

SPARSE REPRESENTATION OF DEPTH MAPS FOR EFFICIENT TRANSFORM CODING

Gene Cheung [#], Akira Kubota ^o, Antonio Ortega ⁺

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

ABSTRACT

Compression of depth maps is important for “image plus depth” representation 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 characteristics to compactly and faithfully reproduce the original signal. In contrast, given that depth maps are not directly viewed but are only used for view synthesis, in this paper we manipulate depth values themselves, without causing severe synthesized view distortion, in order to maximize sparsity in the transform domain for compression gain. We formulate the sparsity maximization problem as an l_0 -norm optimization. Given l_0 -norm optimization is hard in general, we first find a sparse representation by iteratively solving a weighted l_1 minimization via linear programming (LP). We then design a heuristic to push resulting LP solution away from constraint boundaries to avoid quantization errors. Using JPEG as an example transform codec, we show that our approach gained up to 2.5dB in rate-distortion performance for the interpolated view.

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. While multiview images/videos can potentially provide richer user experiences such as free viewpoint TV [1] and immersive teleconferencing, encoding and transmitting multiple views incurs a high transmission cost. A recent approach for efficient multiview representation is *image/video plus depth* [2], where, in addition to texture maps (images), depth maps of different viewpoints are also encoded and transmitted, so that a desired intermediate view can be synthesized at the receiver via depth-image-based rendering (DIBR) techniques like 3D warping, using texture and depth maps of neighboring viewpoints. A key to image-plus-depth representation then, is efficient encoding of depth maps.

Recent efforts to encode depth maps [3] exploit depth signal’s unique characteristics, such as smooth surfaces and sharp edges, for efficient compression. Like generic coding schemes like JPEG, however, the goal is nonetheless to reconstruct a signal $\hat{\mathbf{s}}$ as faithfully as possible to the original \mathbf{s} for a given coding rate. In contrast, given that a depth map is for view synthesis only but not direct observation, we remark that *one can 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 [4] that in low texture regions of a scene, depth value of a pixel can vary to some extent with little ill effect to the synthesized views. In this paper, we propose to exploit this degree of freedom to manipulate depth values (to some defined extent) to maximize representation sparsity in transform domain for

compression gain.

A biorthogonal 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 a_i ’s:

$$\mathbf{s} = \sum_{i=1}^N a_i \phi_i \quad (1)$$

Only transform coefficients a_i ’s are encoded and transmitted to the receiver for reconstruction of signal \mathbf{s} . a_i ’s are obtained using a complementary set of basis functions $\bar{\phi}_i$ ’s:

$$a_i = \langle \mathbf{s}, \bar{\phi}_i \rangle \quad (2)$$

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 a transform coder depends to a very large extent on the *sparsity* of representation of signal \mathbf{s} in the transform domain; i.e., the number of zero coefficients a_i ’s in (1). Much effort in transform coding is spent on finding basis functions ϕ_i ’s that maximize representation sparsity for a given class of signals. In the case of depth map encoding, we solve the complementary problem: given a set of biorthogonal basis functions ϕ_i ’s, we find the signal (depth map) \mathbf{s} within a well defined subspace that maximizes sparsity of its representation in the transform domain.

In particular, we perform sparsity maximization for depth map encoding as follows. For a given code block, we first define a range of depth values called a *don’t care region* (DCR) for each pixel in the block, where any depth value within range will lead to a synthesized view distortion of this pixel by no more than a threshold value T . We then formulate the sparsity maximization problem as an l_0 -norm optimization problem. Given finding l_0 -norm is hard in general, we propose to iteratively solve weighted l_1 minimization until convergence (each can be solved via linear programming (LP)), similarly to the approach in [5]. Finally, to avoid rounding errors during quantization, we design a heuristic to push the LP-derived sparse solution to another one in the interior of DCR. Experimental results show that our discovered sparse representations can improve standard JPEG compression of original depth maps by up to 2.5dB in rate-distortion performance for the interpolated view.

The outline of the paper is as follows. We first overview related work in Section 2 and discuss how DCRs are derived in Section 3. Formulation and solution to the sparsity maximization problem are presented in Section 4. Results and conclusion are presented in Section 5 and 6, respectively.

