TRANSFORM DOMAIN SPARSIFICATION OF DEPTH MAPS USING ITERATIVE QUADRATIC PROGRAMMING

Gene Cheung[#], Junichi Ishida[o], Akira Kubota[o], Antonio Ortega[*]

[#] National Institute of Informatics, [o] Chuo University, [*] University of Southern California

ABSTRACT

Compression of depth maps is important for "texture plus depth" format of multiview images, which enables synthesis of novel intermediate views via depth-image-based rendering (DIBR) at decoder. Previous depth map coding schemes exploit unique depth data characteristics to compactly and faithfully reproduce the original signal. In contrast, since depth map is only a means to the end of view synthesis and not itself viewed, in this paper we explicitly manipulate depth values, without causing severe synthesized view distortion, in order to maximize representation sparsity in the transform domain for compression gain—we call this process transform domain sparsification (TDS). Specifically, for each pixel in the depth map, we first define a quadratic penalty function, with minimum at ground truth depth value, based on synthesized view's distortion sensitivity to the pixel's depth value during DIBR. We then define an objective for a depth signal in a block as a weighted sum of: i) signal's sparsity in the transform domain, and ii) per-pixel synthesized view distortion penalties for the chosen signal. Given that sparsity (l_0-norm) is non-convex and difficult to optimize, we replace the l_0-norm in the objective with a computationally inexpensive weighted l_2-norm; the optimization is then an unconstrained quadratic program, solvable via a set of linear equations. For the weighted l_2-norm to promote sparsity, we solve the optimization iteratively, where at each iteration weights are readjusted to mimic sparsity-promoting l_τ-norm, $0 \leq \tau \leq 1$. Using JPEG as an example transform codec, we show that our TDS approach gained up to 1.7dB in rate-distortion performance for the interpolated view over compression of unaltered depth maps.

Index Terms— Depth-image-based rendering, transform coding, sparse representation

1. INTRODUCTION

Continuing cost reduction of consumer-level cameras means images and videos previously taken by one camera from a single viewpoint can now be captured economically by an array of cameras to record multiple viewpoints. If, in addition to captured texture maps (RGB images), depth maps (per-pixel physical distances between camera and locations of the scene's objects in 3D space) are also available[1], then novel intermediate views can also be synthesized via *depth-image-based-rendering* (DIBR) techniques such as 3D warping [1], using neighboring texture and depth maps as anchors. Having both captured and seemingly unlimited intermediate views available at the client can translate to richer visual experiences such as free viewpoint TV [2], where the client can interactively select any desired viewpoint of the scene of interest for observation. However, encoding and transmitting texture and depth maps of a large number of captured views—a format called *texture-plus-depth* [3]—can incur a high transmission cost. One practical necessity for texture-plus-depth format then, is efficient encoding of depth maps.

Recent efforts to encode depth maps [4] exploit depth signal's unique characteristics, such as smooth surfaces and sharp edges, for efficient compression. However, the goal in these approaches is to reconstruct a signal $\hat{\mathbf{s}}$ as close as possible to the original $\mathbf{s}$ for a given coding rate. In contrast, given that a depth map is a means to synthesize intermediate views and not itself observed, we remark that *one can explicitly manipulate depth values without directly causing signal degradation as observed by users*, as long as the manipulation does not lead to severe synthesized view distortions. In fact, it has been shown [5] that in low texture regions of a scene, errors in depth have limited effect on the synthesized views. In this paper, we propose to exploit this degree of freedom to manipulate depth values (to some controlled extent) to maximize representation sparsity in transform domain for compression gain, a process we call *transform domain sparsification* (TDS).

An orthogonal transform coder maps a signal $\mathbf{s} \in \mathcal{R}^N$ to a set of N pre-defined basis functions ϕ_i's spanning the same signal space $\mathcal{R}^N$ of dimension N. In other words, a given signal $\mathbf{s}$ in $\mathcal{R}^N$ can be written as a linear sum of those basis functions using coefficients α_i's:

$$\mathbf{s} = \sum_{i=1}^{N} \alpha_i \phi_i \tag{1}$$

Only non-zero quantized transform coefficients $\hat{\alpha}_i$'s are encoded and transmitted to the receiver for reconstruction of approximate signal $\hat{\mathbf{s}}$. α_i's are obtained using a complementary set of basis functions $\bar{\phi}_i$'s; i.e., $\alpha_i = \langle \mathbf{s}, \bar{\phi}_i \rangle$, where $\langle \mathbf{x}, \mathbf{y} \rangle$ denotes a well defined inner product between two signals $\mathbf{x}$ and $\mathbf{y}$ in Hilbert space $\mathcal{R}^N$.

