiplications and lose information since the gradient is represented with half the sample distance of the original image in such a scheme. The Köthe approach implementation performed by Gabriel Peyre in [51] produces the correct sampling rate by defining a grid to double the original sampling rate, creating half integer positions for the new inserted rows and columns. Thus, the higher sample density overcomes the issues generated by an insufficient sampling density over the computation of the products of derivatives. The derivatives that are evaluated in the interpolated positions at integers and half integers can be defined as: $\hat{g}_{11_l}$, $\hat{g}_{12_l}$ and $\hat{g}_{22_l}$. Note that the dimension of these tensor components will be matched to the product operation which doubles the bandwidth $sub \times N^2$, where sub factor was left as the original code equal to 2.

Now by apply Equations (61), (62), (63) to the oversampled derivatives $\hat{g}_{11_l}$, $\hat{g}_{12_l}$ and $\hat{g}_{22_l}$ a global structure tensor can be obtained as follows:

$$\hat{g}_{11}(x,y) = \sum_{l=1}^{L} g_{11_l} \tag{68}$$

$$\hat{g}_{12}(x,y) = \sum_{l=1}^{L} g_{12_l} \tag{69}$$

$$\hat{g}_{22}(x,y) = \sum_{l=1}^{L} g_{22_l} \tag{70}$$

Where L is the total number of bilinear upscaled low-resolution observations. Note that every observation is joined with equal weight, since all of them were acquired with the same sensor and with the same bandwidth. Nevertheless, other investigations like [52] for multi-spectral images have suggested to use different weights based on intrinsic properties of each observation

considering that they were collected with different acquisition conditions or device configurations, which is not the case for the multiple frames of this study. Future work can consider certain heuristics in which if a given frame has some higher level of blurring or noise than the others, the algorithm can decrease its contribution to the final tensor.

Additionally, it was evaluated to divide each tensor component by the number of frames L, to keep the proportionality, however initial tests shown that by doing this the performance is slightly worse, thus the division by L was discarded.

Continuing with the structure tensor implementation, the eigenvalues of this symmetric matrix are modified via a decomposition of the following form:

$$\widehat{\mathbf{T}} = \begin{bmatrix} \hat{g}_{11} & \hat{g}_{12} \\ \hat{g}_{21} & \hat{g}_{22} \end{bmatrix} = S^T \cdot S = \ Q \cdot \lambda \cdot Q^T \tag{71}$$

Where Q contains the eigenvector columns and λ contains a diagonal matrix with the eigenvalues:

$$Q^T = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}$$

$$\lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

The Köthe implementation contains the eigenvalues modification which is chosen in this work and defined as follows:

$$\lambda_1 = \sigma_0 \cdot \eta_0; \qquad \lambda_2 = \frac{\sigma_0}{\eta_0}$$

Where the best eigenvalues for the algorithm performance, among several tested even from other tensor implementations like [30], were σ_0=8 and $\eta_0 = 6$. Besides these selected values obtained by experimentation, another section will present a strategy for eigenvalue modification.

Finally, a down-sampling process is applied to the oversampled tensor components $\hat{g}_{11}$, $\hat{g}_{22}$ and $\hat{g}_{12}$ so that the final tensor components g_{11}, g_{22} and g_{12} are generated with a size of N^2 again. It is important to mention that the down-sampling process is done just by taking the values at integer positions without any class of anti-aliasing filtering, other scheme could be tested to evaluate if this simple process can be improved in a future research.

6.4 Structure Tensor Properties and Visualization

This section shows some of the properties of the structure tensor used for this work by discussing the computed eigenvalues and eigenvectors at given positions of an image. It is common to list eigenvalues in decreasing order, in this case λ_1 is the largest. Next, the Structure Tensor was computed for a single frame which is the image Art from the dataset Middlebury just for exemplary purposes. The four components of the tensor are shown in Figure 54. In this case for the single image the structure tensor is the outer product of the gradient with itself and therefore there is only one non-zero eigenvalue.

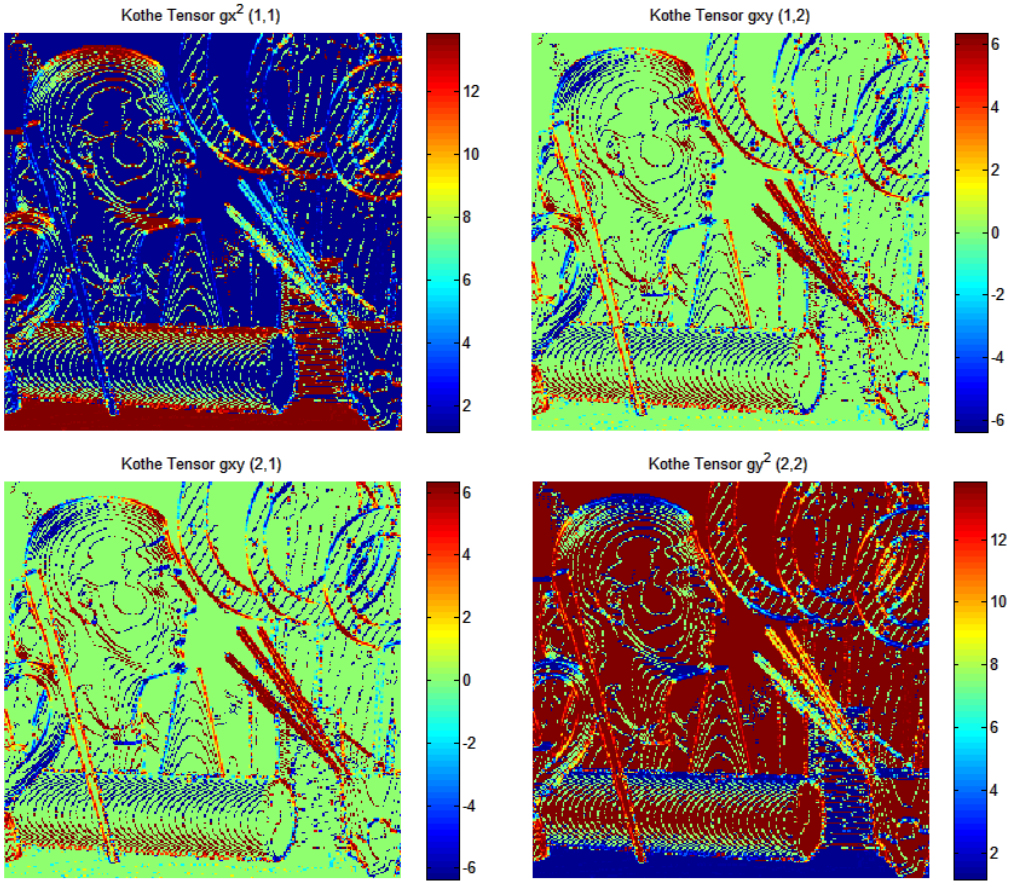


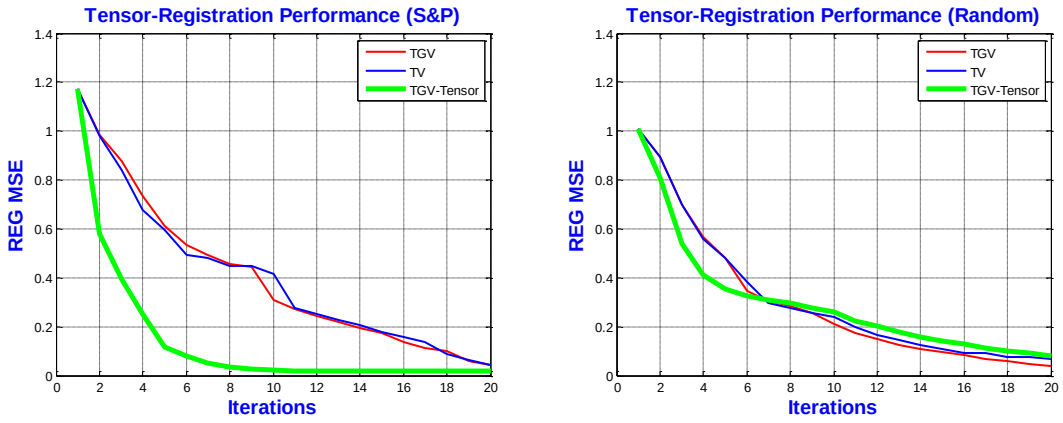
Figure 54: Structure Tensor of Image Art from Middlebury Dataset.

In the case of a tensor that has been computed from multiple images both eigenvalues are non-zero. The largest eigenvalue $\lambda_1$is associated with the normal direction to the edges. Making this eigenvalue larger compared to λ_2 is a simple and effective method of improving image reconstruction and therefore registration. This is the main contribution of the tensor modification strategy in this work. A different section it will evaluate a strategy to vary the eigenvalues accordingly to the image location so that the values have small and similar values between them in a smooth region.

6.5 Structure Tensor Characterization

6.5.1 Tensor-Registration Sinergy

The Lukas-Kanade registration process and the TGV regularization with the Tensor are extremely linked in the overall iterative algorithm. This close cooperative performance is shown in this sub-section by plotting the mean squared error evolution of the parameter estimation frame average of each iteration, similar to the recorded case shown in Figure 24, which was generated using only the TGV regularization term. This time the characterization was performed comparing the three regularization terms (TGV, TGV w/Tensor & TV) to appreciate the tensor impact on the algorithm under two different types of impulsive noise: Salt and Pepper and Random Value. This comparison is shown in Figure 55.



(a) (b)

Figure 55: 10-Frames Tensor Registration Performance under: (a) Salt & Pepper Noise (b) Random Value Noise.

As can be seen in Figure 55 the TGV with tensor promotes a faster alignment of the frames while the TGV and the TV keep a very similar behavior. Although this is definitely an outstanding execution of the tensor, it is also necessary to note that the ADMM with the TGV and the tensor

perform more operations at each iteration due to the additional subproblems shown in the previous sections. This amount of extra time will be reported in the results of the section 7.1 Super-Resolution Performance vs. Number of Frames.

6.5.2 Tensor Components Performance

The multichannel fusion introduced in section 6.1 Multichannel Tensor Fusion and applied over the low-resolution observations initially points to an iterative improvement of the tensor in regards to improvements in the edge directions. This is, it is expected that the tensor from the first iteration will have no well-defined components which generates low and wide peaks among the edges in the fusion frames for a given cross section of the tensor components as will be shown in this section. To demonstrate this, an inspection was done of the first and last tensors of a total of 20 iteration under S&P and Ransom Value types of impulsive noise. Figure 56 shows the tensor components of the recovered image computed at iterations 1 and 20 under a process with S&P noise, while Figure 57 plots the values of a row of the computed tensor component $\hat{g}_{22}$ of Figure 56. The same comparisons were done for a process with Random Value noise in Figure 58 and Figure 59. Unfortunately, the expected pronounced increment in the peaks around the transitions from the depth maps are not big from iteration 1 to iteration 20, it is almost the same. Although the width of the peak around the same areas are narrowed, which is the desired effect, it is hard to say that it was actually a significant improvement. Despite of this, the reduced increment in the tensor values and the narrow peaks generates the improvements shown in the overall process for the registration as presented in the previous section.

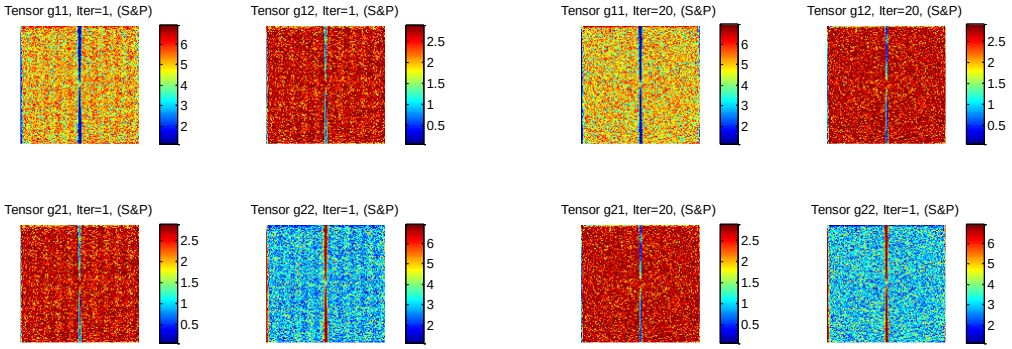


Figure 56: Tensor Components of Double slope synthetic image at iteration 1 & 20 under Salt & Pepper noise.

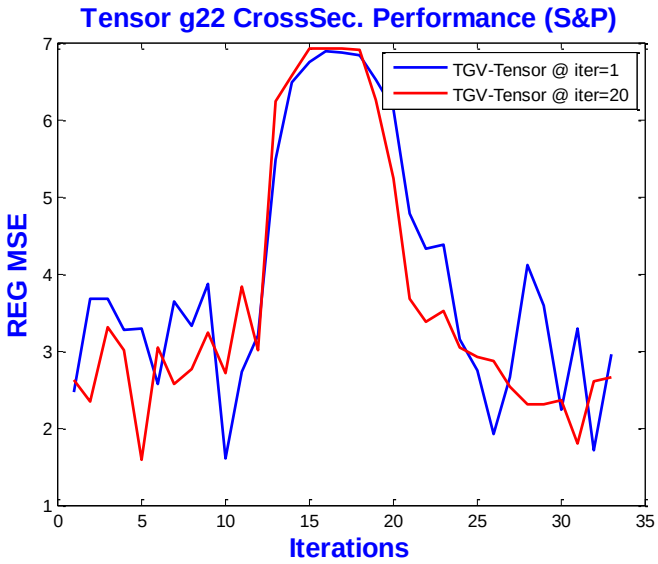


Figure 57: Double Slope Synthetic Image Tensor Cross section sample Comparison at different iterations under Salt and Pepper Noise.

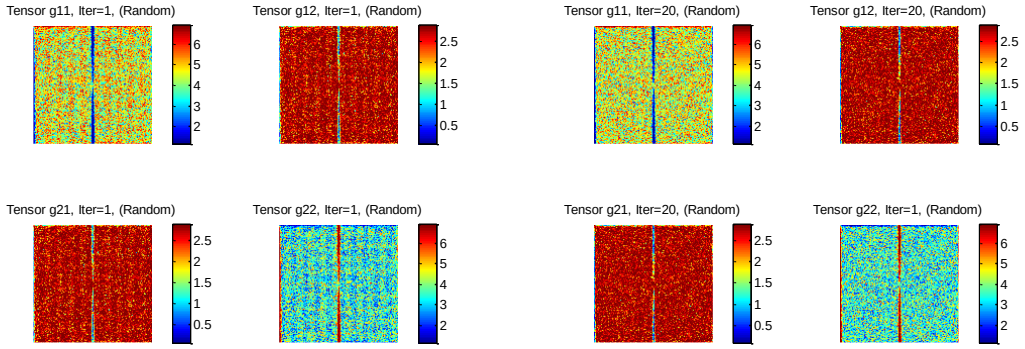


Figure 58: Tensor Components of Double slope synthetic image at iteration 1 & 20 under Random noise.

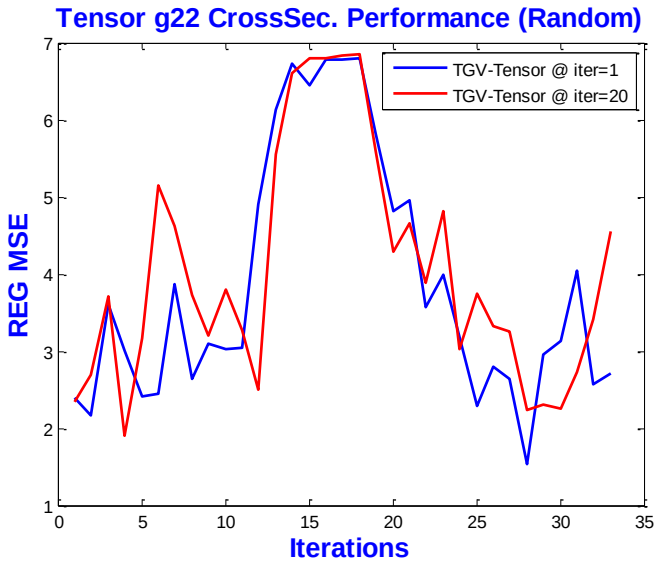


Figure 59: Double Slope Synthetic Image Tensor Cross section sample Comparison at different iterations under Salt and Pepper Noise.

6.6 Tensor Eigenvalue Strategy

This section presents a different scheme to make eigenvalue modifications for the structure tensor which is based on ideas from Weickert [53], who introduces variable eigenvalues depending on the nature of the image region. This kind of assignation is compared with the original scheme of fixed values as introduced in the previous sections.

The Weickert strategy assigns eigenvalues to a structure tensor depending on the coherence k of the original structure tensor eigenvalues at a given location, which is defined as follows for the specific case of 2 by 2 pixels matrices which have only two eigenvalues:

$$k = (\lambda_1 - \lambda_2)^2$$

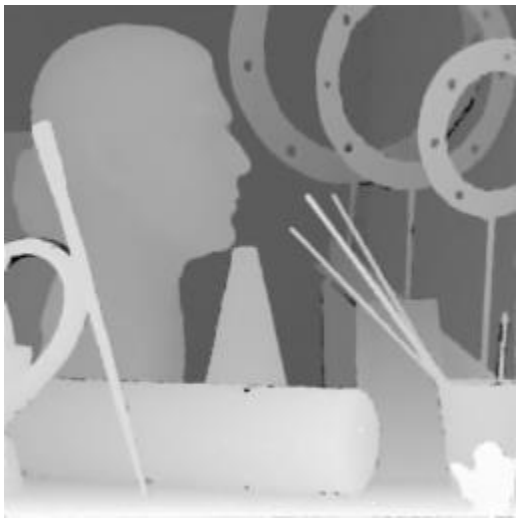


Figure 60: Fixed Eigenvalue Reconstructed Image.

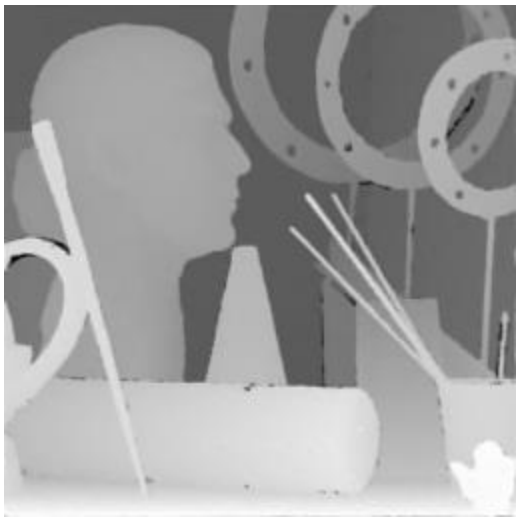


Figure 61: Variable Eigenvalue Reconstructed Image.