2. RELATED WORK

Compression specifically for depth maps has been previously investigated [3], but the goal there was to reconstruct the original signal (depth map) faithfully, while in this paper we manipulate the signal

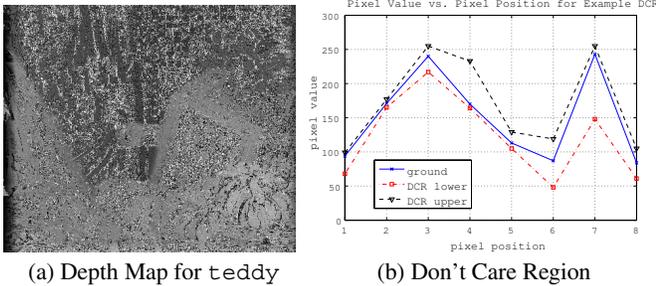


Fig. 1. Depth Map \mathbf{d}_{mid} and Don't Care Region for the first pixel row in a 8×8 block for `teddy` at $T = 7$.

itself (without causing severe synthesized view distortion) for compression gain. A similar recent work is [4], 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 maximize representation sparsity in transform domain for coding gain.

[6] also presented a signal manipulation problem, where given the original image block must fall within assigned quantization bins of the compressed image, the block without high frequency components across block boundaries is searched via projection on convex sets (POCS). [7] discussed *near-lossless image compression*, where any given pixel value cannot be erred by more than $\pm v$ values 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 is different from these previous work in two respects: i) the DCR definition allows each depth pixel to have its own range for value manipulation; and ii) our objective to maximize sparsity in transform domain leads to a challenging l_0 -norm minimization problem (number of non-zero coefficients) which reflects coding rate more accurately than l_1 -norm.

3. DERIVATION OF DON'T CARE REGIONS

We first discuss how DCRs are derived. Each DCR, specified in pixel domain, defines the search space of depth maps in which a sparse representation in transform domain is sought.

We assume a simple setting where we are given three texture images, \mathbf{I}_{left} , \mathbf{I}_{mid} and \mathbf{I}_{right} , captured by a horizontally shifted camera, where \mathbf{I}_{mid} is at the strict middle of \mathbf{I}_{left} to the left and \mathbf{I}_{right} to the right. A texture image \mathbf{I}'_{mid} is synthesized using \mathbf{I}_{left} , \mathbf{I}_{right} and depth image \mathbf{d}_{mid} . In particular, each pixel $I'_{mid}(i, j; d)$ at spatial location (i, j) is a weighted average of two corresponding horizontally shifted pixels in \mathbf{I}_{left} and \mathbf{I}_{right} given depth¹ value d at (i, j) :

$$I'_{mid}(i, j; d) = \frac{1}{2}I_{left}(i + d, j) + \frac{1}{2}I_{right}(i - d, j) \quad (3)$$

Ground truth depth value $d_{mid}(i, j)$ is one that minimizes the resulting error $e(i, j; d) = |I'_{mid}(i, j; d) - I_{mid}(i, j)|$ between synthesized pixel $I'_{mid}(i, j; d)$ and captured pixel $I_{mid}(i, j)$, i.e.:

$$d_{min}(i, j) = \arg \min_d e(i, j; d) \quad (4)$$

¹The more precise terminology for pixel displacement by d in (3) should be *disparity* instead of *depth*. Since depth is simply the inverse of disparity, we assume encoding and manipulation of depth maps is equivalent.

Given the above definition, we can now determine a DCR at pixel (i, j) for threshold T as follows: we find the minimum $f(i, j)$ and maximum $g(i, j)$ depth values such that the resulting error $e(i, j; d)$ does not exceed T :

$$\begin{aligned} f(i, j) &= \min \{ d \} \quad \text{s.t. } e(i, j; d) \leq T \\ g(i, j) &= \max \{ d \} \quad \text{s.t. } e(i, j; d) \leq T \end{aligned} \quad (5)$$

See Fig. 1(a) for an example of a depth map \mathbf{d}_{mid} for image sequence `teddy` [9] and Fig. 1(b) for an example of ground truth depth $d_{mid}(i, j)$ (blue), DCR lower and upper bounds $f(i, j)$ (red) and $g(i, j)$ (black) for a pixel row in a 8×8 block in `teddy`. Notice that by construction, $f(i, j) \leq d_{min}(i, j) \leq g(i, j)$.

In general, a larger threshold T offers a larger subspace for an algorithm to search for sparse representations in transform domain leading to compression gain, at the expense of larger resulting synthesized distortion. We examine in Section 5 how the optimal T can be located experimentally.