Compression efficiency of transform coding depends to a large extent on the *representation sparsity* of signal $\mathbf{s}$ in the transform domain; i.e., the number of zero quantized coefficients $\hat{\alpha}_i$'s. Much effort in transform coding is spent on finding basis functions ϕ_i's that maximize representation sparsity for a class of signals. In the case of depth map encoding, we solve the complementary problem: given a set of orthogonal basis functions ϕ_i's, we "sparsify" the signal $\mathbf{s}$ in transform domain, optimally trading off its representation sparsity and its adverse effect on synthesized view distortion via DIBR.

In particular, we perform sparsification of depth map as follows. For each depth pixel in a code block, we first define a *quadratic penalty function*, where larger deviation from its nadir (ground truth depth value) leads to a larger penalty. Synthesized view's distortion sensitivity to the pixel's depth value determines the sharpness of the constructed parabola. We then define an objective for a depth signal in a code block as a weighted sum of: i) signal's sparsity in the transform domain, and ii) per-pixel synthesized view distortion penalties for chosen signal. Given that sparsity (l_0-norm) is non-convex and difficult to optimize, we replace the l_0-norm in the objective with a computationally inexpensive weighted l_2-norm; the optimization

[1] Depth maps can be estimated from texture maps, or captured explicitly using time-of-flight cameras.

is then an unconstrained quadratic program, solvable via a set of linear equations. For the weighted l_2-norm to promote sparsity, we solve the optimization iteratively, where at each iteration weights are readjusted to mimic sparsity-promoting l_τ-norm, $0 \leq \tau \leq 1$. Using JPEG as an example transform codec, we show that our TDS approach gained up to 1.7dB in rate-distortion performance for the interpolated view over compression of unaltered depth maps.

The outline of the paper is as follows. We first overview related work in Section 2 and discuss how we define penalty functions per depth pixel in Section 3. We discuss our chosen objective and formulate our optimization in Section 4. We then present our sparsity-promoting iterative quadratic algorithm in Section 5. Results and conclusion are presented in Section 6 and 7, respectively.

2. RELATED WORK

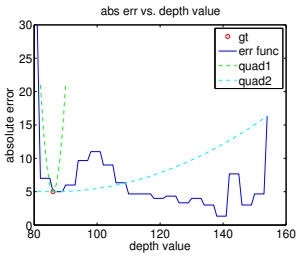
Compression for depth maps has previously been investigated [4], but the goal there was to reconstruct the original signal (depth map) faithfully, while in this paper we explicitly manipulate the signal (without causing severe synthesized view distortion) for compression gain. A similar recent work is [5], where the authors analyzed how compression error in depth values can lead to distortion in synthesized views, and proposed a new metric for mode selection in H.264 encoding of depth maps. We differ in that we explicitly sparsify the depth signal in transform domain for coding gain.

[6] also presented a signal manipulation problem, where given the constraint that code blocks must fall within their assigned quantization bins of the compressed image, high frequency components across block boundaries are eliminated via projection on convex sets (POCS). [7] discussed *near-lossless image compression*, where any given pixel value can have an error of no more than $\pm v$ during compression. More recently, [8] proposed to trade off signal quality with l_1-norm of the transform coefficients for coding gain in a lapped biorthogonal transform setting. Our work differs from these previous work in two respects: i) we define one penalty function per depth pixel to reflect the unique sensitivity of synthesized view distortion to the pixel value; and ii) we use iterative weighted l_2-norm minimization to promote sparsity in the transform domain, which can be efficiently solved compared to l_1-norm minimization.

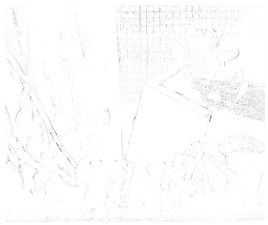
In our earlier work [9], we have studied the depth map sparsification problem, where *don't care regions* (DCR) were first defined per-pixel restricting the search space of depth signals, then weighted l_1-norm was minimized iteratively to find a sparse representation in the transform domain. The problem setting in [9] is a restricted one, however, where a single disparity map at the middle viewpoint location of the desired synthesized view is encoded. [9] further assumes ground truth texture map at the same middle viewpoint is available to construct DCR. In this paper, we consider the more realistic scenario where left and right depth maps need to be encoded for view synthesis at any location in between. Further, only left and right texture maps at those same anchor locations are available to construct our per-pixel penalty functions. Finally, we propose iterative quadratic programs to find sparse solutions, which is far more computationally efficient than iterative weighted l_1-norm minimization in [9].