In contrast, the assignation for the proposed fixed eigenvalue modification framework is similar to the strategy in [53] and defines the modified eigenvalues as follows:

$$\lambda_1 = \alpha \cdot \sqrt{\sigma_0 \cdot \eta_0}$$

$$\lambda_2 = \begin{cases} \alpha \cdot \sqrt{\sigma_0 \cdot \eta_0} & if\ k \leq thr, \\ \alpha / \sqrt{\sigma_0 \cdot \eta_0} & else. \end{cases}$$

The performance for both cases can be appreciated in Table 10. Although both scheme reached high PSNR and SSIM, all the experiments for the validation of the algorithms were performed with the fixed eigenvalues scheme which showed slightly better performance evolution and final results as can be seen in Figure 62.

Table 10: ADMM TGV with Tensor Performance for different Eigenvalue configurations.

Eigenvalues	**T_iter**	**Tend**	**L_PSNR**	**L_SSIM**
Fixed	500	288.612	37.935	0.9863
Variable	495	370.307	37.571	0.9848

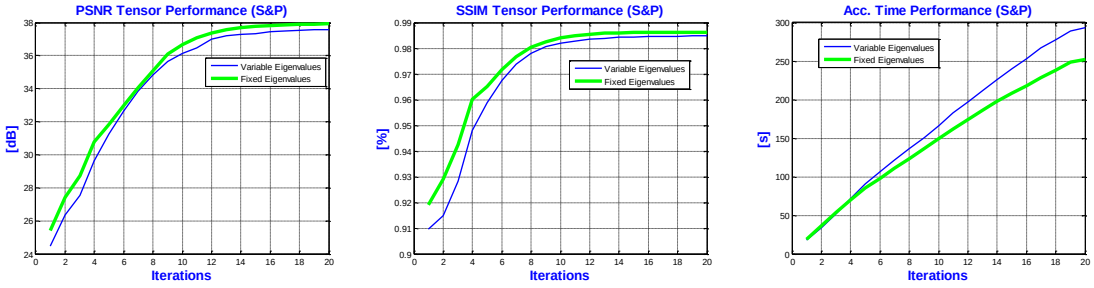


Figure 62: Tensor performance evolution comparison for different Eigenvalue configuration.

Chapter 7: Results

This chapter pretends to evaluate objectively and subjectively the proposed algorithm. To do this, a set of different experiments were performed in the following subsections to demonstrate the effectiveness of the employed numerical method using the three regularization terms: TV, TGV and TGV with Tensor. The objective metrics consider the PSNR and SSIM quality figures of merit as well as additional secondary results are included such as total processing time, evolution across iterations and the actual up-scaled resulting depth maps. The PSNR is defined as:

$$PSNR = 10\log_{10}\left(\frac{\left(Max(\hat{X})\right)^2}{MSE}\right) \tag{72}$$

Where:

$$MSE = \frac{1}{N \cdot N}\left(\left\|\hat{X} - X_{GT}\right\|^2\right) \tag{73}$$

$\hat{X}$ is the up-scaled and denoised image

X_{GT} is the ground truth image.

In addition to the PSNR, a second metric has been employed for the evaluation of the proposed framework, which considers image degradations as perceived changes in the structural information as proposed by Wang and Bovik in [54] rather than perceived visual errors to quantify image degradation. This metric assumes that human visual perception is highly adapted for extracting structural information, generating in this way a Structural Similarity (SSIM) Index. Thus, an image can be distorted in different ways and levels but it can have the same MSE,

however the SSIM is able to objectively assess the image quality in a model closer to the Human Vision System. In this way, the overall similarity figure is defined as:

$$S(x,y) = f\big(l(x,y), c(x,y), s(x,y)\big)$$

Where x and y are two images, $l(x,y)$, $c(x,y)$ and $s(x,y)$ are the luminance, contrast and structure comparisons respectively. Each term is defined trying to meet the conditions of symmetry ($S(x,y) = S(y,x)$), boundedness ($S(x,y) \leq 1$) and unique maximum ($S(x,y) = 1 \iff x = y$) as follows:

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \tag{74}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \tag{75}$$

$$s(x,y) = \frac{2\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \tag{76}$$

Where:

- $\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i$ and $\mu_y = \frac{1}{N}\sum_{i=1}^{N} y_i$ are the mean luminance of each image;
- $\sigma_x = \left(\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2\right)^{1/2}$ and $\sigma_y = \left(\frac{1}{N-1}\sum_{i=1}^{N}\big(y_i - \mu_y\big)^2\right)^{1/2}$ are the standard deviation of the luminance as an estimate of the contrast;
- $\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)\left(y_i - \mu_y\right)$ is the correlation; and
- C_1, C_2, and C_3 are constants to avoid instability in the case both where means and standard deviations are equals.

Then the SSIM index function that relates the three terms is:

$$SSIM(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma} \tag{77}$$

Where α, β, and γ are parameters that weight the contribution of each term.

By combining (74), (75) and (76) with (77) then it is obtained a final expression assuming $\alpha = \beta = \gamma = 1$ and $C_3 = C_2/2$:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

And for the purposes of this work:

$$SSIM(\hat{X}, X_{GT}) = \frac{(2\mu_{\hat{X}}\mu_{X_{GT}} + C_1)(2\sigma_{\hat{X}X_{GT}} + C_2)}{(\mu_{\hat{X}}^2 + \mu_{X_{GT}}^2 + C_1)(\sigma_{\hat{X}}^2 + \sigma_{X_{GT}}^2 + C_2)} \tag{78}$$

7.1 Super-Resolution Performance vs. Number of Frames

Two images from the Middlebury dataset were selected to make a characterization of the super-resolution performance for different number of frames. One of them has several areas with fine details (Art), while the other one has more simple shapes with gradual changes in depth (Bowling).

The common experimental conditions are: Impulsive Noise Density=10%; Gaussian Noise sigma = 3; Magnification factor = 4 (2 in each direction); low-resolution observations are 128x128 pixels and resulting up-scaled depth map is 256x256 pixels. The first evaluated depth map is Art. Figure 63 through Figure 66 show the results for this image.

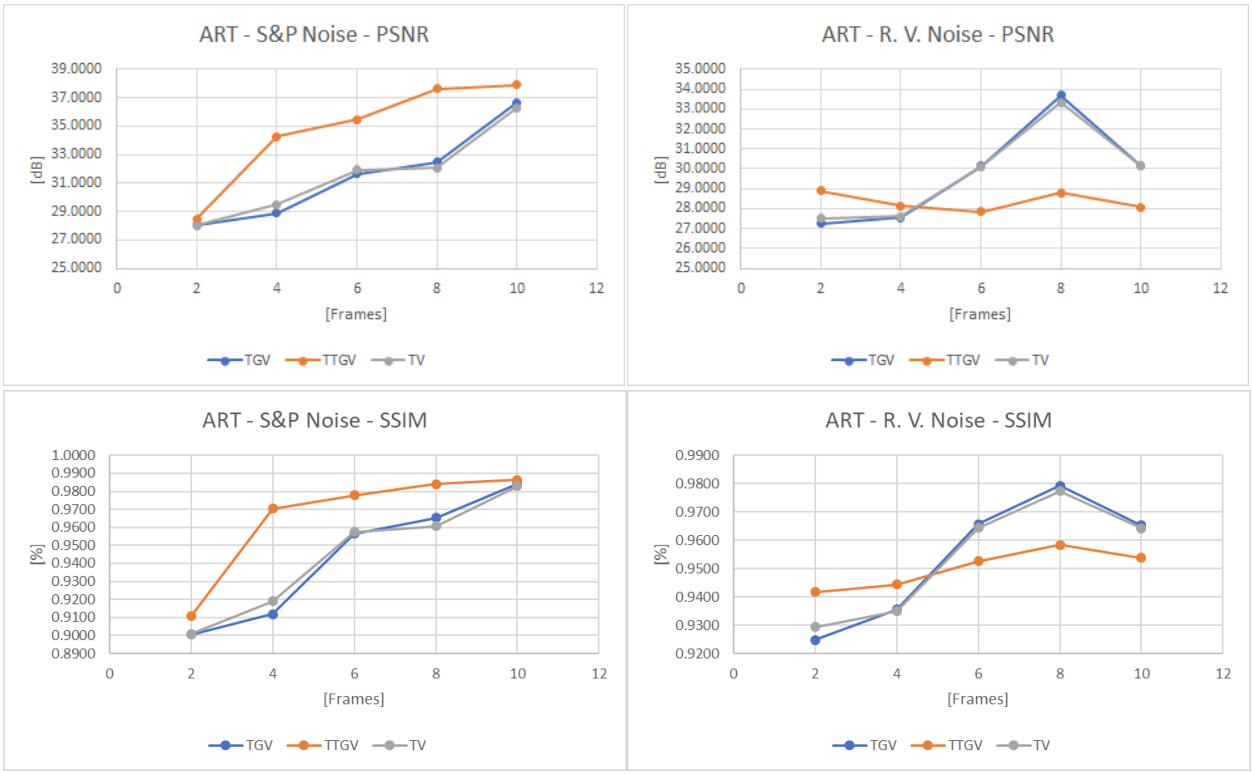


Figure 63: Up-scaling and Reconstruction Performance vs. Number of Frames - Middlebury Art Image.

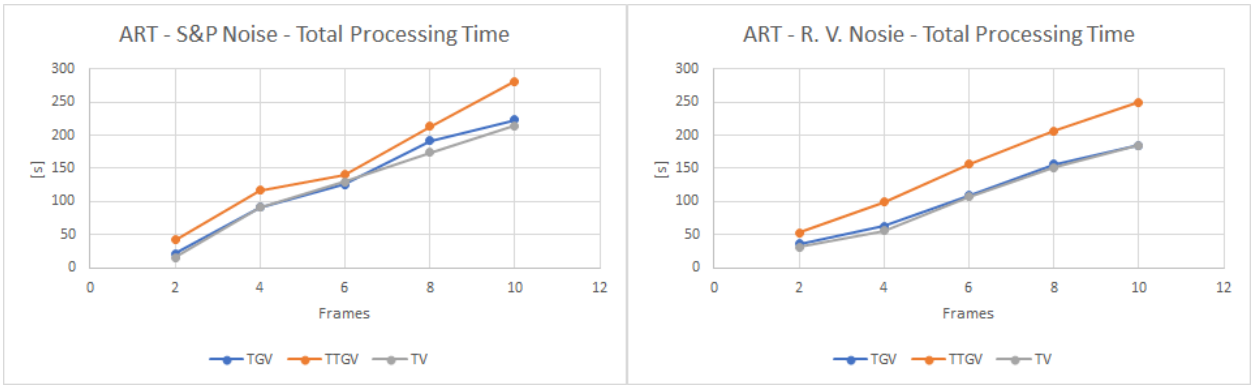


Figure 64: Total processing Time vs. Number of Frames – Middlebury Art Image.

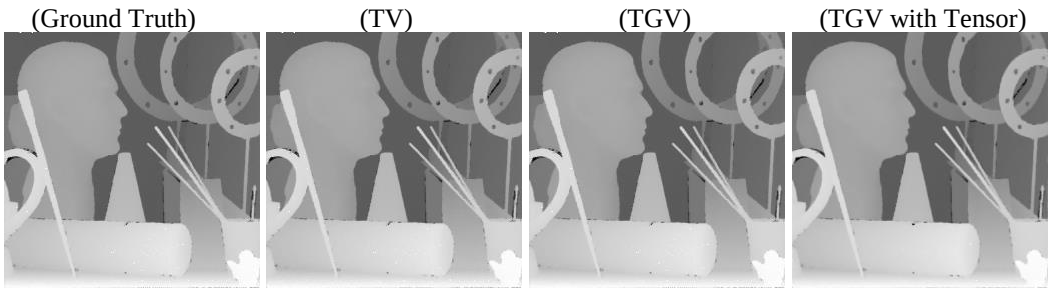


Figure 65: Art Performance with S&P Noise: 10 frames.

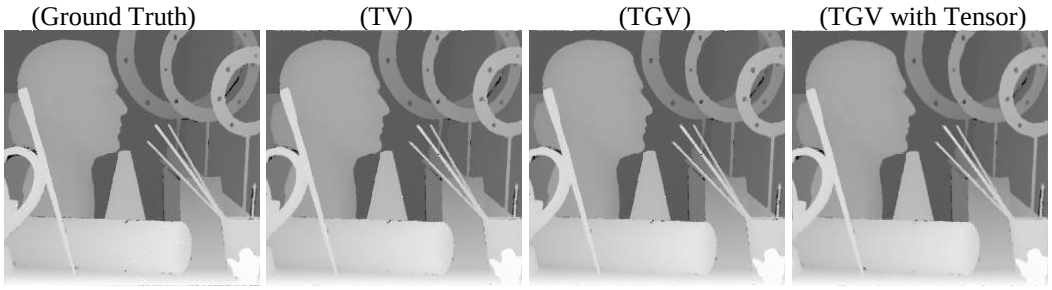


Figure 66: Art Performance with Random Value Noise: 8 frames.

Now the second simulated image also from the Middlebury dataset is the Bowling depth map. The following figures present the algorithm performance with this image.

Figure 67: Noise Performance vs. Number of Frames - Middlebury Bowling Image.

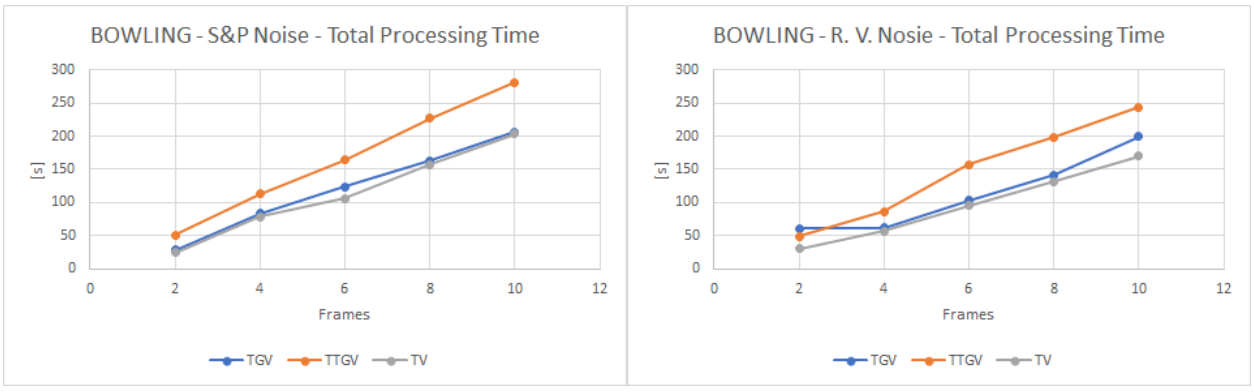


Figure 68: Total processing Time vs. Number of Frames – Middlebury Bowling Image.

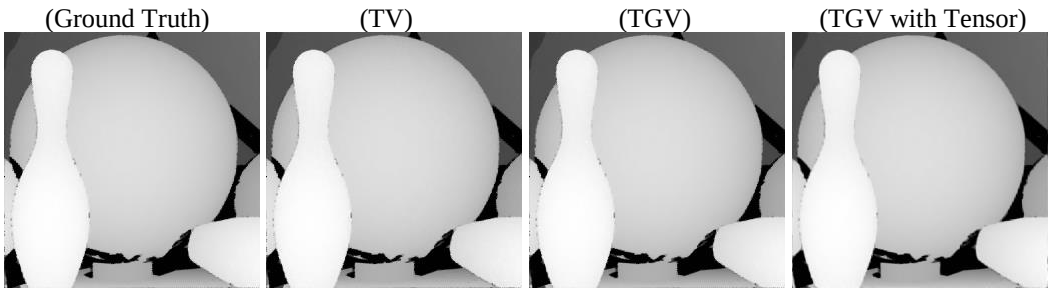


Figure 69: Bowling Performance with S&P Noise: 8 frames.

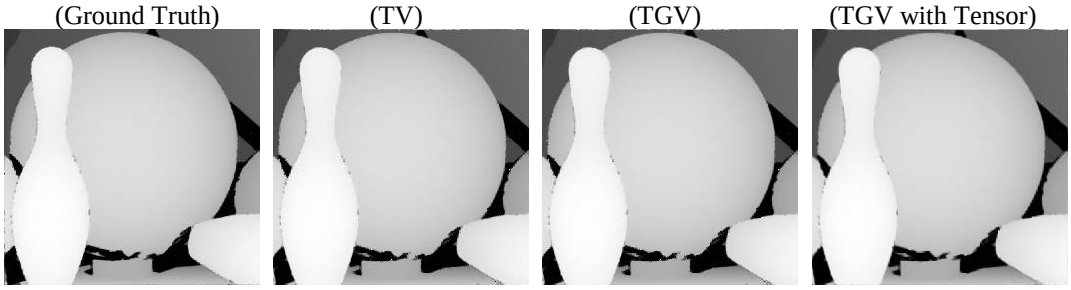


Figure 70: Bowling Performance with Random Value Noise: 6 frames.

Finally, the next plots compare the PSNR and SSIM evolution with algorithm iteration for the selected cases from the above experiments in a similar fashion as the registration process was previously assessed. This time the accumulative execution time is included so that the advantage of the faster method can be analyzed and a fair comparison can be established.

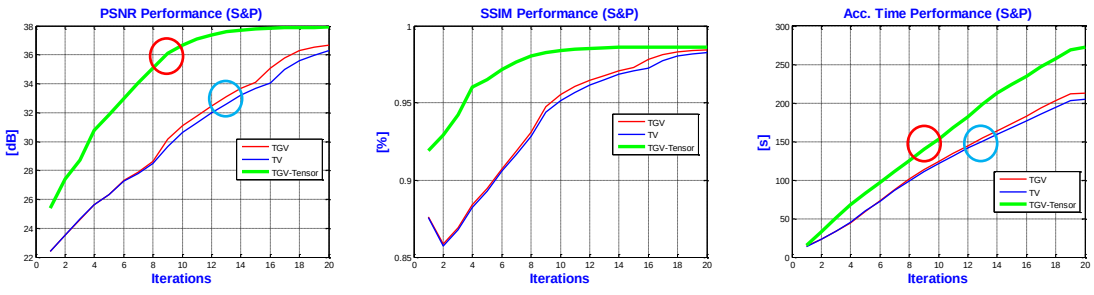


Figure 71: Art Performance Evolution with S&P Noise: 10 frames.

