FAST L1 SMOOTHING SPLINES WITH AN APPLICATION TO KINECT DEPTH DATA

*Mariano Tepper, Guillermo Sapiro**

Department of Electrical and Computer Engineering, Duke University.

ABSTRACT

Splines are a popular and attractive way of smoothing noisy data. Computing splines involves minimizing a functional which is a linear combination of a fitting term and a regularization term. The former is classically computed using a (sometimes weighted) L2 norm while the latter ensures smoothness. In this work we propose to replace the L2 norm in the fitting term with an L1 norm, leading to automatic robustness to outliers. To solve the resulting minimization problem we propose an extremely simple and efficient numerical scheme based on split-Bregman iteration and a DCT-based filter. The algorithm is applied to the problem of smoothing and impainting range data, where high-quality results are obtained in short processing times.

Index Terms— Splines, robust fitting, split-Bregman, grid data

1. INTRODUCTION

Smoothing a dataset consists in finding an approximating function that captures important patterns in the data, while disregarding noise or other fine-scale structures. Let $y \in \mathbb{R}^{n_1 \times \cdots \times n_m} \to \mathbb{C}$ be an m-dimensional discrete signal, where n_j ($1 \leq j \leq m$) is the domain of y along the j-th dimension. We can model y by

$$y = \hat{y} + r, \tag{1}$$

where r represents some noise and $\hat{y}$ is a smooth function. A very common regularization choice is to enforce C^2 continuity, in which case $\hat{y}$ is called a (cubic) spline. Smoothing y relies upon finding the best estimate of $\hat{y}$ under the proper smoothness and noise assumptions. We can approximate $\hat{y}$ by minimizing an objective functional

$$\mathrm{F}(z) = \mathrm{R}_y(z) + s\mathrm{P}(z), \tag{2}$$

where $\mathrm{R}_y(z)$ is a data fitting term, defined by the distribution of r, $\mathrm{P}(z)$ is a regularization term, and s is a scalar that determines the balance between both terms. Such scalar can be automatically derived using Bayesian or MDL techniques.

Regularly sampled signals are extremely common in practice, and their analysis becomes easier and faster. In particular, we follow the most common choice when dealing with discrete m-dimensional data, which is assuming a "rectangular" Cartesian sampling pattern. When the sampling is isotropic, i.e., "square," we refer to this type of data as grid data.

For clarity, we will describe in depth the case $m = 1$, where we have $n = n_1$ samples. We extend the results later to the general m-dimensional case.

Let us begin by explaining the smoothing term. The C^2 continuity requirement leads to define $\mathrm{P}(z) = \|Dz\|_2^2$, D being a discrete *second-order* differential operator. Assuming repeating border elements, that is, $y_0 = y_1$ and $y_{n+1} = y_n$, and that the data are equally spaced we obtain

$$D = \begin{pmatrix} -1 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -1 \end{pmatrix}. \tag{3}$$

Regarding the fitting term, the classical assumption is that the noise r in Eq. (1) has Gaussian distribution with zero mean and unknown variance, which leads to setting $\mathrm{R}_y(z) = \|z - y\|_2^2$. Smoothing then can be formulated as the least-squares regression

$$\hat{y} = \underset{z}{\operatorname{argmin}} \ \|z - y\|_2^2 + s\,\|Dz\|_2^2 . \tag{4}$$

For clarity, we call the estimate $\hat{y}$ obtained with this method an L2 spline. It is a well known fact that least squares estimates are highly non-robust to outliers. Although there is no agreement on a universal and formal definition of an outlier, it is usually regarded as an observation that does not follow the patterns in the data. Notice that smoothing should produce an estimate $\hat{y}$ taking into account only important patterns, that is, the inliers, in the data. In this sense, the L2 formulation cannot correctly handle outliers by itself.

In order to solve this problem, we propose to take a different assumption on the distribution of the noise r in Eq. (1). By choosing a distribution with fatter tails than the Gaussian distribution, the derived estimator will correctly handle outliers. We thus assume that r follows a Laplace distribution with zero mean and unknown scale parameter, a common practice in other problems as we will further discuss below. This leads to the fitting term $\mathrm{R}_y(z) = \|z - y\|_1$ and the regression then becomes

$$\hat{y} = \underset{z}{\operatorname{argmin}} \ \|z - y\|_1 + s\,\|Dz\|_2^2 .$$

*Work partially supported by NSF, ONR, NGA, ARO, DARPA, and NSSEFF. This work was partially done while the authors were with the Department of Electrical and Computer Engineering, University of Minnesota.