3. DEFINING PENALTY FUNCTIONS

A pixel $I_l(m,n)$ in the left texture map can be mapped to a shifted pixel $I_r(m, n - D_l(m,n) * \gamma)$ in the right texture map, where $D_l(m,n)$ is the disparity value in the left depth map corresponding to left texture pixel $I_l(m,n)$, and γ is the camera-shift scaling factor for this camera setup. To derive synthesized view's distortion sensitivity to left depth pixel $D_l(m,n)$, we define error function $E_l(k;m,n)$ given depth error e: it is the difference in texture



(a) Per-pixel penalty function



(b) Penalty curvature for `Teddy`

Fig. 1. Error and quadratic penalty functions constructed for one pixel in right view (view 6), and curvature of penalty functions for entire right view in `Teddy`.

pixel values between left pixel $I_l(m,n)$, and incorrectly mapped right pixel $I_r(m, n - (D_l(m,n) + e) * \gamma)$ due to error e. We write:

$$E_l(e;m,n) = |I_l(m,n) - I_r\,(m, n - (D_l(m,n) + e) * \gamma)\,| \quad (2)$$

Error function $E_r(e;m,n)$ for the right depth map can be derived similarly. As an example, the blue curve in Fig. 1(a) is the resulting $E_r(e;m,n)$ for the right view (view 6) of multiview sequence `Teddy` [10]. One can see that, in general, as the depth value deviates from the ground truth disparity value $D_r(m,n)$ (red circle), the error increases.

For mathematical convenience (so that the to-be-discussed optimization can be formulated as an unconstrained quadratic program), we fit a per-pixel quadratic penalty function $g_i(s_i)$ to the error function and use $g_i(s_i)$ instead:

$$g_i(s_i) = (1/2)a_i s_i^2 + b_i s_i + c_i \quad (3)$$

where s_i is the disparity value corresponding to pixel location i, and a_i, b_i and c_i are the quadratic function parameters. The procedure we use to fit $g_i(s_i)$ to the error function is as follows. Given threshold ρ, we first seek the nearest disparity $D_l(m,n) - e$ value below ground truth $D_l(m,n)$ that results in error $E_l(-e;m,n)$ exceeding $\rho + E_l(0;m,n)$. Using only two data points at $D_l(m,n) - e$ and $D_l(m,n)$, and assuming $g_i(s_i)$ has minimum at ground truth depth value $D_l(m,n)$, we can construct one quadratic function. Similar procedure is applied to construct another penalty function using two data points at $D_l(m,n)+e$ and $D_l(m,n)$ instead. The sharper of the two constructed functions (larger a) is the chosen penalty function for this pixel.

Continuing with our earlier example, we see in Fig. 1(a) that two quadratic functions (in dashed lines) with minimum at ground truth depth value are constructed. The narrower of the two is chosen as the penalty function. In Fig. 1(b), the per-pixel curvature (parameter a) of the penalty functions of the right depth depth of `Teddy` is shown. We can clearly see that larger curvatures (larger penalties) occur at object boundaries, agreeing with our intuition that a synthesized view is more sensitive to depth pixels at object boundaries.

4. PROBLEM FORMULATION

Given the defined per-pixel penalty functions, we now formulate our depth signal sparsification problem TDS—trading off representation sparsity in the transform domain with synthesized view distortion—as iterative unconstrained quadratic programs.

4.1. Defining Objective Function

Let $\mathbf{s}$ be a depth signal in the pixel domain, where s_i is the depth value of the i-th pixel. Let Φ be a chosen orthogonal transform.

Hence $\alpha = \Phi \mathbf{s}$ is the transform coefficient vector. Since only non-zero quantized transform coefficients are coded, in general the fewer the number of non-zero entries in α, the better the compression performance. The number of non-zero entries in α is equivalent to the l_0-norm $\|\alpha\|_{l_0}$ of α:

$$\|\alpha\|_{l_0} = |\{i|\alpha_i > 0\}| \tag{4}$$

Using the per-pixel penalty functions $g_i(s_i)$'s we defined (in the pixel domain) in the previous section, we can now write our objective function for TDS as a sum of l_0-norm and weighted penalties of coefficient vector α (in the transform domain):

$$\min_{\alpha} \ \|\alpha\|_{l_0} + \lambda \sum_i g_i(\phi_i^{-1}\alpha) \tag{5}$$

where ϕ_i^{-1} is the i-th row of the inverse transform Φ^{-1}, and λ is a weight parameter to control the relative importance of representation sparsity and synthesized view distortion of the sought solution.