From Figure 71 it can be observed that the TGV with tensor reaches in about 9 iterations a PSNR of 36 dB, which took a total processing time of less than 150 seconds denoted by the red circles. On the other hand, the other regularization terms TV and TGV reaches iteration 13 after 150 seconds, developing a PSNR only around 33 dB by the same time, denoted by the blue circles. Unfortunately, this kind of fast reconstruction behavior of the TGV with Tensor is present only with the Salt & Pepper noise. It can be seen in Figure 72 that the trend is even contrary after iteration 4 for Random Value noise. Figure 73 and Figure 74 show the results for the Bowling image.

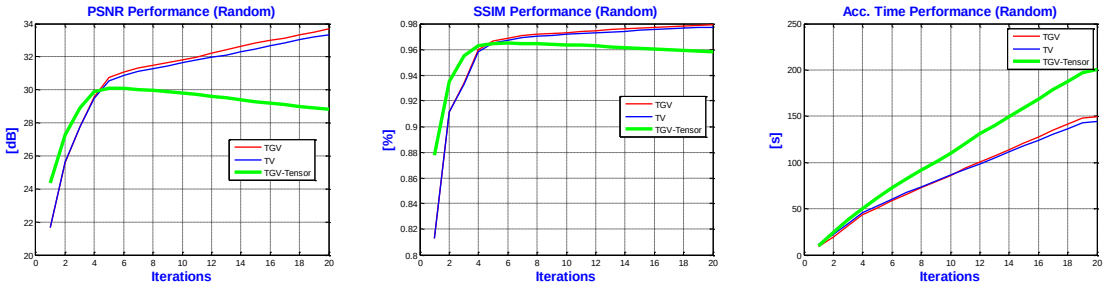


Figure 72: Art Performance Evolution with Random Value Noise: 8 frames.

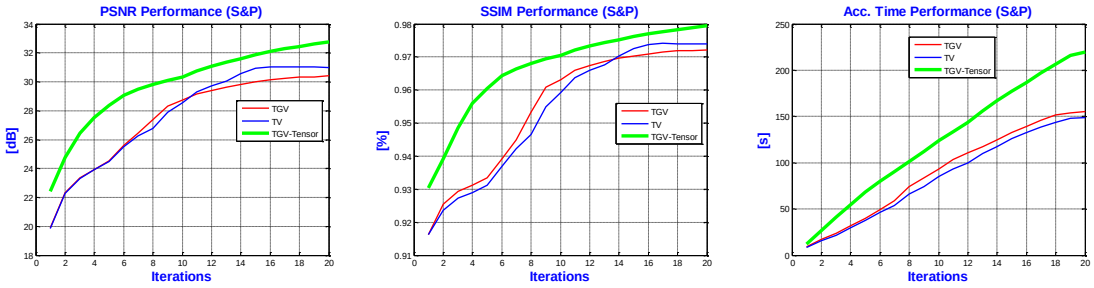


Figure 73: Bowling Performance Evolution with S&P Noise: 8 frames.

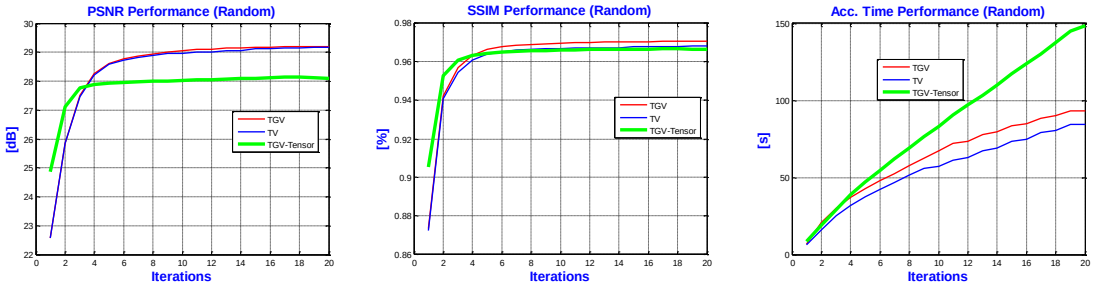


Figure 74: Bowling Performance Evolution with Random Value Noise: 6 frames.

7.2 Noise Susceptibility Impact

The set of experiments of this section tries to characterize the behavior of the proposed framework under several scenarios with different levels of impulsive noise: Salt and Pepper and Random Value. The noise density is varied from 5% of corrupted pixels up to 30% in steps of 5%. This time only one depth map was selected for the simulated low-resolution observations, it was the Art depth map, which has several areas with fine details. Figure 75 and Figure 76 show the evolution of these experiments.

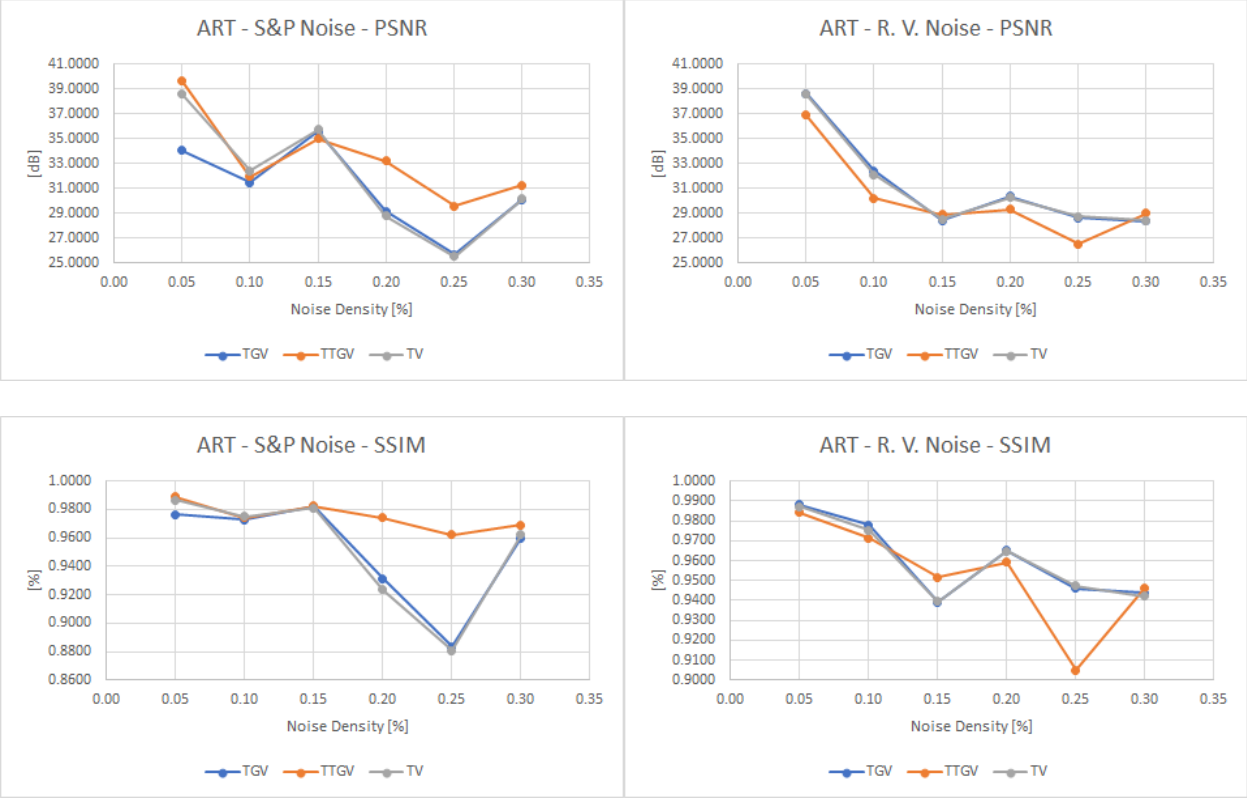


Figure 75: Up-scaling and Reconstruction Performance vs. Noise Density - Middlebury Art Image.

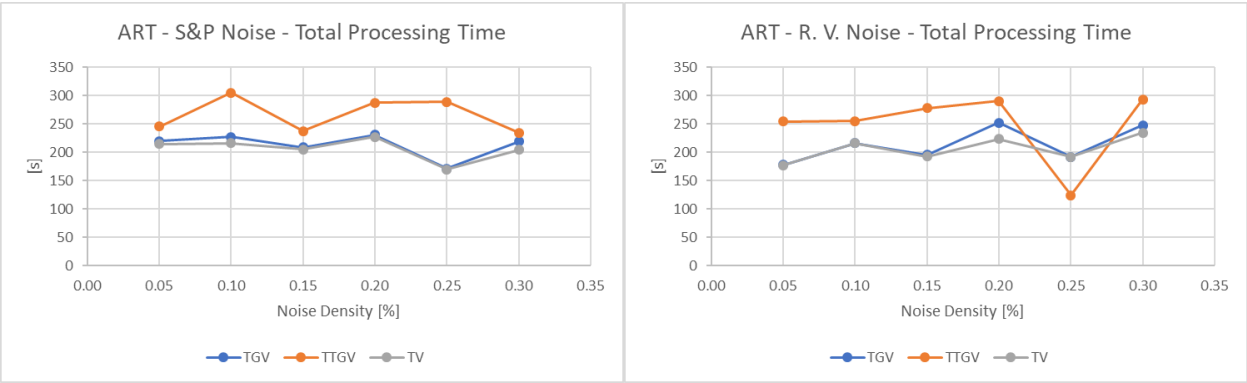


Figure 76: Total processing Time vs. Noise Density – Middlebury Art Image.

Figure 77, Figure 78 and Figure 79 present the final image results of the three regularization terms under Salt and Pepper noise while Figure 80, Figure 81 and Figure 82 present the respective image results under Random Value noise.

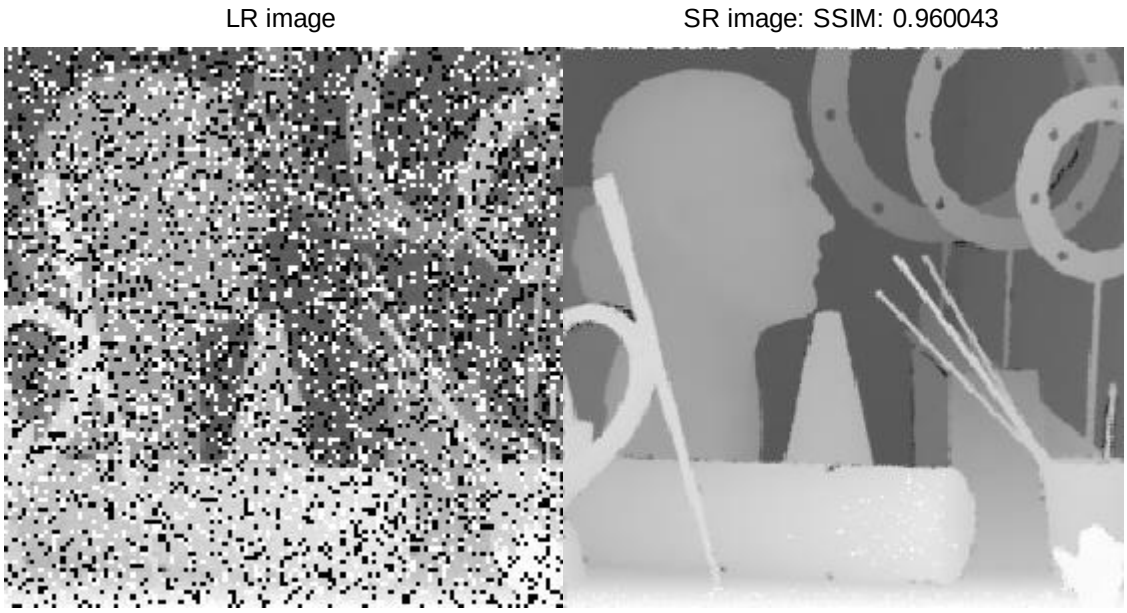


Figure 77: TGV Performance with 30% of pixels with S&P.

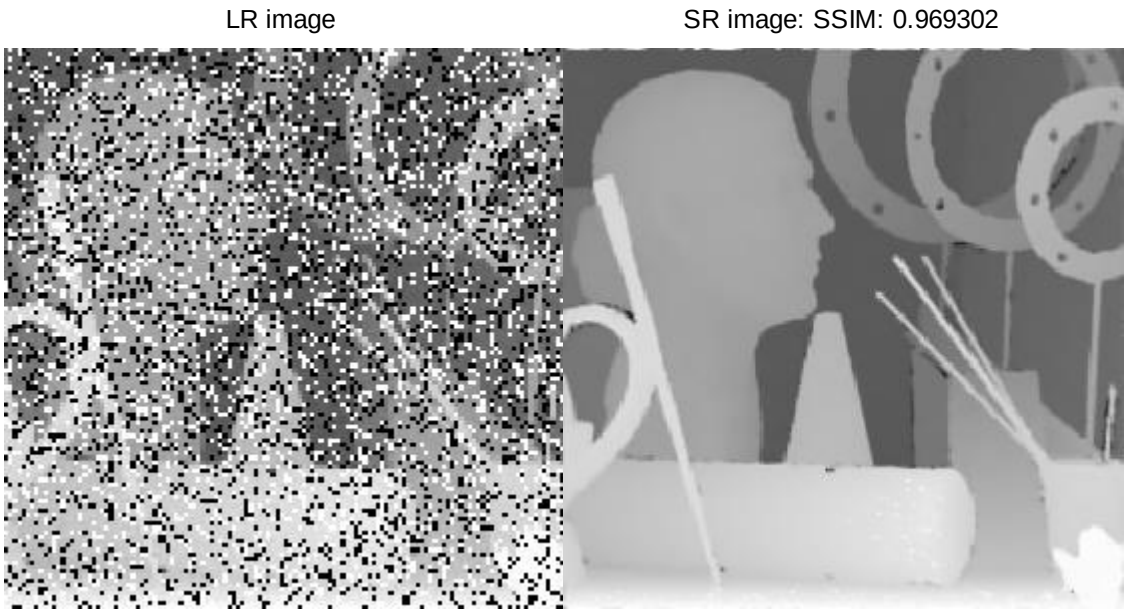


Figure 78: TGV with Tensor Performance with 30% of pixels with S&P.

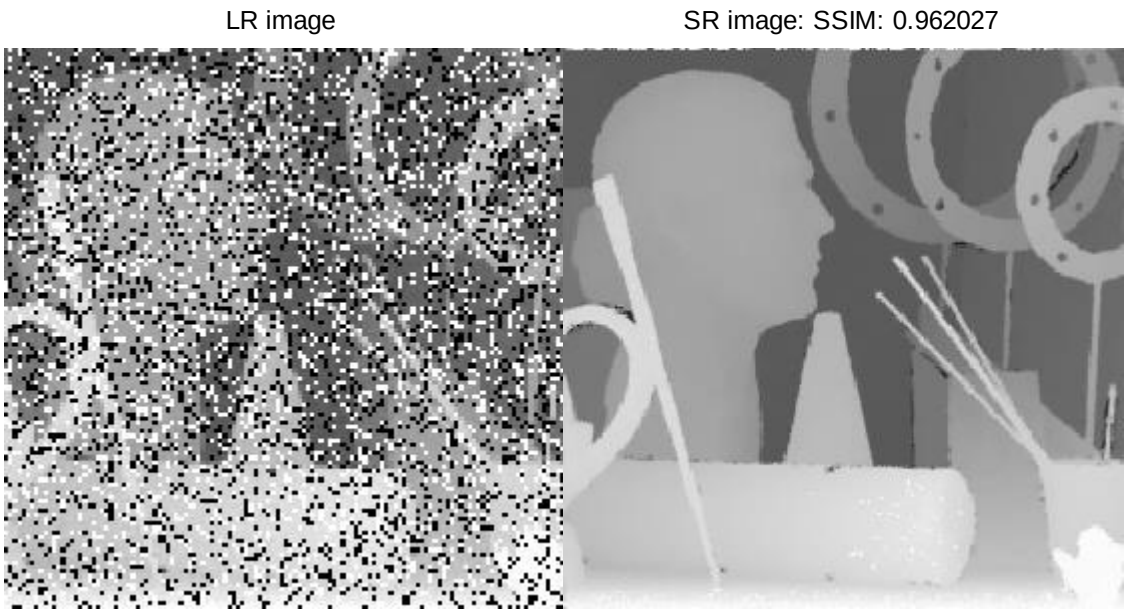


Figure 79: TV with Tensor Performance with 30% of pixels with S&P.

Now the results for Random Value (RV) impulsive noise experiments are shown:

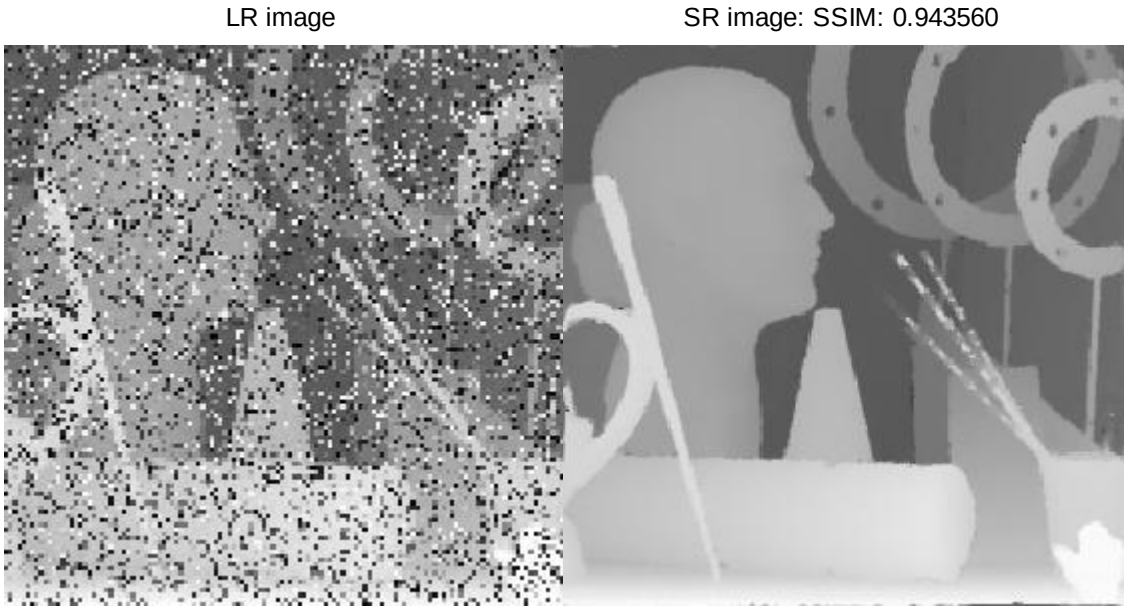


Figure 80: TGV Performance with 30% of pixels with RV.