We call this estimate $\hat{y}$ an L1 spline.

Let us point out that the use of L1 fitting terms for solving inverse problems is not new. For example, Nikolova proved the theoretical pertinence of using L1 fitting terms for image denoising [1]. Other interesting works have addressed this approach for total variation image denoising [2–5] or total variation optical flow [6–8].

We developed an iterative algorithm for computing L1 splines, based on split-Bregman iteration [9], that is specially suited for the case of grid data. This algorithm is extremely fast, both in running time and in the number of iterations until convergence. It is also outstandingly simple, making the implementation completely straightforward.

The remainder of the paper is structured as follows. In Section 2, we present the proposed algorithm for computing L1 splines. Then, in Section 3, we show results obtained with L1 splines, which systematically outperform their L2 and robust L2 counterparts in the presence of outliers. We also show that the proposed computational algorithm is very efficient. Finally, in Section 4 we provide some concluding remarks.

2. L1 SMOOTHING SPLINES

Goldstein and Osher [9] rediscovered a very elegant and efficient algorithm for solving L1 constrained problems (related to a number of very efficient optimization algorithms, e.g., see [10]). This algorithm is often denoted in the literature as split-Bregman iteration. This class of algorithms has several nice theoretical properties and has successfully been applied to several problems in practice such as image restoration [11], image denoising [12], compressed sensing [13], and image segmentation [14]; see also [10] and references therein.

We use this technique for solving Eq. (1). We first pose the equivalent problem

$$\min_{z,d} \|d\|_1 + s\|Dz\|_2^2 \quad \text{s.t.} \quad d = z - y. \tag{5}$$

This problem can be solved by applying the following Bregman iteration

$$z^{k+1} = \operatorname*{argmin}_z \; s\|Dz\|_2^2 + \tfrac{\lambda}{2}\left\|d^k - z + y - b^k\right\|_2^2, \tag{6}$$

$$d^{k+1} = \operatorname*{argmin}_d \; \|d\|_1 + \tfrac{\lambda}{2}\left\|d - z^{k+1} + y - b^k\right\|_2^2, \tag{7}$$

$$b^{k+1} = b^k + (z^{k+1} - y - d^{k+1}). \tag{8}$$

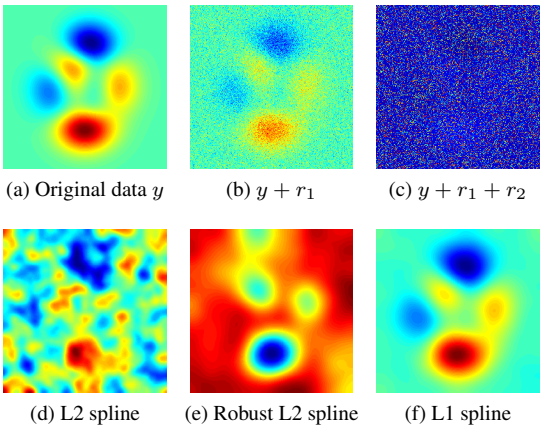
For minimizing Eq. (6), since both terms are differentiable, we obtain

$$z^{k+1} = (I + \tfrac{2s}{\lambda} D^{\mathrm{T}} D)^{-1}(d^k + y - b^k). \tag{9}$$

It can be shown [15] that Eq. (9) can be analytically solved by

$$z^{k+1} = \mathrm{DCT}^{-1}(\Gamma\, \mathrm{DCT}(d^k + y - b^k)), \tag{10}$$

where $\Gamma_{i,i} = \left[1 + \frac{2s}{\lambda}(-2 + 2\cos((i-1)\pi/n))^2\right]^{-1}$, and $\mathrm{DCT}(\cdot)$ and $\mathrm{DCT}^{-1}(\cdot)$ stand for the DCT-II and inverse DCT-II functions. Eq. (10) provides a fast and simple algorithm for computing Eq. (6).



(a) Original data y (b) $y + r_1$ (c) $y + r_1 + r_2$

(d) L2 spline (e) Robust L2 spline (f) L1 spline

Fig. 1. Synthetic 2D example: the original data y is contaminated with Gaussian noise r_1 and then with a large uniform noise r_2. With this input signal, $y + r_1 + r_2$, the proposed method is the only one able to recover the correct shape.