(5) is an unconstrained optimization problem where the optimization variables are transform coefficients α_i's. Because of the combinatorial nature of l_0-norm $\|\alpha\|_{l_0}$ in (5), finding the optimal solution to (5) is non-convex and NP-hard in general. We hence next propose alternative functional to serve as surrogate to the original objective (5) that can be more efficiently solved.

4.2. Weighted l_2-norm Surrogate

Instead of minimizing l_0-norm $\|\alpha\|_{l_0}$ directly, numerous previous approaches seek the minimum l_1-norm solution instead [11]. More generally, the weighted l_p-norm $\|\alpha\|_{l_p(\mathbf{w})}$ of coefficient vector α with weight vector $\mathbf{w}$ is defined as:

$$\|\alpha\|_{l_p(\mathbf{w})} = \left(\sum_i w_i |\alpha_i|^p \right)^{1/p} \qquad 0 < p < \infty \tag{6}$$

It was observed empirically that in many practical settings, the l_1-norm promotes sparsity, and that under a restricted condition on α [12], the minimum l_1-norm solution is also the minimum l_0-norm solution. l_1-norm minimization can be formulated as a linear program (LP) and solved via standard LP techniques.

Though LP is solvable in polynomial time, it remains computationally expensive. [12] and others have showed that[2] if a l_1-norm minimizing α^* has no vanishing component (i.e., no $\alpha_i^* = 0$) and each weight w_i of the weighted l_2-norm is assigned $1/|\alpha_i^*|$, then minimum weighted l_2-norm solution α^w coincides with the minimum l_1-norm solution α^*. It seems possible then, that if weights w_i's can be appropriately selected a priori, then one can replace the conventional l_1-norm with a computationally inexpensive weighted l_2-norm and still promotes sparsity.

More specifically, replacing l_0-norm $\|\alpha\|_{l_0}$ in (5) with the weighted l_2-norm $\|\alpha\|_{l_2(\mathbf{w})}$ of coefficient vector α, we have a new functional to serve as surrogate for the original TDS objective:

$$\min_{\alpha} \ \sum_i w_i \alpha_i^2 + \lambda \sum_i g_i(\phi_i^{-1}\alpha) \tag{7}$$

which can be solved very efficiently by rewriting it in the form of an unconstrained quadratic program:

$$\min_{\alpha} \ (1/2)\alpha^T \mathbf{P} \alpha + \mathbf{q}^T \alpha + r \tag{8}$$

where the constants $\mathbf{P}$, $\mathbf{q}$ and r are:

[2][12] also showed convergence of weighted l_2-norm to l_1-norm when there are vanishing components for their iterative algorithm.

1. Initialize weights $w_i = 1/(|\alpha_i^t| + \epsilon)^2$, where α_i^t is the i-th transform coefficient of the ground truth depth signal $\mathbf{s}^t$.
2. Find optimal α^o to (7) for weighted l_2-norm minimization with penalties by solving (10).
3. Set each weight w_i to $(|\alpha_i^o|^2 + \epsilon^2)^{-\frac{2-\tau}{2}}$ if $|\frac{\alpha_i^o}{Q_i}| \geq 0.5$, and $\epsilon^{\frac{4-2\tau}{2}}$ otherwise.
4. Repeat Step 2 to 3 until convergence in quantized coefficients $\hat{\alpha}_i^o$'s.

Fig. 2. Iterative algorithm to solve weighted l_2-norm minimization with penalty functions $g_i(s_i)$'s.

$$\mathbf{P} = \begin{bmatrix} 2w_1 & 0 & \dots & \\ 0 & 2w_2 & 0 & \dots \\ & & & \\ 0 & \dots & \ddots & \end{bmatrix} + \lambda \sum_i a_i \left(\phi_i^{-1}\right)^T \phi_i^{-1}$$

$$\mathbf{q}^T = \lambda \sum_i b_i \phi_i^{-1} \qquad r = \lambda \sum_i c_i \tag{9}$$

The optimal solution α^o to (8) can be easily found by solving the following set of linear equations [13]:

$$\mathbf{P}\alpha^o = -\mathbf{q} \tag{10}$$

One important question remains: how to define appropriate weights w_i's for weighted l_2-norm in (7) to promote sparsity? We discuss this issue next.