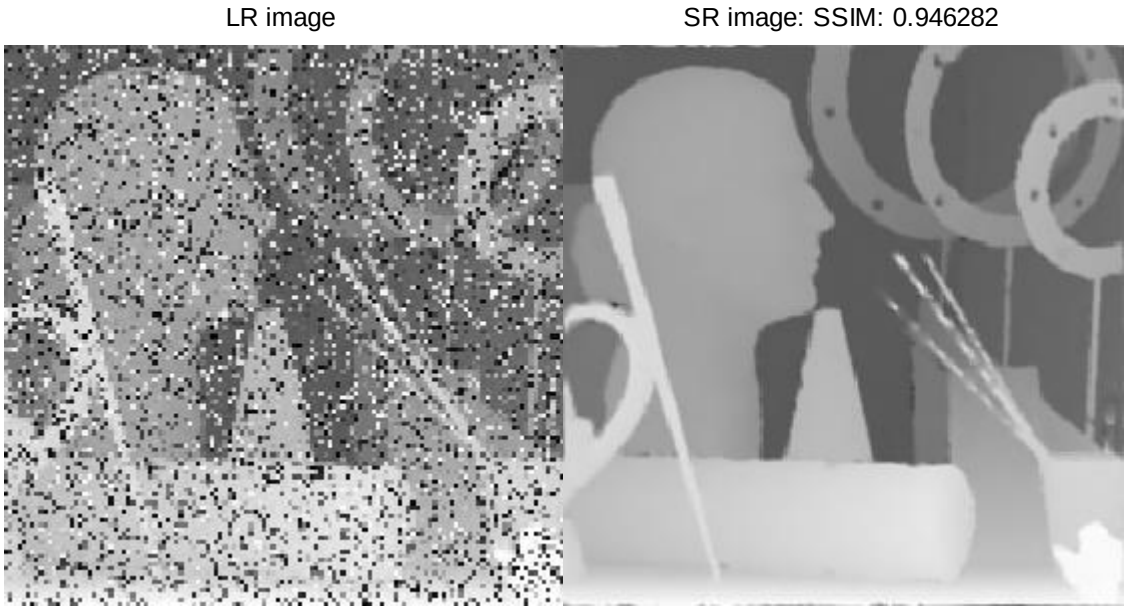


Figure 81: TGV with Tensor Performance with 30% of pixels with RV.

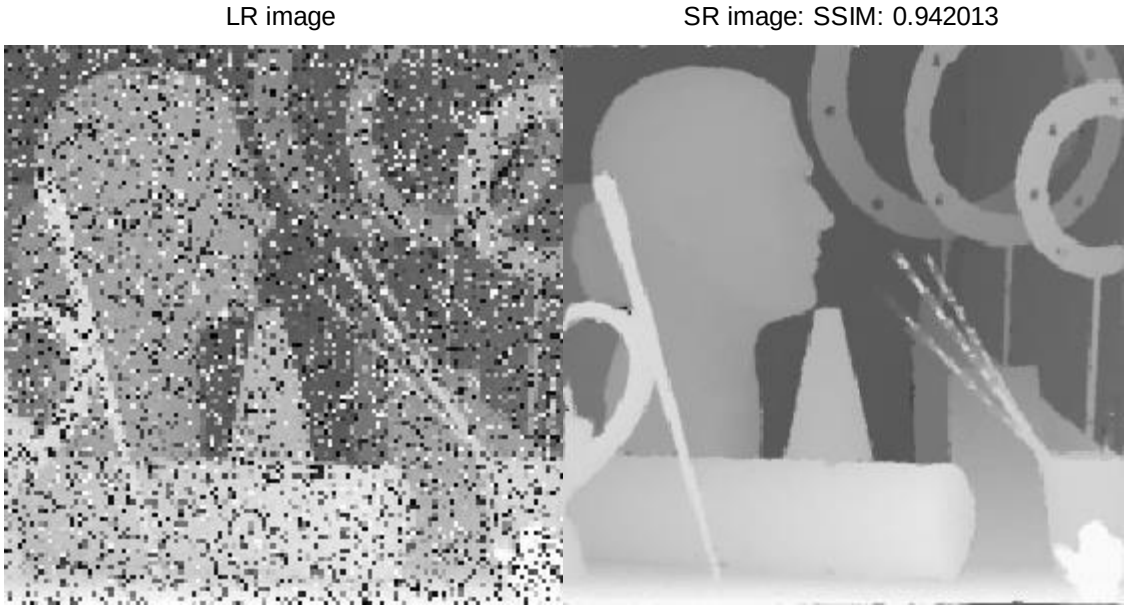


Figure 82: TV with Tensor Performance with 30% of pixels with RV.

The set of experiments performed during this research are in the order of hundreds considering that each configuration needs to be tested for 3 different methods, TV, TGV and TTGV as well as two kinds of impulsive noise and with pure Gaussian noise. The high number of

tests makes it complicated to regard a statistically minimum number of samples for every configuration. The following set of experiments is reproducing the scenario of Figure 75 only for the case of Random Value. The intention is to show that the variability caused by the noise realizations is not great as the previous experiments. Each data point for the cases of 5%, 15% and 25% is the mean of ten realizations of the experiment for the given configurations. A total of 90 experiments were performed.

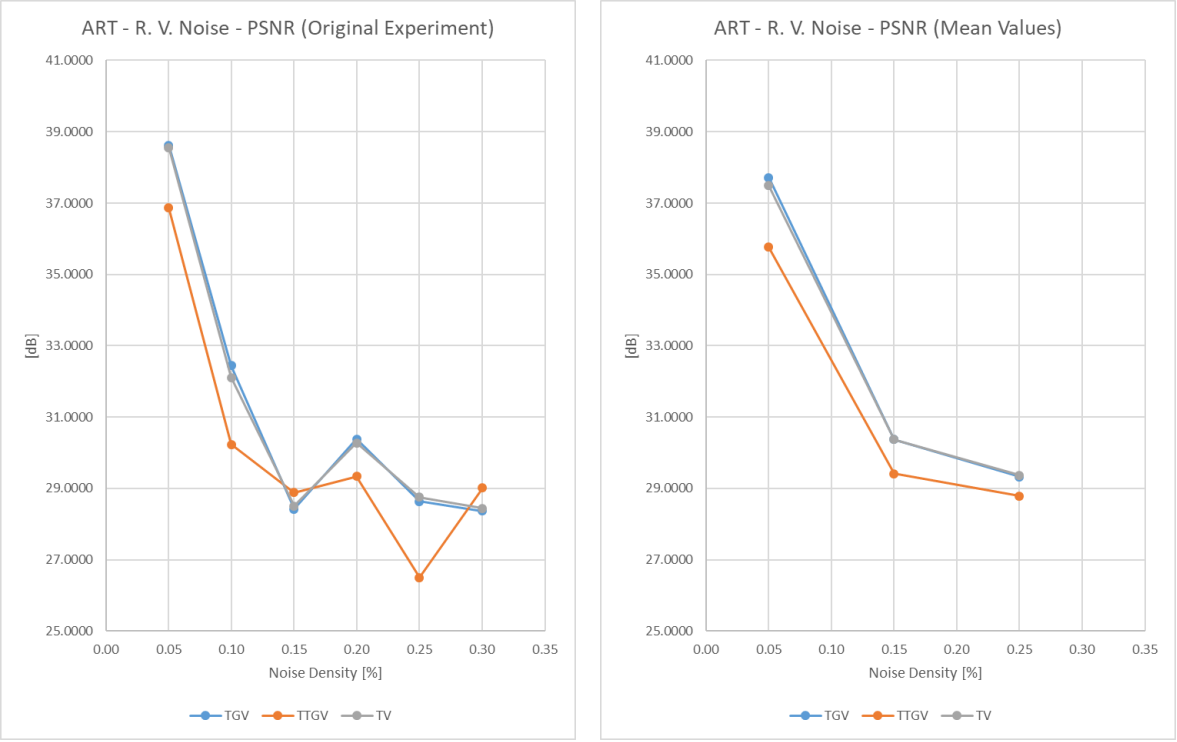


Figure 83: PSNR Comparison of a single realization experiment set vs. Mean values of a 10-realizations experiment set.

On the one hand, the PSNR comparison shown in Figure 83 confirms the tendency of the 3 methods to have a smaller PSNR value for higher noise densities. On the other hand, Figure 84 shows in a clearer way the tendency of the SSIM performance and contrary to the previous results,

the TTGV produces better results after 15% of noise density. The case of the 25% of the original value looks like an outlier that could be generating a bad registration which leads to a low-quality reconstruction and an early stop of the iterative process as the total consumed time reveals in the accumulative time plots of Figure 85 for the original experiment.

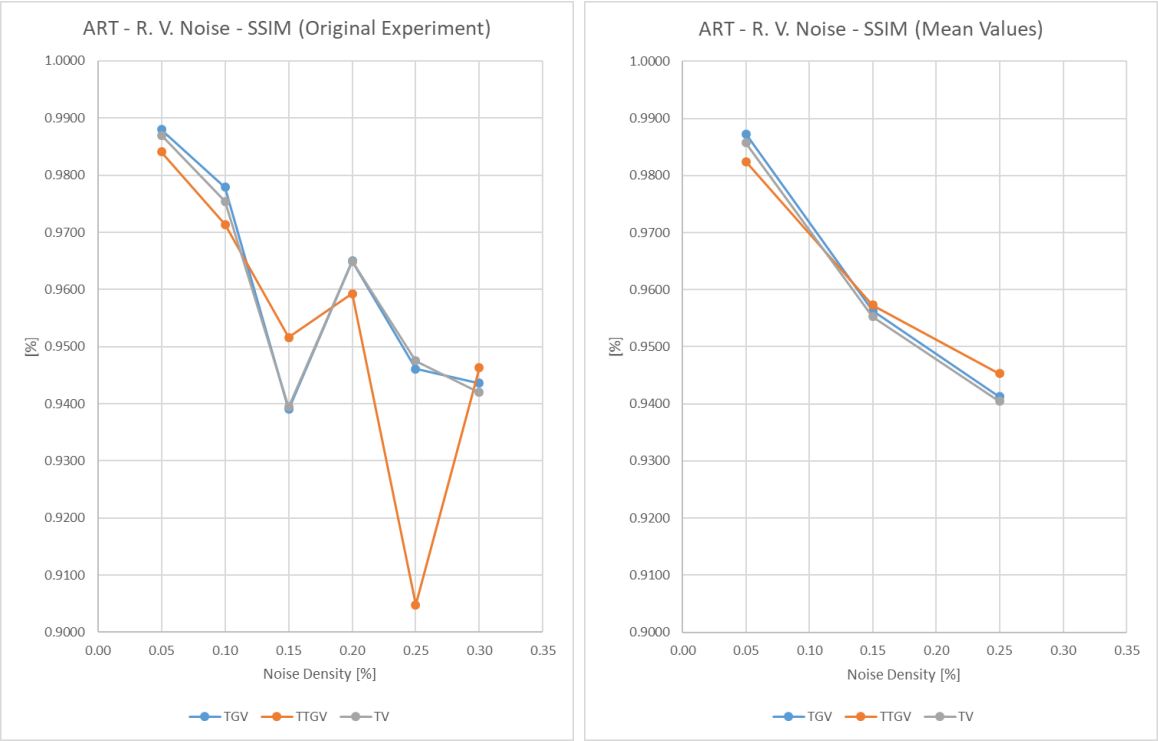


Figure 84: SSIM Comparison of a single realization experiment set vs. Mean values of a 10-realizations experiment set.

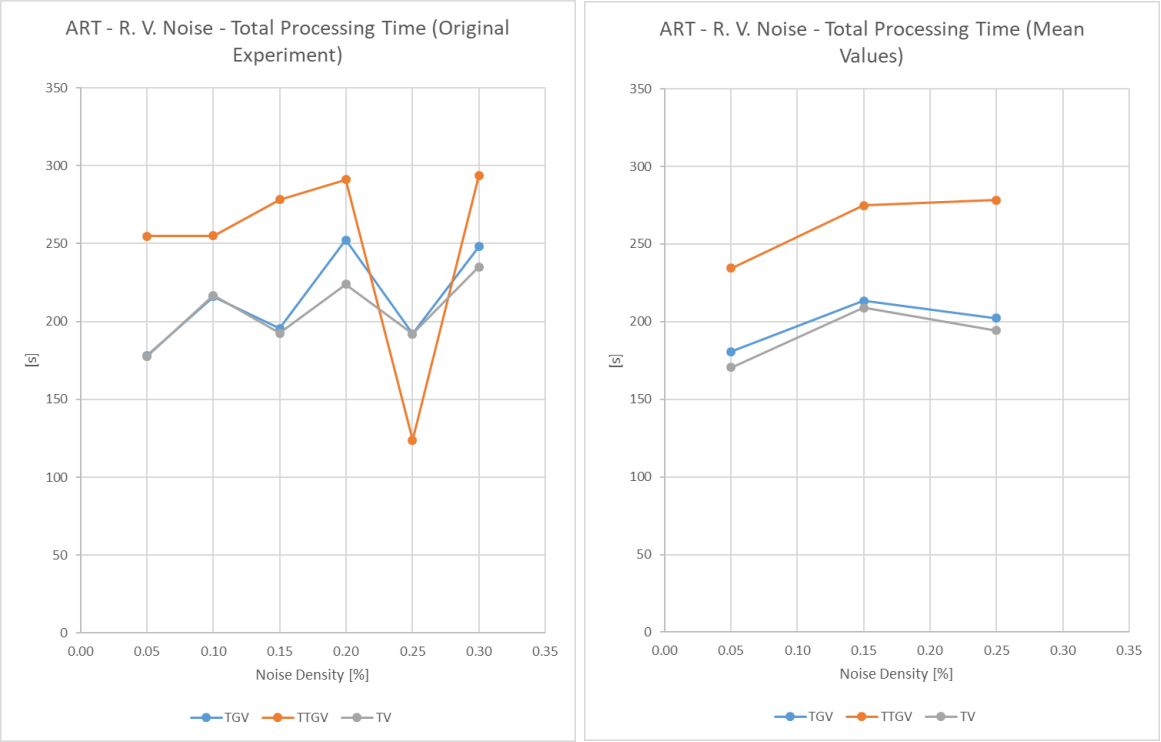


Figure 85: SSIM Comparison of a single realization experiment set vs. Mean values of a 10-realizations experiment set.

Additionally to the presented behavior of the mean values, Table 11, Table 12 and Table 13 introduce the standard deviation of each experiment for the three different noise densities tested to provide an idea of the variability of the experiments and the dispersion in the performance results for the PSNR and the SSIM. As it can be seen, the maximum standard deviation was registered in the case of the ADMM-TV (5% R.V case) for the PSNR with 2.2 dB, while the maximum for the SSIM was in the ADMM-TV (25% R.V case) with 0.03 of normalized percent.

Table 11: 10-realizations Experiment Statistics for 5% R.V. Impulsive Noise.

Case:	**5% R.V. Impulsive Noise**					
	ADMM-TV		**ADMM-TGV**		**ADMM-TTGV**	
Statistics	**PSNR**	**SIMM**	**PSNR**	**SIMM**	**PSNR**	**SIMM**
	[dB]	[%]	[dB]	[%]	[dB]	[%]
MIN	33.5175	0.9798	35.3817	0.9842	30.3922	0.9724
MEAN	37.4880	0.9857	37.7176	0.9873	35.7602	0.9825
MAX	39.5201	0.9883	39.7074	0.9895	37.8730	0.9857
STD	2.2011	0.0030	1.8256	0.0019	2.4428	0.0040

Table 12: 10-realizations Experiment Statistics for 15% R.V. Impulsive Noise.

Case:	**15% R.V. Impulsive Noise**					
	ADMM-TV		**ADMM-TGV**		**ADMM-TTGV**	
Statistics	**PSNR**	**SIMM**	**PSNR**	**SIMM**	**PSNR**	**SIMM**
	[dB]	[%]	[dB]	[%]	[dB]	[%]
MIN	27.2366	0.9215	27.3304	0.9245	26.9528	0.9385
MEAN	30.3775	0.9553	30.3694	0.9563	29.4150	0.9573
MAX	33.9114	0.9758	33.9485	0.9768	31.8365	0.9716
STD	2.0108	0.0181	1.9815	0.0170	1.5763	0.0111

Table 13: 10-realizations Experiment Statistics for 25% R.V. Impulsive Noise.

Case:	**25% R.V. Impulsive Noise**					
	ADMM-TV		**ADMM-TGV**		**ADMM-TTGV**	
Statistics	**PSNR**	**SIMM**	**PSNR**	**SIMM**	**PSNR**	**SIMM**
	[dB]	[%]	[dB]	[%]	[dB]	[%]
MIN	25.6013	0.8601	25.6184	0.8601	27.3753	0.9173
MEAN	29.3680	0.9405	29.3280	0.9413	28.7932	0.9453
MAX	31.6972	0.9656	31.5761	0.9663	30.7563	0.9624
STD	1.7503	0.0301	1.7279	0.0307	1.0668	0.0131

7.3 Overall Performance over multiple images

An overall evaluation is demonstrated with several simulated depth frame sequences for multiple combinations of Gaussian noise contamination levels and impulsive noise densities and types. The following two figures show ground truth images and the subsequent four tables report the algorithm performance for these images respectively. The process performed a magnification factor of 2 in each direction.