Going back to Eq. (7), the optimal value of d can be explicitly computed using shrinkage operators,

$$d^{k+1} = \mathrm{Shrink}(z^{k+1} - y + b^k, 1/\lambda), \tag{11}$$

where $\mathrm{Shrink}(x, \gamma)_i = \mathrm{sign}(x_i)\max(|x_i| - \gamma, 0)$. We thus obtain a very efficient algorithm for computing L1 splines, combining DCT and shrinkage operators.

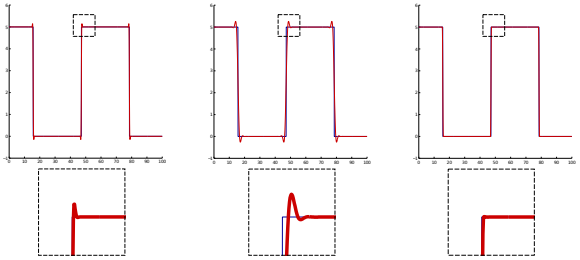
Let us analyze its complexity. The DCT and inverse DCT require $O(n \log n)$ operations, where $n = \prod_{1 \le j \le m} n_j$. The remaining operations are linear in m. The overall complexity of the algorithm is then $O\left(N_i(m + n \log n)\right)$, where N_i is the number of split Bregman iterations. We will later see that in many cases the algorithm converges quickly (N_i can be very small then). The algorithm's complexity is thus dominated by the computation of the DCT and inverse DCT. Of course, these standard operations can be easily computed using GPU, speeding-up the execution by several orders of magnitude.

2.1. Handling missing data

Often in practice there are in y some values y_i that could not be observed (or recorded) for some reason. We would like to be able to handle such cases in such a way that the missing values are inferred from the ones that can be observed. Let W be an $n \times n$ diagonal matrix such that $W_{i,i}$ represents a weight assigned to observation i. W is defined by $W_{i,i} = 0$ if datapoint i is missing, and $W_{i,i} = 1$ otherwise. We then extend Eq. (1) as

$$\hat{y} = \operatorname*{argmin}_z \; \|W(z - y)\|_1 + s\|Dz\|_2^2, \tag{12}$$

which will simply omit the missing points from the computation of the residual while the regularizer will still have a smoothing effect over both present and missing points. Eq. (12) acts as an impainting algorithm, filling the missing



(a) L2 spline (b) Robust L2 spline (c) L1 spline

Fig. 2. In the case of a periodic piece-wise constant signal, the L2 and robust L2 approximations exhibit overshoot and ringing (see zoom-in details on the bottom). These effects are much attenuated by the L1 spline approximation.

values in such a way that continuity between filled values and smoothed ones is preserved.

Then the split-Bregman iteration can be written as

$$z^{k+1} = \underset{z}{\operatorname{argmin}} \; \tfrac{2s}{\lambda} \left\|Dz\right\|_2^2 + \left\|W(d^k - z + y - b^k)\right\|_2^2, \tag{13}$$

$$d^{k+1} = \underset{d}{\operatorname{argmin}} \; \left\|Wd\right\|_1 + \tfrac{\lambda}{2} \left\|W(d - z^{k+1} + y - b^k)\right\|_2^2, \tag{14}$$

$$b^{k+1} = b^k + (z^{k+1} - y - d^{k+1}). \tag{15}$$

Solving Eq. (14) amounts to performing a shrinkage operation on the dimensions i where $W_{i,i} = 1$. In Eq. (15), it suffices to update the dimensions i of b^k where $W_{i,i} = 1$. The minimization of Eq. (13) leads to the iterative procedure [16]

$$z^{l+1} = (I + sD^{\mathrm{T}}D)^{-1} \left(W \left(\tilde{y} - z^l\right) + z^l\right), \tag{16}$$

where $\tilde{y} = y + d^k - b^k$. Then, similarly to Eq. (10), it becomes

$$z^{l+1} = \mathrm{DCT}^{-1} \left(\Gamma \, \mathrm{DCT} \left(W \left(\tilde{y} - z^l\right) + z^l\right)\right). \tag{17}$$

In order to achieve optimal efficiency [9], we perform only one of these iterations in each split-Bregman iteration.