5. ITERATIVE QUADRATIC PROGRAMS

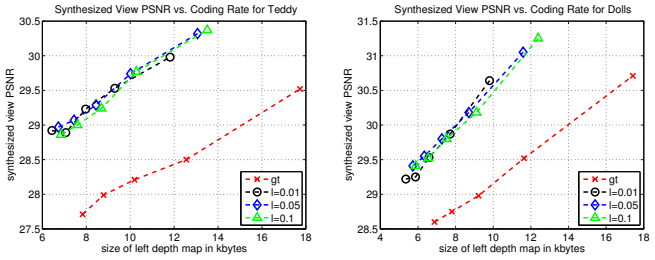
To find appropriate weights w_i's for (7) to promote sparsity, we design an algorithm, inspired by the *iterative re-weighted least squares* (IRLS) work in [12], where (7) is solved iteratively, each time with weights w_i's readjusted. The idea of IRLS is that in an iterative algorithm, one can assume previous iteration's solution α^o serves as a good estimate to optimal solution α^*. Hence, one can define the i-th component weight w_i as $1/|\alpha_i^o|$, so that the iteratively weighted l_2-norm can mimic the sparsity promoting l_1-norm. Similarly then, when solving (7) in an iteration in our algorithm, we also readjust weights w_i's using previous solution α^o. Fig. 2 shows how we adopt IRLS for our optimization in (7).

A few details in Fig. 2 deserve further explanation. First, because assigning $w_i = 1/|\alpha_i^o|$ assumes solution α^o to (7) is already reasonably close to optimal α^*, we need to provide a good initial guess to initialize w_i's. In our optimization, we simply use transform coefficients α_i^t's of ground truth depth signal $\mathbf{s}^t$ as initial guess.

Second, if α_i^o is close to zero, $1/\alpha_i^o$ approaches infinity. To avoid numerical stability problems, a parameter $\epsilon > 0$ is added when assigning weight w_i in Step 3, as done also in [11] and [12]. Experiments show that results are not sensitive to the value of ϵ.

Third, because only *quantized* coefficients $\hat{\alpha}_i = round(\alpha_i/Q_i)$ are encoded, where Q_i is the quantization parameter for coefficient α_i, a coefficient α_i^o with value $|\alpha^o/Q_i| < 0.5$ will not be coded and should not be counted. We hence set α_i^o to zero when updating w_i so that the contribution of this component $w_i\alpha_i^2$ is close to zero.

Finally, if we set τ in the exponent of Step 3 to 1, then each weight w_i approaches $1/|\alpha_i^o|$ and the iterative re-weighting mimics l_1-norm. More generally, τ can be set in Step 3 to value $0 < \tau \leq 1$, however, to mimic l_τ-norm instead. As shown in [12], the advantage of using $\tau < 1$ is that the local convergence rate can be superlinear,



(a) Teddy (b) Dolls

Fig. 3. PSNR Comparison between sparsified representations and original depth maps for Teddy and Dolls.

speeding up the iterative algorithm. The disadvantage is that it may converge to a wrong local minimum if initial guess is too far from optimal solution. Because we do have a good initial guess α^t available, we can afford smaller τ for faster convergence with small risk of wrong convergence. In the experiments, we will test different values of τ.

6. EXPERIMENTATION

6.1. Experimental Setup

To test the Rate-distortion (RD) performance of our TDS approach, we used an implementation of JPEG [14], cjpeg version 8a, for image compression of left and right depth maps in multiview image sequences Teddy and Dolls [10]. In brief, cjpeg performs 2D Discrete Cosine Transform (DCT) on non-overlapping 8×8 pixel blocks, and quantizes the resulting DCT coefficients according to a *quality* setting. Lower *quality* maps to coarser quantization of DCT coefficients.

For a given *quality* setting, we fed the corresponding quantization matrix $\mathbf{Q}$ into our iterative quadratic minimization algorithm, together with defined per-pixel penalty functions and weight parameter λ. The sparsified representations for left and right depth maps are compressed using cjpeg at this *quality* setting, and uncompressed back to pixel domain for view synthesis of the middle image between left and right views. View synthesis is performed using a simple in-house implementation of 3D warping [1] to copy and blend corresponding left and right pixels to the synthesized view. Holes are filled using nearest horizontal synthesized pixels. *quality* setting from 50 to 90 and weight parameter λ ranging from 0.01 to 0.1 were used in our experiment. Sparsified representations were compared against unaltered left and right ground truth depth maps compressed using cjpeg at the same *quality* settings.