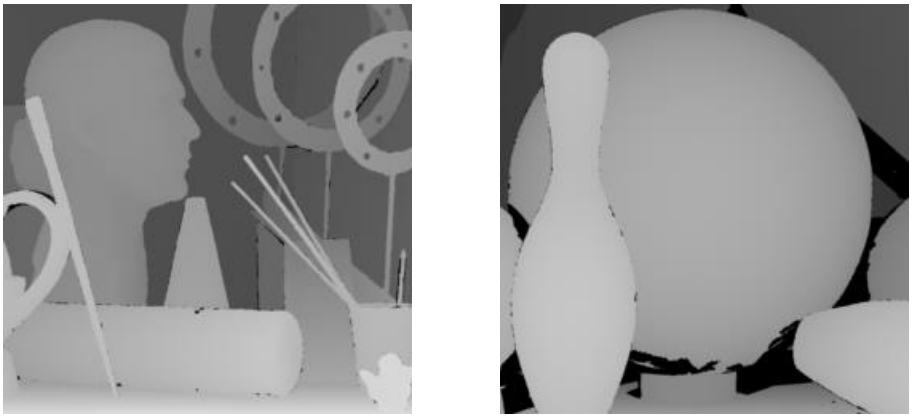


Figure 86: Art and Bowling Images (256x256) Ground Truth.

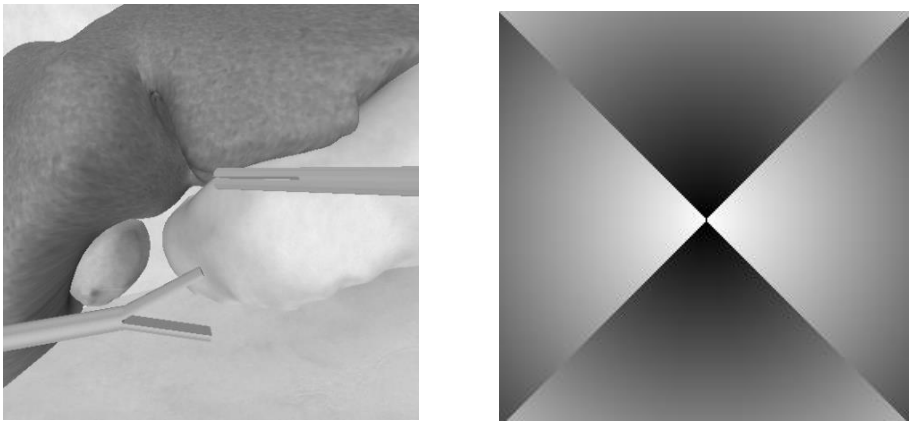


Figure 87: Endoscope(512x512) **[23]** and BWH image (256x256) **[55]** Ground Truth.

Table 14: Middlebury Performance Images for Salt and Pepper Impulsive Noise.

Noise		Method	Art				Bowling			
Sigma	S&P Den		T_iter	Tend	L_PSNR	L_SSIM	T_iter	Tend	L_PSNR	L_SSIM
0	0.10	TV	500	213.353	37.1305	0.9881	470	197.087	30.7972	0.9816
		TGV	500	220.798	37.0085	0.9869	500	202.423	31.2911	0.9823
		TTGV	500	240.971	38.1581	0.9881	500	247.28	30.0169	0.9806
5	0.10	TV	500	212.257	30.5839	0.9529	473	192.928	29.2347	0.9485
		TGV	500	220.339	30.6175	0.9537	500	202.812	29.2677	0.9512
		TTGV	500	246.376	32.404	0.9695	500	238.373	27.4318	0.9551
10	0.10	TV	500	214.764	31.3394	0.7852	500	201.851	29.1854	0.7696
		TGV	500	211.795	31.21	0.7851	500	201.091	29.1467	0.7678
		TTGV	500	239.631	34.656	0.893	500	248.998	31.4081	0.8749
5	0.01	TV	313	152.728	40.5506	0.9742	410	176.161	40.8671	0.9682
		TGV	358	160.16	40.6539	0.9752	419	173.678	41.0394	0.9705
		TTGV	425	209.527	40.917	0.9845	484	227.242	40.4333	0.9808
5	0.15	TV	500	188.682	26.0646	0.8762	469	198.933	27.8189	0.9455
		TGV	500	198.449	26.3429	0.8791	500	209.596	28.1351	0.9501
		TTGV	493	217.581	27.1353	0.9138	500	247.94	27.951	0.959
5	0.30	TV	383	188.547	29.6559	0.9414	439	165.469	25.166	0.9196
		TGV	469	208.67	29.9956	0.9465	396	163.992	25.0833	0.9218
		TTGV	490	224.601	30.5486	0.9592	425	199.094	25.2005	0.9334

Table 15 Middlebury Performance Images for Random Value Impulsive Noise.

Noise		Method	Art				Bowling			
Sigma	R.V. Den		T_iter	Tend	L_PSNR	L_SSIM	T_iter	Tend	L_PSNR	L_SSIM
0	0.10	TV	451	212.761	32.7921	0.9722	403	219.591	30.6203	0.9796
		TGV	500	222.001	32.0706	0.9657	444	234.391	31.2072	0.9804
		TTGV	500	252.662	33.2707	0.9769	500	312.285	30.6795	0.979
5	0.10	TV	500	223.946	32.3168	0.9551	421	199.789	35.8195	0.9645
		TGV	500	227.856	32.9582	0.9592	447	210.98	35.8439	0.9667
		TTGV	500	250.376	32.0021	0.9672	498	257.242	33.5686	0.9744
10	0.10	TV	500	163.806	25.5053	0.7099	500	238.493	29.4772	0.7693
		TGV	500	171.488	25.7804	0.7182	500	242.08	29.3281	0.7688
		TTGV	500	252.295	29.4469	0.8744	500	296.978	30.2934	0.8769
5	0.01	TV	331	139.913	40.196	0.976	384	156.509	41.6377	0.9687
		TGV	383	148.82	40.445	0.9775	413	171.473	41.9189	0.9709
		TTGV	473	203.288	40.5308	0.9844	469	236.013	40.5499	0.9815
5	0.15	TV	374	140.705	34.5009	0.9654	410	237.116	29.4994	0.9484
		TGV	418	147.259	34.5695	0.9668	419	233.201	29.5691	0.9508
		TTGV	500	219.204	31.1974	0.9531	500	287.602	28.1938	0.9584
5	0.30	TV	500	228.22	29.793	0.9432	473	219.093	27.1373	0.933
		TGV	500	230.796	29.8418	0.9448	493	202.714	26.759	0.9334
		TTGV	500	248.538	29.0838	0.9397	500	302.469	27.3848	0.9482

Table 16: Endoscope and BWH Performance Images for Salt and Pepper Impulsive Noise.

Noise		Method	Endoscope				BWH			
Sigma	S&P Den		T_iter	Tend	L_PSNR	L_SSIM	T_iter	Tend	L_PSNR	L_SSIM
0	0.10	TV	451	694.13	28.6288	0.9445	421	185.951	30.4091	0.9755
		TGV	500	755.36	28.6316	0.9432	447	188.004	30.48	0.976
		TTGV	500	1109.24	32.9446	0.9757	495	251.732	33.1003	0.9854
5	0.10	TV	449	978.00	39.1669	0.9805	500	208.692	28.9744	0.9385
		TGV	467	962.40	39.3793	0.9809	500	212.642	29.7489	0.9497
		TTGV	375	897.71	38.3095	0.9825	500	243.493	32.2427	0.974
10	0.10	TV	448	728.08	27.7371	0.8612	500	191.962	28.928	0.7273
		TGV	476	778.82	27.7146	0.8615	500	195.733	29.0552	0.7287
		TTGV	500	888.76	29.2655	0.8909	500	276.032	32.5625	0.8683
5	0.01	TV	306	690.62	31.9001	0.9672	379	179.453	31.934	0.95
		TGV	370	738.91	32.0006	0.9681	415	182.777	32.1649	0.9574
		TTGV	500	1114.43	40.5385	0.9845	500	270.086	35.4196	0.979
5	0.15	TV	433	920.00	34.3854	0.9726	500	204.397	29.1122	0.9314
		TGV	471	950.25	34.4438	0.9732	500	191.483	28.6441	0.9323
		TTGV	500	1027.14	34.9447	0.9762	498	270.495	30.3038	0.9652
5	0.30	TV	352	702.64	37.1555	0.9745	322	163.349	27.9378	0.931
		TGV	415	762.13	37.2428	0.9749	500	220.287	28.0347	0.9438
		TTGV	500	925.05	35.1067	0.9742	500	271.046	30.0653	0.9635

Table 17: Endoscope and BWH Performance Images for Random Value Impulsive Noise.

Noise		Method	Endoscope				BWH			
Sigma	R.V. Den		T_iter	Tend	L_PSNR	L_SSIM	T_iter	Tend	L_PSNR	L_SSIM
0	0.10	TV	500	974.17	31.7959	0.961	475	205.944	33.0063	0.9865
		TGV	500	1051.89	32.3028	0.9676	476	210.434	33.8328	0.9875
		TTGV	494	1166.80	33.7842	0.9798	499	263.849	31.5267	0.9798
5	0.10	TV	496	1000.50	34.8171	0.9731	411	186.635	30.7767	0.9537
		TGV	499	1048.44	34.5673	0.9725	421	193.923	30.5419	0.9608
		TTGV	500	1125.16	32.6102	0.967	500	246.777	30.0037	0.969
10	0.10	TV	500	826.83	28.9712	0.8751	447	158.894	27.8456	0.7008
		TGV	500	857.32	28.9607	0.8754	447	155.675	27.8489	0.7008
		TTGV	453	962.96	30.0477	0.901	493	242.474	30.3909	0.86
5	0.01	TV	426	856.15	39.4616	0.9817	359	157.424	32.8463	0.9589
		TGV	500	951.79	40.5108	0.9825	422	175.994	33.1411	0.9665
		TTGV	500	1151.42	39.4069	0.984	500	256.233	34.1197	0.9797
5	0.15	TV	489	1000.93	34.0243	0.9696	474	201.831	29.2023	0.9473
		TGV	470	997.58	34.336	0.9707	500	209.306	29.2533	0.9556
		TTGV	500	1172.52	32.0012	0.9611	500	292.553	29.2402	0.9628
5	0.30	TV	443	865.09	31.8887	0.9464	429	206.709	30.5897	0.9542
		TGV	409	879.10	31.686	0.9444	427	207.359	31.0847	0.9648
		TTGV	500	1042.59	31.2835	0.9439	500	264.287	30.2868	0.968

7.4 Evaluation with Real ToF Depth frames

This section displays the subjective evaluation of the proposed framework with ToF frames, which are one of the main motivations of this research. Since these ToF depth map sequences are not synthetic images nor high resolution depth maps generated by stereoscopic or structured light, ground truth frames do not exist.

The first part of this section evaluates three frame sequences from the Corbs dataset available in [22]. The second part uses sequences of frames acquired by the Intel ToF camera as part of our own UTEP dataset.

7.4.1 Corbs Depth Maps Dataset Evaluation – Moving Camera

This section includes the three selected sequences of the Corbs dataset. A representative RGB frame of each sequence is displayed in Figure 88. Due to practical and experimental reasons, only a subsection the original depth maps were extracted for evaluation to feed frames of 256 by 256 pixels into the algorithms. A detailed description of the data set as well as infrared frames, time stamps and real-world coordinates of the acquisition equipment at the capture shoot are available at [22]. This section is limited to provide a rough idea of the amount of displacement between frames by showing frame 1, 5 and 10 out of a total of 10 frames used for the experiments included in this section.

a) b) c)

Figure 88: RGB frames of sequences: a) Race Car (R01); b) Desk (D01) and c) Electric Central Control (E01).

Figure 89 shows the statistics of the missing pixels that each ToF frame sequence presents for diverse reasons as previously discussed such as reflectivity issues, low IR amplitude, etc. These provide a good reference to know the amount of inpainting of pixels that the framework will process.

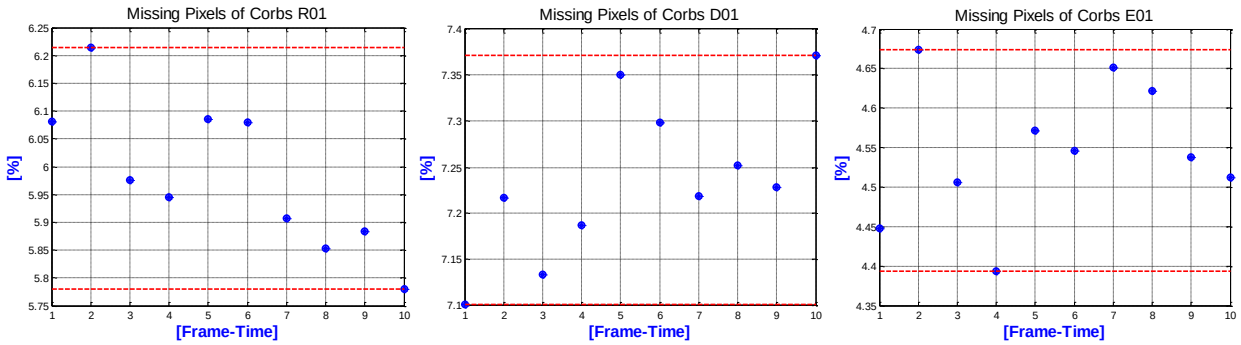


Figure 89: Missing pixel statistics of Corbs sequences R01, D01 and E01.

The frames were acquired on the order of 30 frames per second (fps) as the time stamps reveal for the extracted information from the ground truth trajectory of the optical center of the RGB camera provided by the sensors mounted in the system, which is displayed in Table 18. As can be seen later in the results for these sequences, this is an adequate speed for the camera regarding the distance of the objects in the scene. Figure 90 to Figure 97 show the detailed results.

Table 18: Example of Ground Truth Position trajectory of 10 Frames from D01 Sequence Raw Data.

Frame	**Time**	**dt**	**Rate**	**x**	**dx**	**y**	**dy**	**z**	**dz**	**Distance**	**Speed**
[#]	[ms]	[ms]	[F/s]	[m]	[mm]	[m]	[mm]	[m]	[mm]	[mm]	[mm/s]
1	0.00			1.4328		1.3694		-0.3254			
2	32.99	32.99	30.31	1.4327	-0.1	1.3693	-0.1	-0.3255	-0.1	0.17	5.25
3	66.92	33.93	29.47	1.4326	-0.1	1.3690	-0.3	-0.3253	0.2	0.37	11.03
4	100.00	33.08	30.23	1.4327	0.1	1.3686	-0.4	-0.3253	0	0.41	12.46
5	132.99	32.99	30.31	1.4326	-0.1	1.3683	-0.3	-0.3252	0.1	0.33	10.05
6	166.92	33.93	29.47	1.4323	-0.3	1.3681	-0.2	-0.3254	-0.2	0.41	12.15
7	200.00	33.08	30.23	1.4324	0.1	1.3676	-0.5	-0.3252	0.2	0.55	16.56
8	232.99	32.99	30.31	1.4323	-0.1	1.3674	-0.2	-0.3254	-0.2	0.30	9.09
9	266.92	33.93	29.47	1.4323	0	1.3670	-0.4	-0.3255	-0.1	0.41	12.15
10	300.00	33.08	30.23	1.4321	-0.2	1.3668	-0.2	-0.3256	-0.1	0.30	9.07

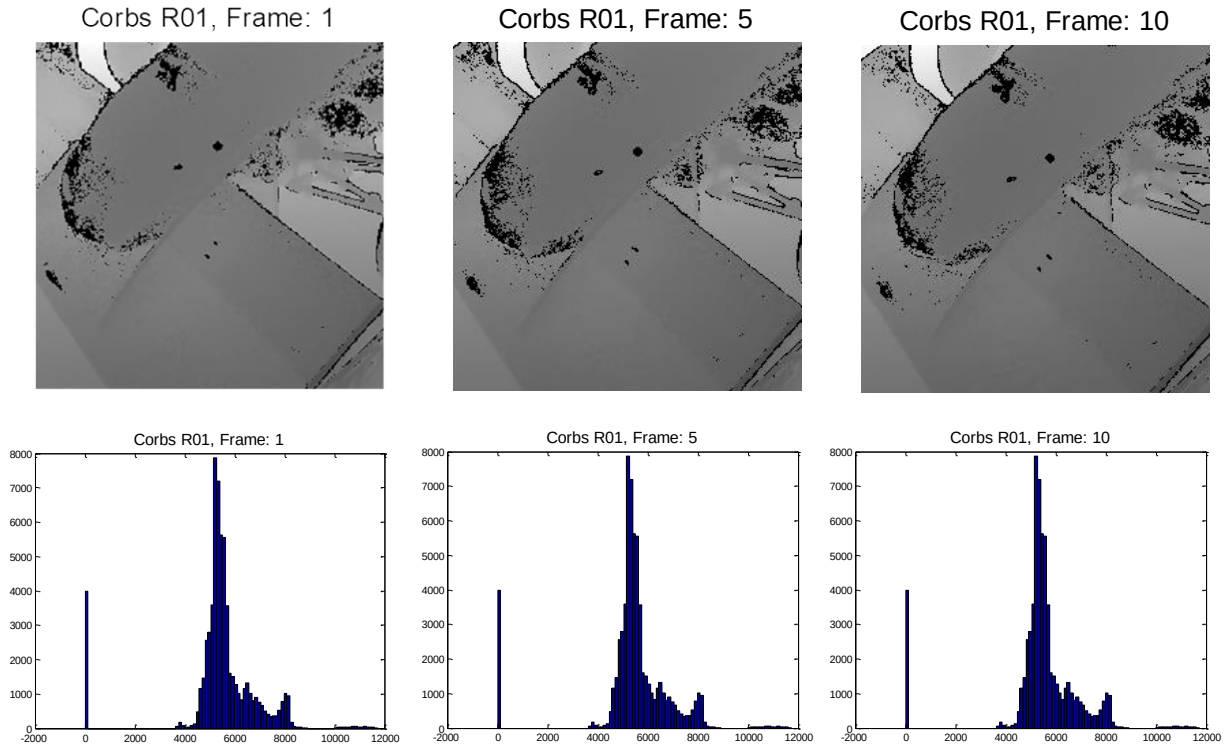


Figure 90: Race Car (R01) sequence, selected frames (256x256) and their respective histograms.

Figure 91: TV / TGV / TTGV Results for sequence R01 (512x512).