2.2. Handling multidimensional data

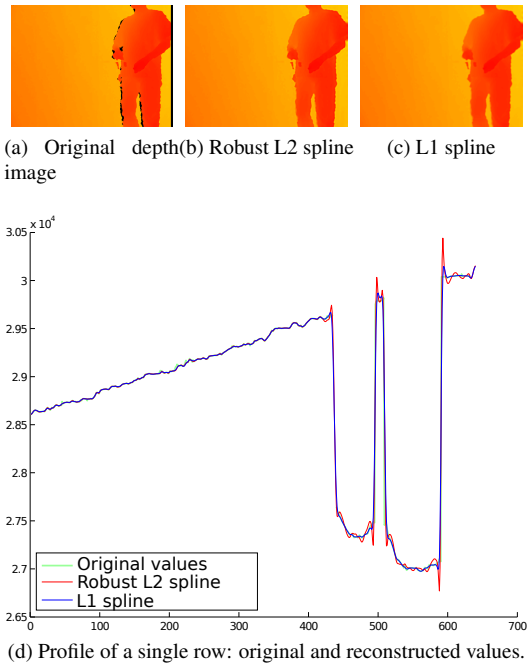
Let us now return to the general case of m-dimensional data. Following Buckley [15], we extend Eq. (10) as

$$\hat{y} = \mathrm{DCT}_m^{-1} \left(\Gamma^m \circ \mathrm{DCT}_m(y)\right), \tag{18}$$

where $\mathrm{DCT}_m(\cdot)$ and $\mathrm{DCT}_m^{-1}(\cdot)$ stand for the m-dimensional DCT-II and inverse DCT-II functions, and $\circ$ denotes the Schur (element-wise) product. Notice that the multidimensional DCT is simply a composition of one-dimensional DCTs along each dimension. Γ^m is an m-th order tensor defined by $\Gamma^m = 1^m \div (1^m + s\Lambda^m \circ \Lambda^m)$, where 1^m is an m-th order tensor of ones, and $\div$ denotes the element-wise division. Finally, Λ^m is an m-th order tensor, defined by

$$\Lambda^m_{i_1,\ldots,i_m} = \sum_{j=1}^{m} \left(-2 + 2\cos\left((i_j - 1)\pi/n_j\right)\right). \tag{19}$$

where n_j denotes the size of Λ^m along the j-th dimension.



(a) Original depth image (b) Robust L2 spline (c) L1 spline

(d) Profile of a single row: original and reconstructed values.

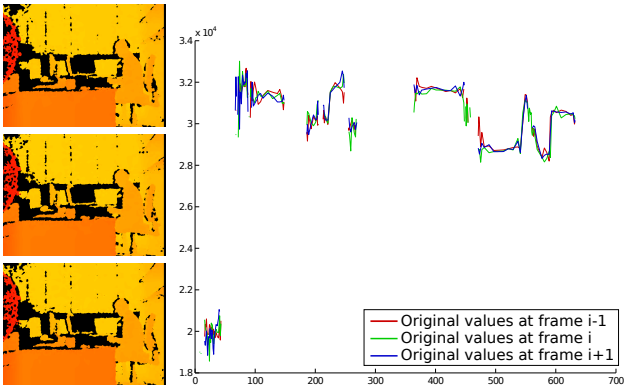
Fig. 3. Smoothing and interpolating using a single depth frame. We fit a 2D spline to the depth image. Since the data presents several "jumps," the robust L2 spline must undersmooth the data to be able to fit it correctly. The L1 spline presents a good trade-off between fitting and smoothing.

3. EXPERIMENTAL RESULTS

For all experiments we adhere to the following setup. Using generalized cross validation, we find the best estimate $\hat{s}$ for s for the robust L2 formulation [16]. We then find L2 splines (Eq. (4)), robust L2 splines [16], and/or L1 splines (Eq. (1)), setting $s = \hat{s}$. This protocol allows us to show that, even when s is chosen to fit optimally the robust L2 formulation, the proposed method provides better estimates. For the L1 formulation, we simply set $\lambda = 1$ for all examples. We set the number of split-Bregman iterations to a hundred.

In Fig. 1 we present a two-dimensional example. We depict the original signal $\hat{y} \in [1, \ldots, 256]^2 \to [-6.5497, 8.1054]$ in Fig. 1a, and we add two types of noise: first, Gaussian noise r_1 with zero mean and variance $\sigma^2 = 2$ (Fig. 1b), and then uniform noise r_2 in the interval $[-5 \cdot \max(\hat{y} + r_1), \ldots, 5 \cdot \max(\hat{y} + r_1)]$ (Fig. 1c). Only the L1 spline correctly recovers the original signal.