6.2. Experimental Results

First, we tried running our algorithm for different τ's so that the weighted l_2-norm in our iterative quadratic algorithm mimics sparsity promoting l_τ-norm. We found that changing τ made no noticeable difference in the converged solutions. Because there is a convergence speedup when τ is set small, we used $\tau = 0$ for the rest of the experiments.

In Fig. 3(a) we see the RD performance for Teddy—quality of middle synthesized view in Peak Signal-to-noise Ratio (PSNR) versus size of the left compressed depth map—using our sparsified representations for different weight λ and ground truth depth maps. We see that there is an optimum λ that induces the right tradeoff between representation sparsity in the transform domain and synthesized view distortion, so that the RD performance is maximized. We found this to be $\lambda = 0.05$ experimentally. In particular, our sparse representations offered up to 1.7dB gain over ground truth depth maps.

Fig. 3(b) show the same performance comparison for Dolls. Sparsified representations outperformed compression of unaltered ground truth depth map by up to 1.5dB in this case.

7. CONCLUSION

Efficient depth map coding is important in texture-plus-depth format for depth-image-based rendering (DIBR) at the decoder. In this paper, we sparsify representation of the depth signal in the transform domain (for coding gain) while inducing minimal synthesized view distortion. We first define our objective function of a depth signal as a sum of l_0-norm and weighted per-pixel penalty functions. We then replace l_0-norm with a weighted l_2-norm, so that the optimization can be solved efficiently. The optimization is then solved iteratively, and weights are adjusted in each iteration so that the weighted l_2-norm mimic a sparsity promoting l_τ-norm. Results show that our sparsified representations outperformed compression of unaltered depth maps by up to 1.7dB in synthesized view PSNR.

8. REFERENCES

[1] W. Mark, L. McMillan, and G. Bishop, "Post-rendering 3D warping," in *Symposium on Interactive 3D Graphics*, New York, NY, April 1997.

[2] A. Kubota, A. Smolic, M. Magnor, M. Tanimoto, T. Chen, and C. Zhang, "Multi-view imaging and 3DTV," in *IEEE Signal Processing Magazine*, November 2007, vol. 24, no.6.

[3] P. Merkle, A. Smolic, K. Mueller, and T. Wiegand, "Multi-view video plus depth representation and coding," in *IEEE International Conference on Image Processing*, San Antonio, TX, October 2007.

[4] M. Maitre, Y. Shinagawa, and M.N. Do, "Wavelet-based joint estimation and encoding of depth-image-based representations for free-viewpoint rendering," in *IEEE Transactions on Image Processing*, June 2008, vol. 17, no.6, pp. 946–957.

[5] W.-S. Kim, A. Ortega, P. Lai, D. Tian, and C. Gomila, "Depth map distortion analysis for view rendering and depth coding," in *IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009.

[6] R. Rosenholtz and A. Zakhor, "Iterative procedures for reduction of blocking effects in transform image coding," in *IEEE Transactions on Circuits and Systems for Video Technology*, March 1992, vol. 2, no.1, pp. 91–95.

[7] R. Ansari, N. Memon, and E. Ceran, "Near-lossless image compression techniques," in *Journal of Electronic Imaging*, July 1998, vol. 7, no.3.

[8] M. Winken, D. Marpe, and T. Wiegand, "Global and local rate-distortion optimization for lapped biorthogonal transform coding," in *IEEE International Conference on Image Processing*, Hong Kong, September 2010.

[9] G. Cheung, A. Kubota, and A. Ortega, "Sparse representation of depth maps for efficient transform coding," in *28th Picture Coding Symposium*, Nagoya, Japan, December 2009.

[10] "2006 stereo datasets," http://vision.middlebury.edu/stereo/data/scenes2006/.

[11] E. J. Candes, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted l_1 minimization," in *The Journal of Fourier Analysis and Applications*, December 2008, vol. 14, no.5, pp. 877–905.

[12] I. Daubechies, R. Devore, M. Fornasier, and S. Gunturk, "Iteratively reweighted least squares minimization for sparse recovery," in *Commun. Pure Appl. Math*, 2009.

[13] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge, 2004.

[14] "Independent JPEG group," http://www.ijp.org/.