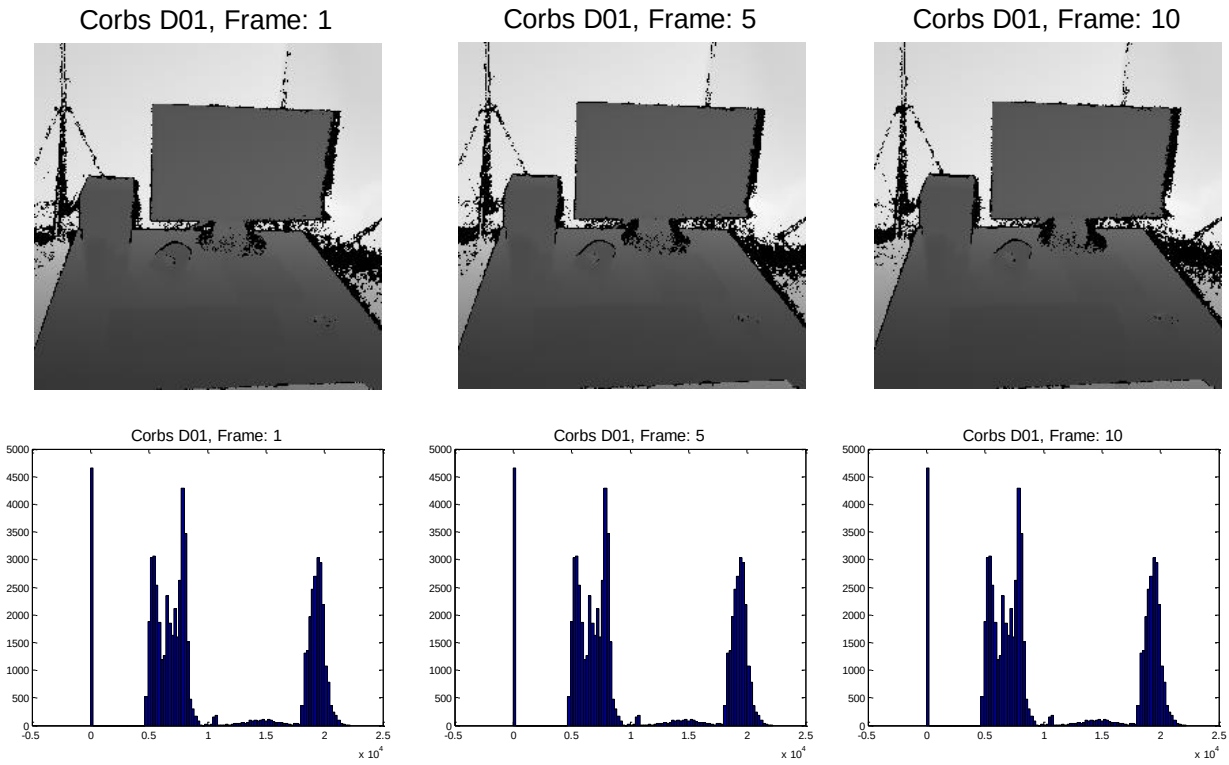


Figure 92: Desk (D01) sequence, selected frames (256x256) and their respective histograms.

Figure 93: TV / TGV / TTGV Results for sequence D01 (512x512).

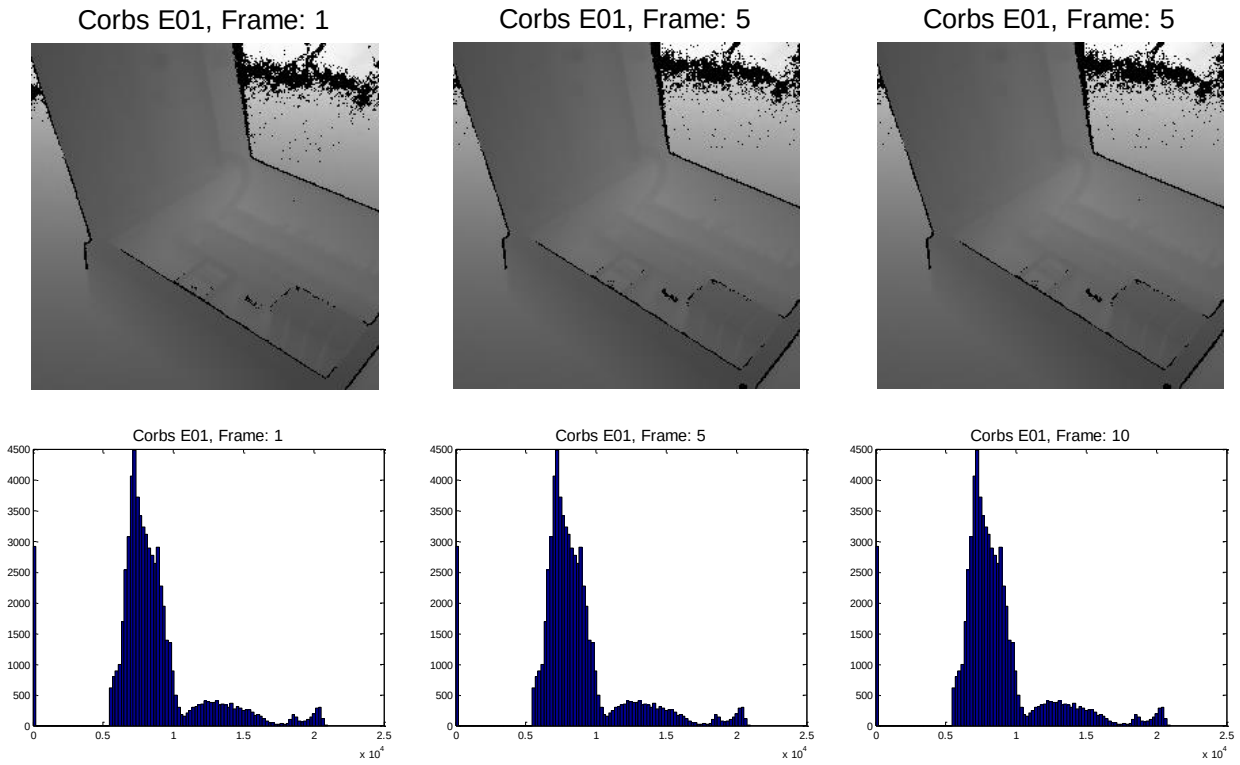


Figure 94: Elec. Central Control (E01) sequence, selected frames (256x256) and their respective histograms.

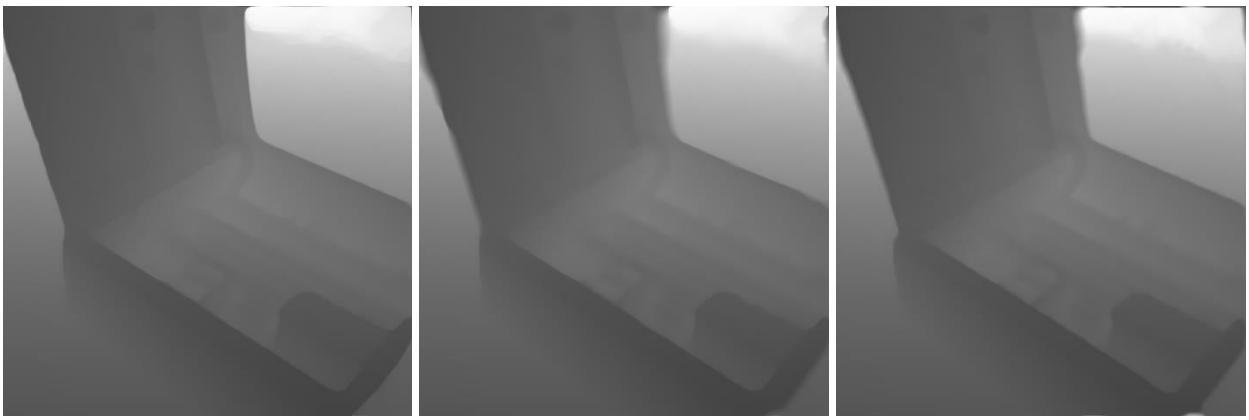


Figure 95: TV / TGV / TTGV Results for sequence E01 (512x512).

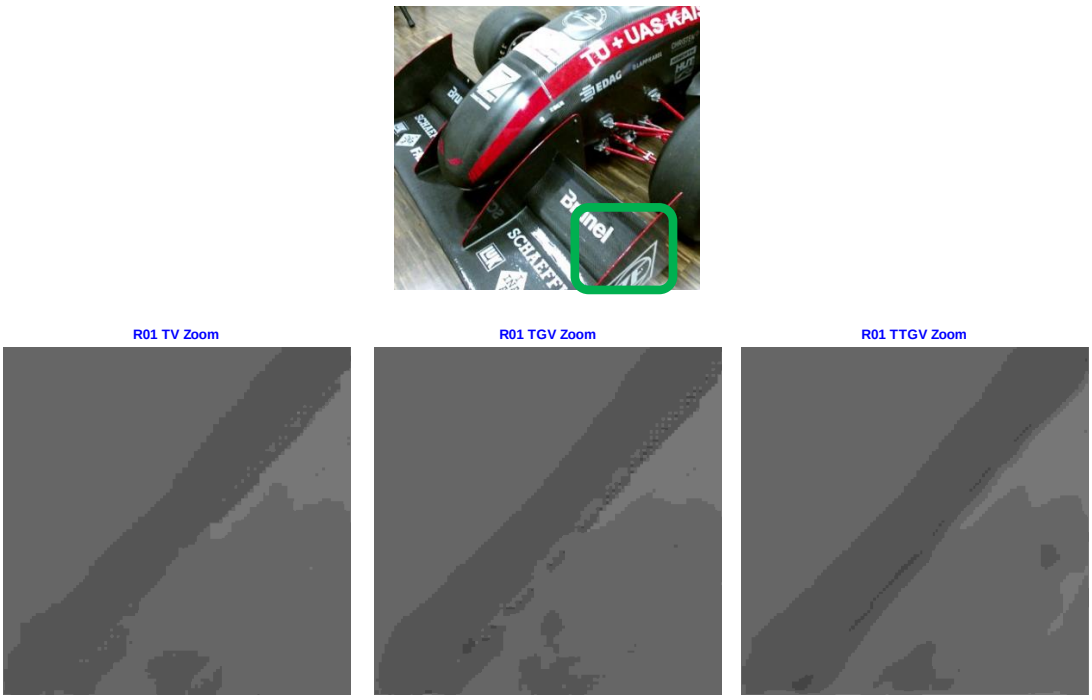


Figure 96: R01 Zoomed Region.

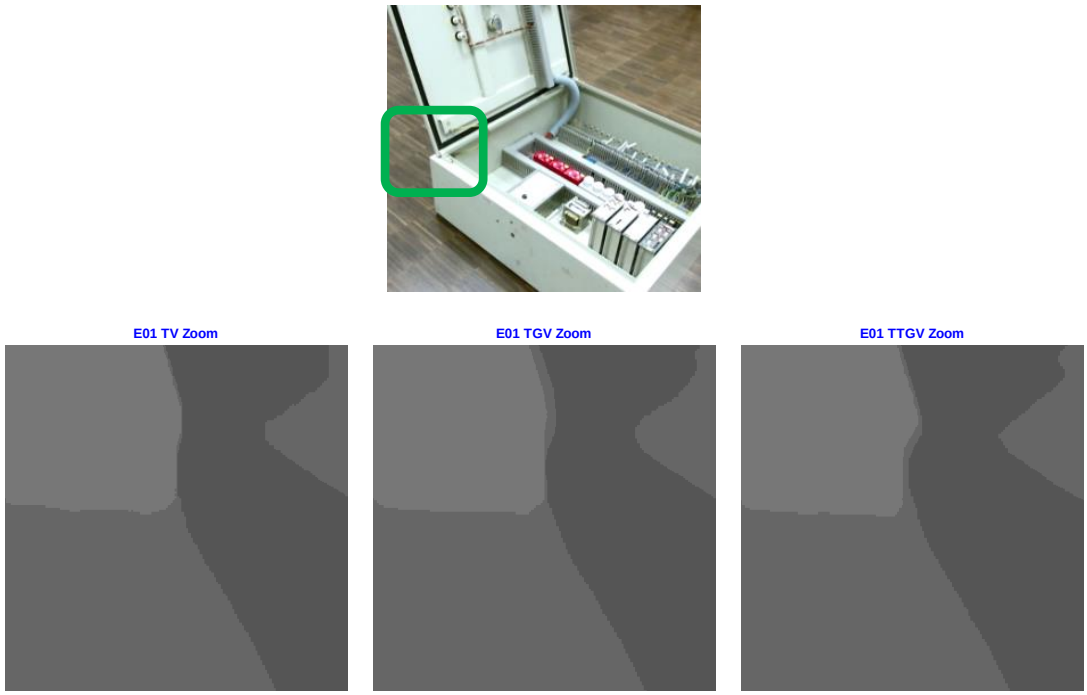


Figure 97: D01 Zoomed Region.

7.4.2 UTEP Depth Maps Dataset Evaluation – Static Camera

This section presents several frame sequences collected with an Intel-Softkinetic ToF camera configured at a rate of 25 frames per second, part of the UTEP data set collected locally. In contrast with frames from the previous section, the camera was not displaced this time. Moreover, two of the frame sequences contain elements that are slightly moving while the rest of the scene remains static, including the camera. This represents a major problem for the Lucas - Kanade registration algorithm employed in the proposed framework since it does not operate correctly when only part of the scene is moving. On the other hand, these cases were included to quantify the performance of the algorithm with this kind of frame sequences.

a) b) c) d)

Figure 98: RGB frames of sequences: a) Fiona (CF01); b) Plant (PB01), c) Mexican hat (MX01) and d) World (PW).

As was done for the Corbs frames, also the statistics for the missing pixels were included for these experiments. However, the sequence CF01 was employing a technique of preprocessing in which all of the pixels that were presenting a low amplitude due to the distance of the camera to the objects of the scene were set to 3000, which is the typical maximum range of a valid reflected

signal. In this way, no zero value pixels were present in the frames. Additionally, CF01 has in the left side a small dog who is moving the head slightly in the last frames, generating a bad registration around this area in the final results as can be seen in the corresponding results. On the other hand, sequence PB01 had extremely small displacement of a few leaves of the tree, those that did not affect the registration process. It is important to mention that the sequence MX01 has a reduced number of missing pixels, and for this one any zero value was set to the maximum during the preprocessing of the frames. Figure 99 to Figure 107 show the detailed results.

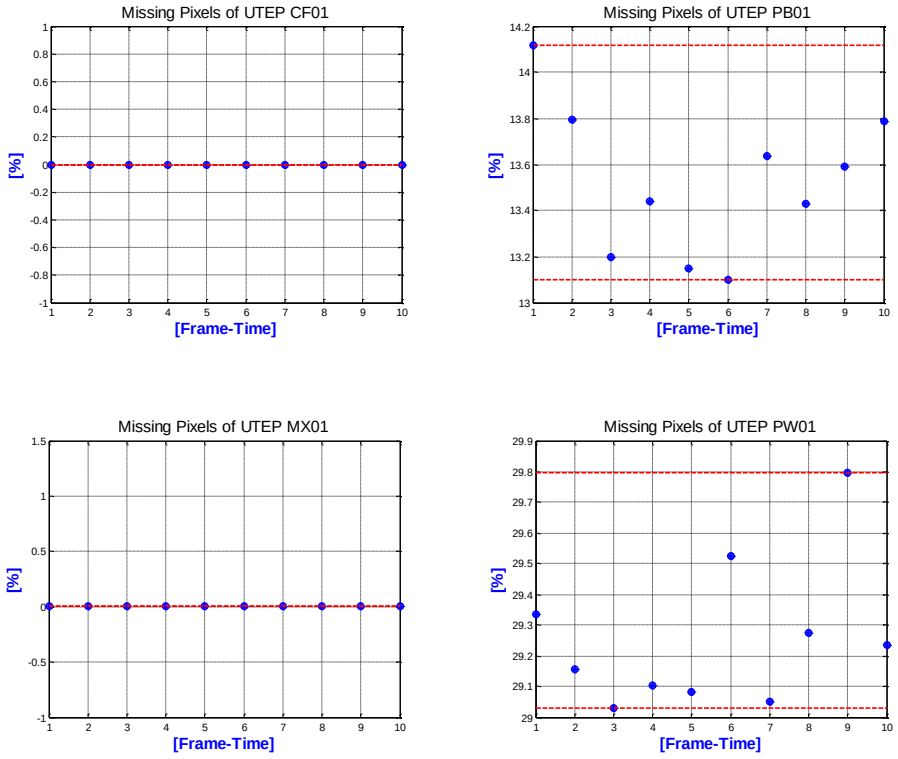


Figure 99: Missing pixel statistics of UTEP sequences FC01, PB01, MX01 and PW01.

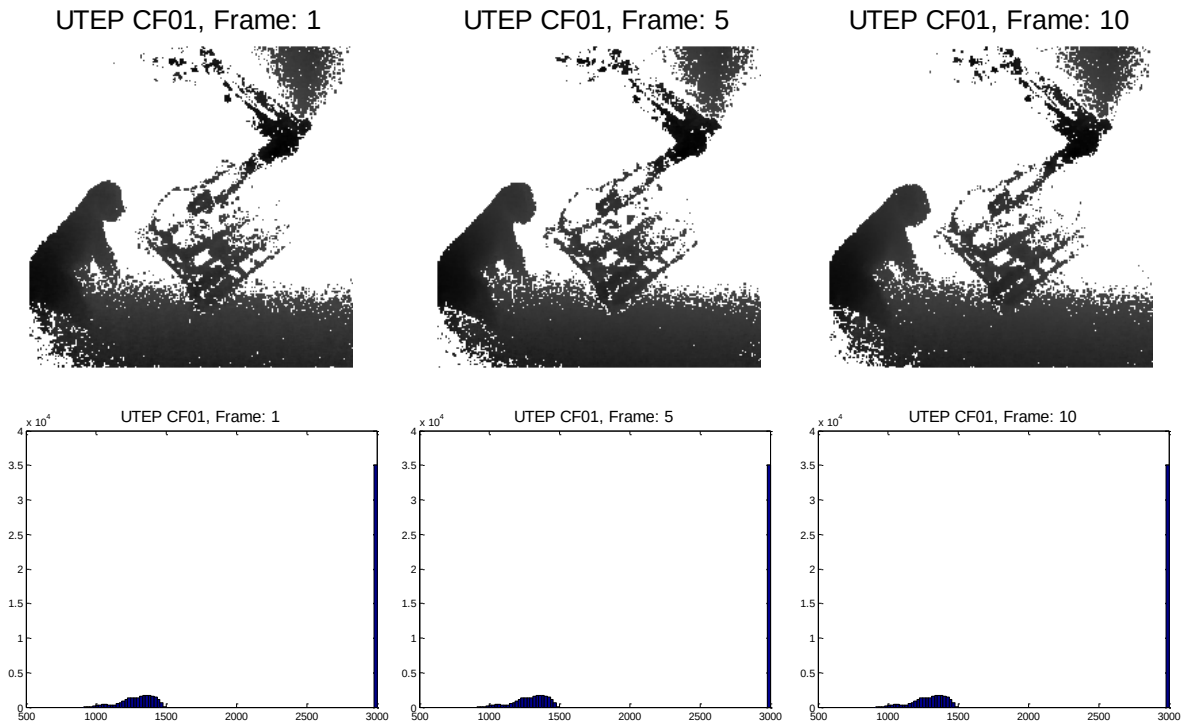


Figure 100: Fiona (CF01) sequence, selected frames (240x240) and their respective histograms.