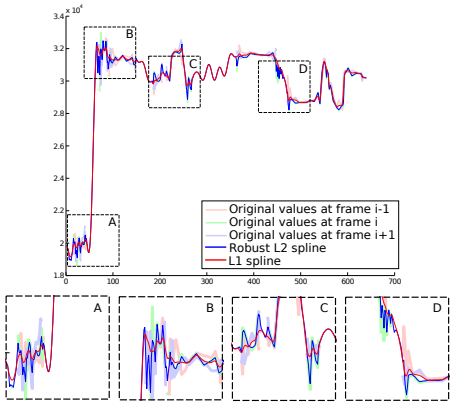
We also test in a synthetic example the ability to recover signals with sharp transitions, see Fig. 2. In this case we use a simple piece-wise constant function. We can observe clear overshoot (plus ringing) effects on the L2 and robust L2 splines. The robust L2 spline also results in transitions with less vertical slopes, creating a bluring effect. With the L1 spline we obtain a much better reconstruction, with almost non-existent overshooting.



(a) Depth frames $i-1$, i, and $i+1$ and profile of a single row.

(b) Robust L2 spline

(c) L1 spline



(d) Profile of a single row. Left, original values; right, original and reconstructed values.

Fig. 4. Smoothing and interpolating using multiple frames. To process frame i, we build a $640 \times 480 \times 3$ tensor using frames $i-1$, i, and $i+1$, which we then smooth/interpolate with a 3D robust L2 or L1 spline. Since the data presents several "jumps," the robust L2 spline must under-smooth the data to be able to fit it correctly. The L1 spline presents a good trade-off between fitting and smoothing.

We now perform smoothing of depth data obtained with a Kinect camera. This kind of data is particularly challenging for several reasons. First, it presents relatively smooth areas separated by sharp transitions. Second, edges are highly noisy, that is, edge pixels oscillate over time between foreground and background. Lastly, it contains missing data, which appear for two different reasons: (1) the disparity between the IR projector and the IR camera produces "shadows," and (2) the depth cannot be recovered in areas where the IR pattern is not clearly observable (e.g., because they receive direct sunlight or interference from another Kinect). We use splines to interpolate and denoise these data, showing the advantage of L1 splines over its robust L2 counterpart. The displayed images are part of the LIRIS human activities dataset [17].

In the first example, shown in Fig. 3, we use a single depth frame (with standard Kinect resolution of 640×480). The missing data are represented in black, while depth data goes from red to yellow as depth increases. Both, the L1 spline and the robust L2 spline are able to interpolate the missing data with reasonable values. Notice, however, that the latter exhibits, as aforementioned, overshooting and ringing (clearly perceived in the 1D profile). These effects are much milder in the L1 reconstruction.

Table 1. Execution times (in seconds) and number of iterations until convergence of the proposed algorithm.

	Size	Robust L2 spline	L1 spline	
		Time	Time	Iters.
Fig. 1	256×256	0.541	0.982	100
Fig. 3	480×640	0.911	0.266	2
Fig. 4	$480 \times 640 \times 3$	7.511	2.202	3

In Fig. 4a we can clearly observe that the position of the missing data is not consistent across frames. We can integrate data from several frames to achieve more accurate interpolations, by performing 3D reconstructions. Thus, in this example, we treat depth data as a 3D signal (2D + time), by considering three consecutive frames. The data dimensionality is then $640 \times 480 \times 3$. A full depth video can be smoothed by using 3D splines as a sliding-window type of filter. The robust L2 spline again presents a noisier behavior and with significative overshooting. On the other hand, the L1 spline is much smoother in smooth areas while correctly preserving abrupt transitions.

Running times. We present in Table 1 the running-time and number of iterations until convergence for every example in this work. All code is written in pure Matlab, with no C++ nor mex optimizations. All experiments were run on a MacBook Pro with a 2.7GHz Intel Core i7 processor. The time of the robust L2 and the L1 splines is comparable. Finally, note that in most cases the algorithm converges in very few iterations (here convergence means that $\left\|z^{k+1}-z^{k}\right\|_2 / \left\|z^{k}\right\|_2 < 10^{-3}$). In the example in Fig. 1, the maximum number of iterations (100) is reached with a final error of $10^{-2.8}$.