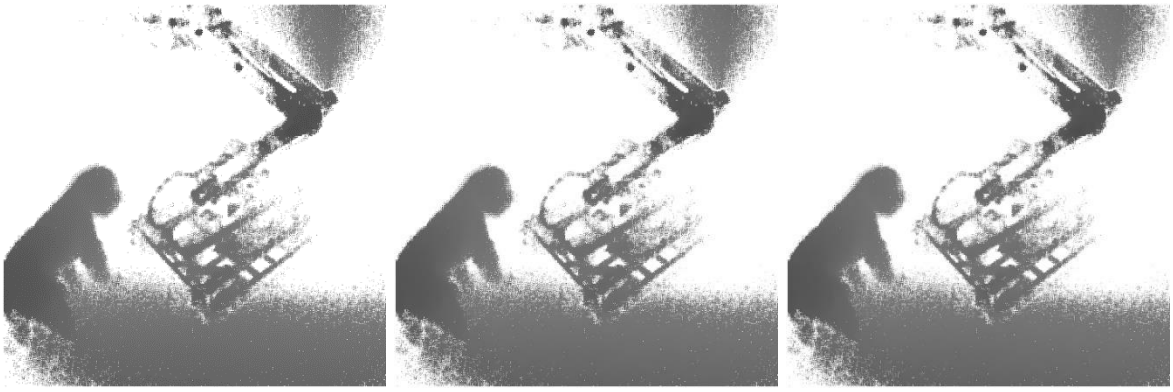


Figure 101: TV / TGV / TTGV Results for sequence CF01 (480x480).

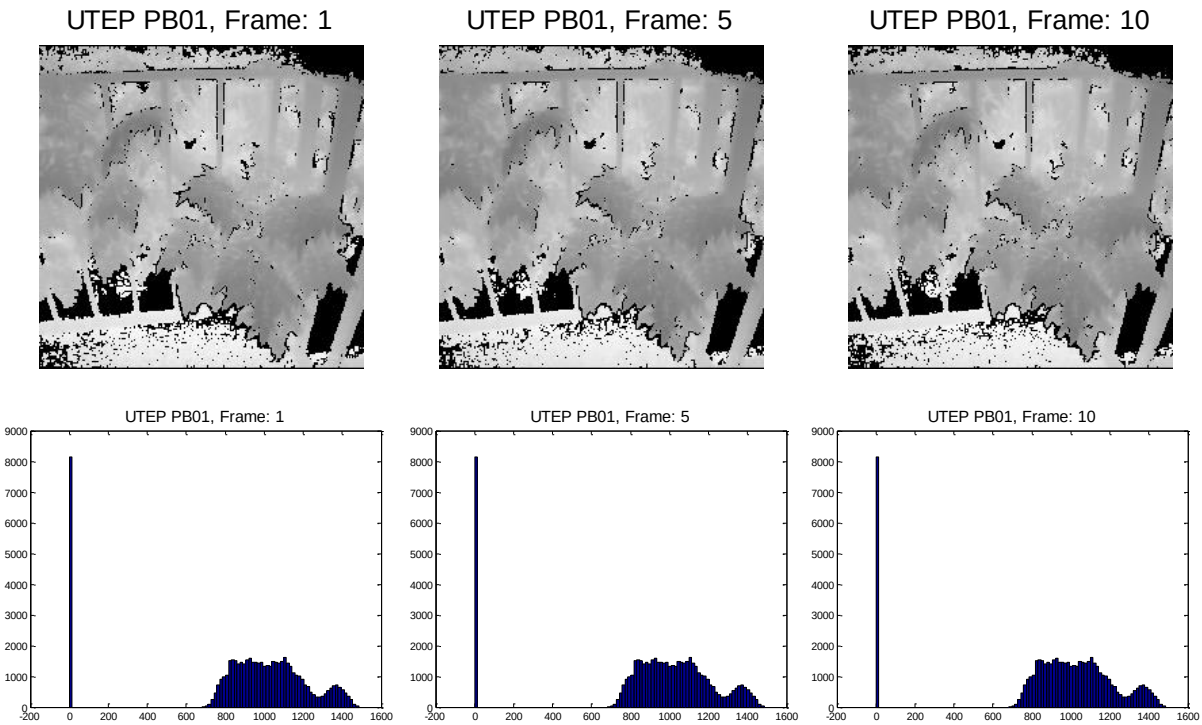


Figure 102: Plants (PB01) sequence, selected frames (240x240) and their respective histograms.

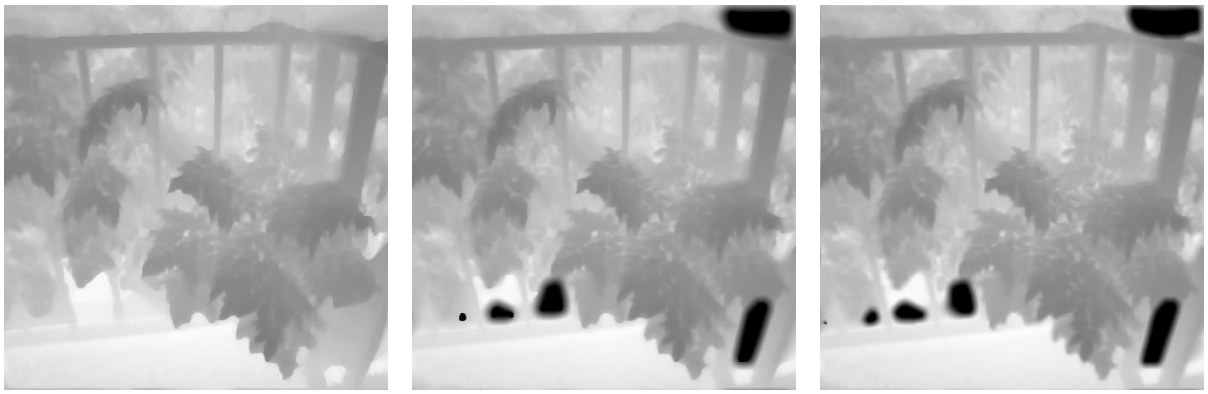


Figure 103: TV / TGV / TTGV Results for sequence PB01 (480x480).

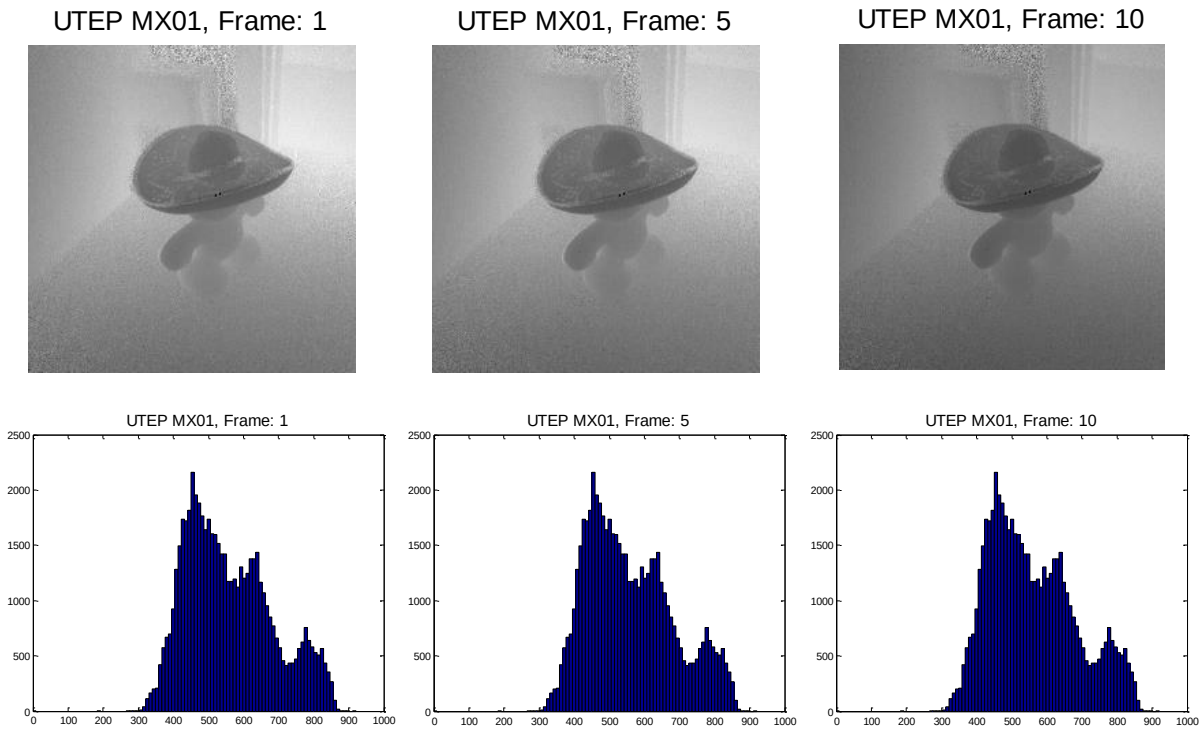


Figure 104: Mexican Hat (MX01) sequence, selected frames (240x240) and their respective histograms.

Figure 105: TV / TGV / TTGV Results for sequence MX01 (480x480).

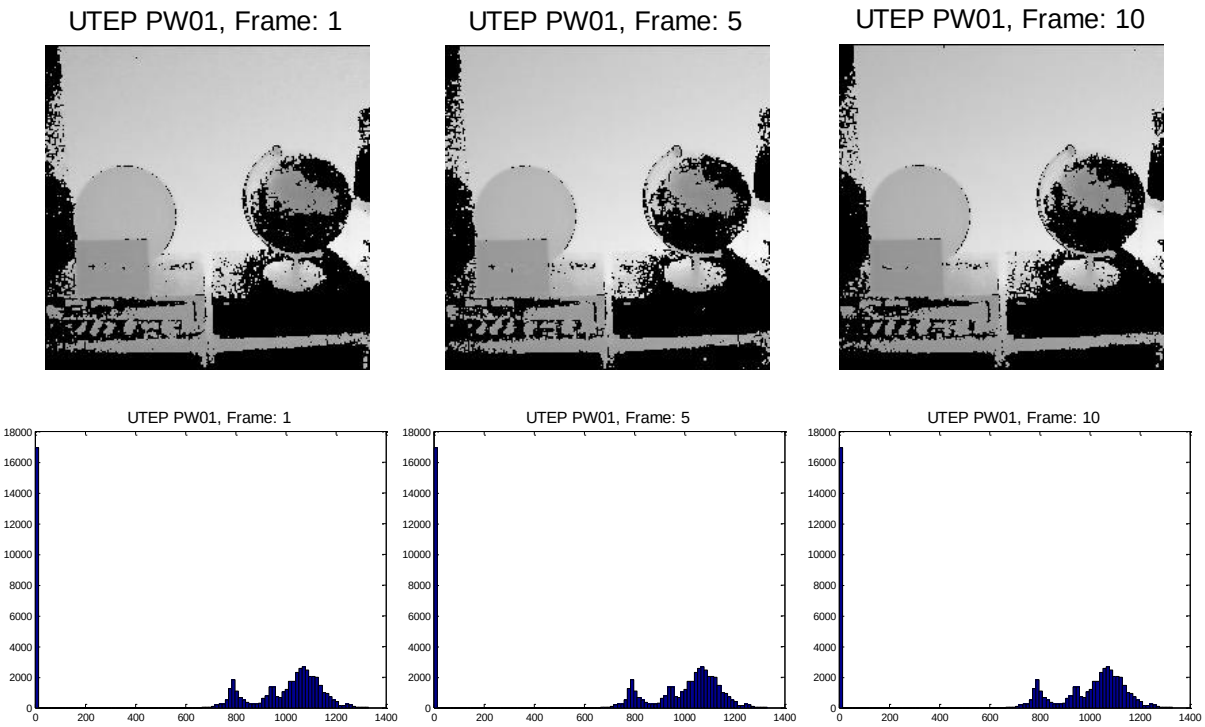


Figure 106: World (PW01) sequence, selected frames (240x240) and their respective histograms.

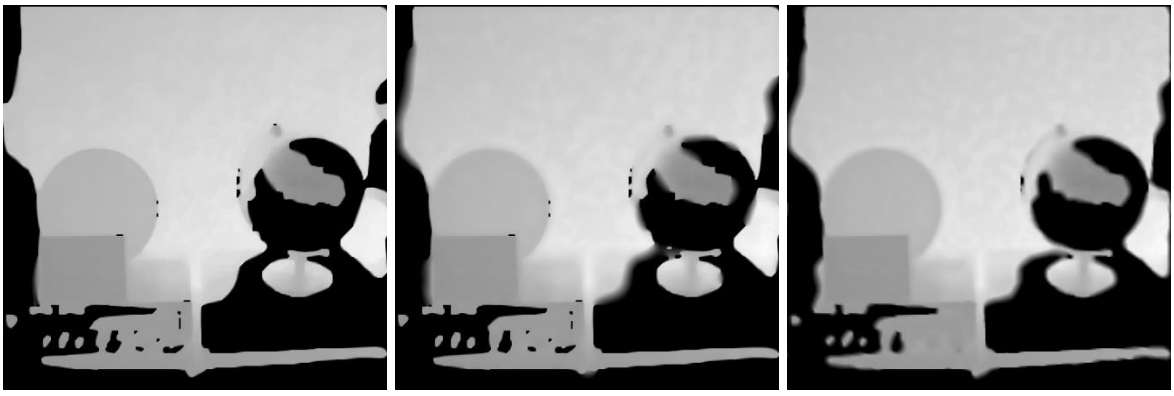


Figure 107: TV / TGV / TTGV Results for sequence PW01 (480x480).

Chapter 8: Conclusions and Future Work

This work has proposed a new framework that improves super-resolution and denoising of ToF depth frames and simulated corrupted stereo depth maps using variational methods. Three different regularization functionals, TV, TGV and TGV with tensor (TTGV), were tested over a range of simulated and real images. The best performance with the single realization experiments with simulated depth frames for the main proposed regularization term, TGV with tensor, was under Salt and Pepper impulsive noise scenarios reaching higher PSNRs and SSIMs values in less computational time. On the other hand, the performance of the TV and TGV regularization terms were superior in both figures of merit, PSNR and SSIM, in general after a few iterations. However, for a 10-realization experiment shown that the TTGV had better performance for high levels of random value impulsive noise.

In this context, it is also important to remark that in the case of the real images it is highly important to avoid artifacts, such phenomena were present overall in the TV regularization term, while the TGV and TGV with tensor left big regions where the information was missing and it could be dangerous to add wrong information in the case of safety applications. Also, it can be seen that some details are preserved with the TGV with tensor that the other two methods were not able to preserve.

The goal of this research was to improve the performance of the structure tensor so that the synergy between the denoising capability of the proposed frame work in conjunction with the registration process could be maximized and the algorithm be able to reach higher performance levels in reduced computational time as well as final better image quality results than the simple TV and TGV functional.

The proposed framework is highly relevant for other technologies in which the physical construction of the imagers is still limited in regards to the spatial resolution. In this context, systems using thermopile as unit pixel cell sensors easily could use this framework to upscale and denoise sequence of frames.

Improved registration process need to be evaluated such as the ITK approach [56]. More robust the algorithms and other optical flow and motion estimation techniques are required to be integrated and evaluated.

Other future work would be to implement estimation of impulsive noise levels automatically to eliminate the *a priori* knowledge requirement.

Finally, if a practical implementation is pursued, then the use of parallel processing needs to be the first step to be implemented and Graphical Processor Units (GPU) can offer a suitable option to accomplish this.

References

[1] S. B. Gokturk, H. Yalcin and C. Bamji, "A time-of-flight depth sensor – system description, issues and solutions," in *Computer Vision and Pattern Recognition Workshop*, Washington, 2004.

[2] C. Niclass, A. Rochas, P.-A. Besse and E. Charbon, "A CMOS 3D camera with millimetric depth resolution," in *Custom Integrated Circuits Conference*, Orlando, 2004.

[3] G. Zach, M. Davidovic and H. Zimmermann, "Sunlight-proof optical distance measurements with a dual-line," in *Analog Integr Circ Sig Process*, 2011.

[4] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color Images," in *Sixth International Conference on Computer Vision*, Bombay, 1998.

[5] M. Keller and A. Kolb, "Real-time simulation of time-of-flight sensors," *Simulation Modelling Practice and Theory*, vol. 17, no. 5, p. 967–978, 2009.

[6] S. Lee, K. Bae, K.-m. Kyung and T.-C. Kim, "Adaptive switching filter for noise removal in highly corrupted depth maps from time-of-flight image sensors," *Proc. SPIE*, vol. 8290, pp. 8290 - 8290 - 7, 2012.

[7] J. Kopf, M. F. Cohen, D. Lischinski and M. Uyttendaele, "Joint bilateral up-sampling," *ACM Transactions on Graphics (TOG)*, no. 3 - Article No. 26, 2007.

[8] J. Diebel and S. Thrun, "An application of markov random fields to range sensing," Stanford AI Lab, Stanford, 2005.

[9] S. Schwarz, M. Sjöström and R. Olsson, "A weighted optimization approach to time-of-flight sensor fusion," *IEEE Transactions on Image Processing*, vol. 23, no. 1, pp. 214-225, october 2014.

[10] L. Jovanov, H. Q. Luong and W. Philips, "S5-3Depth video enhancement for 3D displays," in *SID Mid-Europe Chapter Spring Meeting*, 2013.

[11] A. A. Goshtasby, 2-D and 3-D Image Registration for Medical, Remote Sensing and Industrial Applications, New Jersey: Wiley-Interscience, 2005.

[12] E. Trucco and A. Verri, Introductory Techniques for 3-D Computer Vision, New Jersey: Prentice Hall, 1998.

[13] J. Geng, "Structured-light 3D surface imaging: a tutorial," *Advances in Optics and Photonics,* vol. 3, no. 2, pp. 128-160, 2011.

[14] B. Freedman, A. Shpunt, M. Machline and Y. Arieli, "Depth Mapping Using Projected Patterns". United States of America Patent 20080240502A1, 2 October 2008.

[15] S. Foix, G. Alenya and C. Torras, "Lock-in time-of-flight (ToF) cameras: a survey," *IEEE Sensors,* vol. 11, no. 9, pp. 1917-1926, 2011.

[16] D. Piatti and F. Rinaudo, "SR-4000 and CamCube3.0 Time of Flight (ToF) cameras: tests and comparison," *Remote Sensing,* vol. 4(4), pp. 1069-1089, 2012.

[17] M. Hansard, S. Lee, O. Choi and R. Horaud, Time-of-Flight Cameras: Principles, Methods and Applications, London: Springer, 2013.

[18] L. Li, "Time-of-flight camera – an introduction," Texas Instruments Technical White Paper SLOA 190B, 2014.

[19] D. R. Lanman, "Scientist Research web page," MIT, 1 May 2014. [Online]. Available: http://mesh.brown.edu/dlanman. [Accessed 24 March 2014].

[20] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," April 2002. [Online]. Available: http://vision.middlebury.edu/stereo/data/. [Accessed 17 December 2014].

[21] D. Ferstl, C. Reinbacher, R. Ranftl, M. Ruther and H. Bischof, "Image guided depth upsampling using anisotropic total generalized variation," in *IEEE International Conference on Computer Vision (ICCV),* Sidney, 2013.

[22] O. Wasenmüller, M. Meyer and D. Stricker, "CoRBS: Comprehensive RGB-D Benchmark for SLAM using Kinect v2," IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Placid, 2016.

[23] T. Kohler, S. Haase, S. Bauer, J. Wasza, T. Kilgus, L. Maier-Hein, H. FeuBner and J. Hornegger, "ToF Meets RGB: Novel Multi-Sensor Super-Resolution for Hybrid 3-D Endoscopy," in *Medical Image Computing and Computer-Assisted Intervention - MICCAI,* 2013.

[24] S. Gasparrini, E. Cippitelli, S. Spinsante and E. Gambi, "A depth-based fall detection system using a Kinect® sensor," *Sensors,* vol. 14, no. 2, pp. 2756-2775, 2014.

[25] Y. Soh, J.-Y. Sim, C.-S. Kim and S.-U. Lee, "Superpixel-based depth image super-resolution," in *Three-Dimensional Image Processing (3DIP) and Applications II, Proceedings SPIE*, 2012.

[26] A. Blake, K. Pushmeet and C. Rother, Markov Random Fields for Vision and Image Processing, Massachusetts Institute of Technology, 2011.

[27] J.-Y. Bouguet, "Camera calibration toolbox for Matlab," Computer Vision Research Group. Dept. of Electrical Engineering, California Institute of Technology, 2 December 2013. [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html#examples. [Accessed July 2014].

[28] Q. Yang, R. Yang, J. Davis and D. Nister, "Spatial-Depth Super Resolution for Range Images," in *IEEE Conference on Computer Vision and Pattern Recognition CVPR*, Minneapolis, 2007.

[29] D. Min, J. Lu and M. N. Do, "Depth video enhancement based on weighted mode filtering," *IEEE Transactions on Image Processing,* vol. 21, no. 3, pp. 1176-1190, 2012.

[30] D. Ferstl, M. Ruther and H. Bischof, "Variational depth superresolution using example-based edge representations," in *IEEE International Conference on Computer Vision (ICCV)*, Santiago, 2015.

[31] B. D. Lucas and T. Kanade, "An alternative image registration technique with an application to stereo vision," in *Proceedings International Joint Conference on Artificial Intelligence (IJCAI)*, Vancouver, 1981.

[32] SoftKinetic, "Camera Calibration DepthSenseSDK v1.0," SoftKinetic International SA/NV, Belgium, 2015.

[33] H. Yu, K. Zhao, Y. Wang, L. Kan, M. Sun and J. Wenyan, "Registration and fusion for ToF camera and 2D camera reading," in *Chinese Automation Congress (CAC)*, Changsha, 2013.

[34] D. Ferstl, C. Reinbacher, G. Riegler, M. Ruther and H. Bischof, "Learning depth calibration of time-of-flight cameras," in *Proceedings of British Machine Conference*, 2015.

[35] I. Silva, B. Mederos, L. Ortega-Maynez and S. Cabrera, "Multi-frame super-resolution for mixed Gaussian and impulse noise based on blind inpainting," in *IEEE Signal Processing and Signal Processing Education Workshop (SP/SPE)*, Salt Lake City, 2015.

[36] S. Villena, M. Vega, R. Molina and A. K. Katsaggelos, "A non-stationary image prior combination in super-resolution," *Digital Signal Processing*, vol. 32, pp. 1-10, 2014.

[37] A. K. Katsaggelos, R. Molina and J. Mateos, Super Resolution of Images and Video. Synthesis Lectures on Image, Video, and Multimedia Processing, Morgan & Claypool Publishers, 2007.

[38] P. Milanfar, Super-Resolution Imaging, CRC Press, Taylor & Francis Group, 2011.

[39] K. Bredies and T. Valkonen, "Inverse problems with second-order total generalized variation constraints," in *Proceedings of SampTA 2011 - 9th International Conference on Sampling Theory and Applications*, Singapore, 2011.

[40] C. He, C. Hu, X. Yang, H. He and Z. Qi, "An adaptive total generalized variation model with augmented Lagrangian method for image denoising," *Mathematical Problems in Engineering*, vol. 2014, no. Article ID 157891, 2014.

[41] M. Argaez, C. Ramirez and R. Sanchez, "An ℓ1-algorithm for underdetermined systems and applications," in *IEEE Conference Procedings of the 2011 Annual Conference on North American Fuzzy Information Processing Society NAFIPS*, El Paso, 2011.

[42] M. Argaez, R. Sanchez and C. Ramirez, "Face recognition from incomplete measurements via ℓ1-optimization," *American Journal of Computational Mathematics*, vol. 2, no. 4, pp. 287-294, 2012.

[43] B. Wahlberg, S. Boyd, M. Annergren and Y. Wang, "An ADMM algorithm for a class of total variation regularized estimation problems," in *Proceedings of 16th IFAC Symposium on System Identification*, Brussels, 2012.

[44] S. Boyd, P. Neal, C. Eric, B. Peleato and E. Jonathan, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1-122, 2011.

[45] J. Douglas and H. H. Rachford, "On the numerical solution of heat conduction problems in two and three space variables," *Transactions of the American Mathematical Society* , vol. 82, pp. 421-439, 1956.

[46] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element apporximations," *Comp. & Maths. with Appls.*, vol. 2, pp. 17-40, 1976.

[47] N. Parikh and S. Boyd, "Proximal Algorithms," *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 123-231, 2013.

[48] C. He, C. Hu, W. Zhang, B. Shi and X. Hu, "Fast Total-Variation Image Deconvolution with Adaptive Parameter Estimation via Split Bregman Method," *Mathematical Problems in Engineering*, vol. 2014, no. Article ID 617029, p. 9, 2014.

[49] S. Di Zenzo, "A note on the gradient of a multi-image," *Computer Vision, Graphics, and Image Processing*, vol. 33, no. 1, pp. 116-125, 1986.

[50] U. Köthe, "Edge and junction detection with an improved structure tensor," *Michaelis B., Krell G. (eds) Pattern Recognition. DAGM 2003. Lecture Notes in Computer Science*, vol. 2781, pp. 25-32, 2003.

[51] G. Peyre, "Toolbox Image Version 1.2," 2009. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/16201-toolbox-image?s_tid=prof_contriblnk. [Accessed 6 February 2017].

[52] M. Marin-McGee and M. Velez-Reyes, "A Spectrally Weighted Structure Tensor for Hyperspectral Imagery," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* , vol. 9, no. 9, pp. 4442 - 4449, 2016.

[53] J. Weickert, "Coherence-enhancing diffusion filtering," *International Journal of Computer Vision*, vol. 31, no. 2-3, p. 111–127, 1999.

[54] Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600-612, 2004.

[55] P. F. Judy, "Video Monitor Test Pattern Tutorials - Bridham Women's Hospital Rad. Department," 8 April 1997. [Online]. Available: http://www.engr.uconn.edu/~alberto/Catalog/Patterns/Tutorial.htm. [Accessed 9 January 2017].

[56] H. J. Johnson, M. M. McCormick and I. L. , The ITK Software Guide Book 1: Introduction and Development Guidelines - Volume 1, Kitware, Inc., 2015.

[57] P. J. Davis, Circulant Matrices, New York: Wiley-Interscience Publication, 1970.

Appendices

Appendix A – Norm 1 Proximal Shrinkage

This appendix provides a description of the Norm-1 proximal shrinkage previously used in Section Chapter 5: Numerical Method Implementation for Depth Super-Resolution. Although Chapter 7 uses this operator over a Norm-1, at the end, it operates over the sum of multiple Norm-2's which are the elements fed to the Norm-1.

Consider the following case of the Norm-1 in which it is the summation of Norm-2 terms applied to scalar operands as follows:

$$\|\mathbf{X}\|_1 = \|x_1\|_2 + \|x_2\|_2 + \cdots + \|x_n\|_2$$

If that is the case then this previous expression is equivalent to the following Norm-1 definition:

$$\|\mathbf{X}\|_1 = |\mathbf{x}_1| + |\mathbf{x}_1| + \cdots + |\mathbf{x}_n|$$

And subsequently:

$$\|\mathbf{X}\|_1 = \sqrt{(x_1)^2} + \sqrt{(x_2)^2} + \cdots + \sqrt{(x_n)^2}$$

This appendix first introduces a simple derivation of the Proximal Shrinkage with a Norm-1 operating over a vector and then the case for Norm-2 is introduced.

First consider the proximal as the solution of the following optimization problem, which is convex, multidimensional and with separable variables:

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \|\mathbf{x}\|_1 + \frac{1}{2\lambda}\|\mathbf{x}-\mathbf{v}\|_2^2 \right\} = prox_{\lambda\|\cdot\|_1}(\mathbf{v}) \qquad \text{(A1)}$$

Where **x** and **v** belong to R^n; i.e. $\mathbf{x} = [x_1 \quad x_2 \quad \cdots \quad x_n]^T$ and $\mathbf{v} = [v_1 \quad v_2 \quad \cdots \quad v_n]^T$

Then (A1) can be separated in the following way:

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} |x_i| + \sum_{i=1}^{n} \frac{1}{2\lambda} (x_i - v_i)^2 \right\}$$

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} \left(|x_i| + \frac{1}{2\lambda} (x_i - v_i)^2 \right) \right\}$$

Since there are *n* separated variables, the next expression is equivalent to *n* one-dimensional convex optimization problems:

$$x_i^* = \underset{x_i}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} \left(|x_i| + \frac{1}{2\lambda} (x_i - v_i)^2 \right) \right\}$$

In this way the optimal solution can be composed by the n solutions:

$$\mathbf{x}^* = [x_1^* \quad x_2^* \quad \cdots \quad x_n^*]^T$$

Since the function $|x_i| + \frac{1}{2\lambda}(x_i - v_i)^2$ is convex and non-smooth due to the absolute value operator, there is a need to apply the following lemma:

Lemma: Let *f* be a convex function. If **0** belongs to the subdifferential of *f* at some point ***y***, then *f* attains at ***y*** a minimum value.

Consider the absolute value function $f = |x_i|$, the subdifferential of f denoted by ∂f is computed as:

$$\partial|x_i| = \begin{cases} 1 & if\ x_i > 0 \\ -1 & if\ x_i < 0 \\ [-1,1] & if\ x_i = 0 \end{cases}$$

Now it is possible to compute the subdifferential of the function $|x_i| + \frac{1}{2\lambda}(x_i - v_i)^2$:

$$\partial\left(|x_i| + \frac{1}{2\lambda}(x_i - v_i)^2\right) = \partial|x_i| + \partial\left(\frac{1}{2\lambda}(x_i - v_i)^2\right) = \partial|x_i| + \frac{1}{\lambda}(x_i - v_i)$$

Therefore:

$$\partial\left(|x_i| + \frac{1}{2\lambda}(x_i - v_i)^2\right) = \begin{cases} 1 + \frac{1}{\lambda}(x_i - v_i) & if\ x_i > 0 \\ -1 + \frac{1}{\lambda}(x_i - v_i) & if\ x_i < 0 \\ [-1,1] + \frac{1}{\lambda}(x_i - v_i) & if\ x_i = 0 \end{cases} \quad \text{(A2)}$$

Then, it is necessary to find the *y* value such that the subdifferential (A2) evaluated in *y* attains a 0 which contains a minimum.

$$0 = \begin{cases} 1 + \frac{1}{\lambda}(y - v_i) & if\ y > 0 \\ -1 + \frac{1}{\lambda}(y - v_i) & if\ y < 0 \\ [-1,1] + \frac{1}{\lambda}(y - v_i) & if\ y = 0 \end{cases} \quad \text{(A3)}$$

The last expression defined by (A3) is examined in three cases, in which solving for y in each one will provide the required values for y and v_i. The three cases of the expression (A3) are:

Case 1:

$$y = v_i - \lambda \quad \text{with } \boldsymbol{y > 0}\text{, which requires } \boldsymbol{v_i > \lambda}$$

Case 2:

$$y = v_i + \lambda \quad \text{with } \boldsymbol{y < 0}\text{, which requires } \boldsymbol{v_i < -\lambda}$$

Case 3: This case is analyzed in the range generated by the third condition of (A3) in the following way:

$$0 \; in \; [-1,1] + \tfrac{1}{\lambda}(y - v_i) \text{ with } \boldsymbol{y < 0}$$

$$0 \; in \; \left[-1 + \tfrac{1}{\lambda}(y - v_i), 1 + \tfrac{1}{\lambda}(y - v_i)\right] \text{ with } \boldsymbol{y < 0}$$

This last expression is possible if and only if

$$-1 + \frac{1}{\lambda}(y - v_i) < 1 + \frac{1}{\lambda}(y - v_i)$$

Which is always true. The three cases can be summarized as:

$$y^* = \begin{cases} v_i - \lambda & v_i > \lambda \\ v_i + \lambda & v_i < -\lambda \\ 0 & if \; v_i \; in \; [-\lambda, \lambda] \end{cases} \tag{A4}$$

Equation (A4) can be re-written in a different fashion covering the three cases and still holding the respective conditions:

$$y^* = \max\{|v_i| - \lambda, 0\} \times \frac{v_i}{|v_i|}$$

The case of the Norm-2 $\|\cdot\|_2$ can be obtained in a similar way, but it is based on a more complex argument which is the conjugate function, specifically the Moreau decomposition [47].

Then the proximal for the Norm-2 case is defined as:

$$prox_{\lambda\|\cdot\|_2}(\mathbf{v}) = \begin{cases} \left(1 - \dfrac{\lambda}{\|x\|_2}\right) x & \|x\|_2 \geq \lambda \\ 0 & otherwise \end{cases}$$

By rewriting the last equation, the following is obtained:

$$prox_{\lambda\|\cdot\|_2}(\mathbf{v}) = \{\|\mathbf{v}\|_2 - \lambda, \quad 0\} \times \frac{\mathbf{v}}{\|\mathbf{v}\|_2}$$

Appendix B – Circulant Matrix and Discrete Fast Fourier Transform

Circulant Matrices were originally introduced by Davis in [57] and their application in linear equations makes it possible to solve them in fast and efficient ways. Consider the circulant $n \times n$ matrix A:

$$A = \begin{bmatrix} a_1 & a_n & \cdots & a_3 & a_2 \\ a_2 & a_1 & a_n & & a_3 \\ \vdots & a_2 & a_1 & \ddots & \vdots \\ a_{n-1} & & \ddots & \ddots & a_{n-1} \\ a_n & a_{n-1} & \cdots & a_2 & a_1 \end{bmatrix}$$

This matrix can be specified by its first column and the rest of the columns are just circular shifts of the first column one position down as it can be seen in the generic matrix A. The main applications of circulant matrices are to solve linear equations involving circular convolutions.

Consider the following linear system: $Ax = b$. This system can be written using circular convolutions in the following way: $a * x = b$, where a is the first column of the $n \times n$ matrix A. Thus, the system can be rewritten in the following way using the Discrete Fourier Transform (DFT):

$$\mathcal{F}(a * x) = \mathcal{F}(a)\mathcal{F}(x) = \mathcal{F}(b)$$

Then the solution can be found as:

$$x = \mathcal{F}^{-1}\left\{\frac{\mathcal{F}(b)}{\mathcal{F}(a)}\right\}^T$$

This technique is faster than Gaussian elimination and according to [48], one Discrete Fourier Transform and one inverse DFT require on the order of $O(n^2 \log(n^2))$ operation.

Vita

Salvador Canales earned his Bachelor of Engineering degree in Electronic and Computers from Universidad de las Americas Puebla in 2000. In 2006 he received his Master in Computer Engineering from Instituto Tecnológico y de Estudios Superiores de Monterrey. In 2009 he joined the doctoral program in Electrical and Computer Engineering at The University of Texas at El Paso as part time student.

While pursuing his degree, Dr. Canales worked as Senior Electrical Engineer at Delphi Automotive Systems in its Technical Center in Ciudad Juarez. In 2009 he was granted with two patents in the area of DC Motor Driver Circuits and in 2010 he received another patent in the area of Motor Control Algorithms. He is currently full time Design Engineer with Aptiv (formerly Delphi). In 2013 he participated in the development of the first Automotive Grade 3D Time of Flight Gesture Recognition Camera Prototype, which motivates his dissertation topic.

Dr. Canales' dissertation entitled, "Improving Time-of-Flight and other Depth Images: Super-resolution and Denoising using Variational Methods," was supervised by Dr. Sergio Cabrera and Dr. Boris Mederos. His post-graduation plans are to continue to do research and publish in the field of imaging and image processing.