4. CONCLUSIONS

We have presented a new method for robustly smoothing regularly sampled data. We do this with modified splines, where we replace the classical L2-norm in the fitting term by an L1-norm. This automatically handles outliers, thus obtaining a robust approximation.

We also presented a new technique, using split-Bregman iteration, for solving the resulting optimization problem. The algorithm is extremely simple and easy to code. The method converges very quickly and has a small memory footprint. It also makes extensive use of the DCT, thus being straightforward to implement in GPU. These characteristics make this method very suitable for large-scale problems.

5. REFERENCES

[1] M. Nikolova, "Minimizers of cost-functions involving nonsmooth data-fidelity terms. application to the processing of outliers," *SIAM J. Numer. Anal.*, vol. 40, no. 3, pp. 965–994, Mar. 2002.

[2] S. Alliney, "A property of the minimum vectors of a regularizing functional defined by means of the absolute norm," *IEEE Trans. Signal Process.*, vol. 45, no. 4, pp. 913–917, Apr. 1997.

[3] T. F. Chan and S. Esedoğlu, "Aspects of total variation regularized L1 function approximation," *SIAM J. Appl. Math.*, vol. 65, no. 5, pp. 1817–1837, July 2005.

[4] J. F. Aujol, G. Gilboa, T. Chan, and S. Osher, "Structure-texture image decomposition-modeling, algorithms, and parameter selection," *Int. J. Comput. Vision*, vol. 67, no. 1, pp. 111–136, Apr. 2006.

[5] M. Nikolova, M. K. Ng, and C. P. Tam, "On $\ell 1$ data fitting and concave regularization for image recovery," Tech. Rep., École Normale Supérieure de Cachan, Cachan, France, Mar. 2012.

[6] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime TV-L1 optical flow," in *DAGM*, Berlin, Heidelberg, 2007, pp. 214–223, Springer-Verlag.

[7] A. Wedel, T. Pock, C. Zach, H. Bischof, and D. Cremers, "An improved algorithm for TV-L1 optical flow," in *Statistical and Geometrical Approaches to Visual Motion Analysis*, D. Cremers, B. Rosenhahn, A. L. Yuille, and F. R. Schmidt, Eds., vol. 5064, pp. 23–45. Springer-Verlag, 2009.

[8] Lars L. Rakêt, Lars Roholm, Mads Nielsen, and François Lauze, "TV-L1 optical flow for vector valued images," in *EMMCVPR*, 2011, pp. 329–343.

[9] T. Goldstein and S. Osher, "The split Bregman method for L1-regularized problems," *SIAM J. Img. Sci.*, vol. 2, no. 2, pp. 323–343, Apr. 2009.

[10] P. L. Combettes and J. C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, H. H. Bauschke, R. S. Burachik, P. L. Combettes, Elser, D. R. Luke, and H. Wolkowicz, Eds., pp. 185–212. Springer, May 2011.

[11] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin, "An iterative regularization method for total variation-based image restoration," *Multiscale Model. Simul.*, vol. 4, no. 2, pp. 460–489, 2005.

[12] Jinjun Xu and S. Osher, "Iterative regularization and nonlinear inverse scale space applied to Wavelet-Based denoising," *IEEE Trans. Image Process.*, vol. 16, no. 2, pp. 534–544, Feb. 2007.

[13] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, "Bregman iterative algorithms for L1-minimization with applications to compressed sensing," *SIAM J. Imaging Sci.*, vol. 1, no. 1, pp. 143–168, 2008.

[14] Tom Goldstein, Xavier Bresson, and Stanley Osher, "Geometric applications of the split Bregman method: Segmentation and surface reconstruction," *J. Sci. Comput.*, vol. 45, no. 1, pp. 272–293, Oct. 2010.

[15] M. J. Buckley, "Fast computation of a discretized thin-plate smoothing spline for image data," *Biometrika*, vol. 81, no. 2, pp. 247–258, June 1994.

[16] D. Garcia, "Robust smoothing of gridded data in one and higher dimensions with missing values," *Comput Stat Data Anal*, vol. 54, no. 4, pp. 1167–1178, Apr. 2010.

[17] C. Wolf, J. Mille, L. E. Lombardi, O. Celiktutan, M. Jiu, M. Baccouche, E. Dellandrea, C. E. Bichot, C. Garcia, and B. Sankur, "The LIRIS human activities dataset and the ICPR 2012 human activities recognition and localization competition," Tech. Rep. RR-LIRIS-2012-004, LIRIS Laboratory, Mar. 2012.