4. FINDING SPARSE REPRESENTATION

Given a defined *pixel-level DCR* described in Section 3, a depth pixel $s(m)$ at location m must be within range $f(m)$ and $g(m)$, i.e., $f(m) \leq s(m) \leq g(m)$, for the resulting synthesized view distortion not to exceed threshold T . Consequently, *block-level DCR*, \mathcal{B} , is simply a concatenation of pixel-level DCRs for all pixel locations in a code block. In other words, a signal \mathbf{s} is in \mathcal{B} if all its pixel components $s(m)$ fall within the permissible bounds:

$$\mathbf{s} \in \mathcal{B} \text{ iff } f(m) \leq s(m) \leq g(m), \quad \forall s(m) \in \mathbf{s} \quad (6)$$

Given a well-defined block-level DCR \mathcal{B} , our goal is to find signal $\mathbf{s} \in \mathcal{B}$ such that sparsity of its representation using basis ϕ_i 's in the transform domain is maximized. More precisely, given matrix Φ containing basis functions ϕ_i 's as rows:

$$\Phi = \begin{bmatrix} \leftarrow \phi_1^T \rightarrow \\ \vdots \\ \leftarrow \phi_N^T \rightarrow \end{bmatrix}, \quad (7)$$

our sparsity optimization can be written as follows:

$$\min_{\mathbf{s} \in \mathcal{R}} \|\mathbf{a}\|_{l_0} \quad \text{s.t. } \mathbf{a} = \Phi \mathbf{s} \quad (8)$$

where $\mathbf{a} = [a_1, \dots, a_N]$ are the transform coefficients and $\|\mathbf{a}\|_{l_0}$ is the l_0 -norm, essentially counting the number of non-zero coefficients in \mathbf{a} :

$$\|\mathbf{a}\|_{l_0} = |\{i : a_i \neq 0\}| \quad (9)$$

4.1. Defining Weighted l_1 Minimization

Minimizing l_0 -norm in (8)—which is combinatorial in nature—is in general difficult. An alternative approach, as discussed in [5], is to iteratively solve a weighted version of corresponding l_1 -norm minimization instead:

$$\min_{\mathbf{s} \in \mathcal{R}} \|\mathbf{a}\|_{l_1^w} \quad \text{s.t. } \mathbf{a} = \Phi \mathbf{s} \quad (10)$$

where l_1^w -norm sums up all weighted coefficients in \mathbf{a} :

$$\|\mathbf{a}\|_{l_1^w} = \sum_i w_i |a_i| \quad (11)$$

1. Initialize weights $w_i = 1$ for all i 's.
2. Solve l_1 -norm minimization (10) for solution a_i 's.
3. Set weights $w_i = 1/(|a_i| + \epsilon)$ for all i 's.
4. Repeat Step 2 and 3 until convergence.

Fig. 2. Iterative algorithm to solve weighted l_1 minimization

It is clear that if weights $w_i = 1/|a_i|$, then weighted l_1^w -norm is the same as l_0 -norm, and an optimal solution to (10) is also optimal to (8). Clearly, one cannot know weights $1/|a_i|$'s *a priori* when a_i 's are the unknown variables in optimization (10). Hence [5] proposed an iterative algorithm so that solution a_i 's to previous iteration of (10) is used as weights for optimization in the current iteration. See Fig. 2 for a pseudo-code of the iterative procedure. ϵ in step 3 is used to avoid posing an ill-conditioned problem when a a_i is very small.

4.2. Solving Weighted l_1 Minimization

(10) can be solved as a linear program (LP)² via a known method like Simplex [10]. For our purpose, however, we need to first take into consideration of transform coefficient quantization, since what we are seeking is to maximize the number of zero *quantized* transform coefficients. Without loss of generality, we assume a transform coefficient a_i is quantized by factor Q_i before coding, i.e., $\text{round}(a_i/Q_i)$ is encoded instead. Hence a_i is encoded as non-zero coefficient only if $a_i/Q_i \geq 0.5$. Instead of optimizing a_i 's directly, we first introduce a set of new variables, *shrinkage coefficients* α_i 's, where α_i is the absolute value of a_i/Q_i less 0.5 if $a_i/Q_i \geq 0.5$, and zero otherwise:

$$\alpha_i = \begin{cases} \left| \frac{a_i}{Q_i} \right| - 0.5 & \text{if } \left| \frac{a_i}{Q_i} \right| \geq 0.5 \\ 0 & \text{o.w.} \end{cases} \quad (12)$$

$$= \max \left\{ \left| \frac{a_i}{Q_i} \right| - 0.5, 0 \right\} \quad (13)$$

We subsequently minimize $\sum_i w_i \alpha_i$'s instead. This way, coefficients a_i 's that are quantized to zero will not be counted in the objective function.

Clearly (13) is not linear. To linearize (13), we write the following inequalities, which is equivalent to (13) when α_i 's are minimized:

$$\alpha_i \geq \frac{a_i}{Q_i} - 0.5 \quad (14)$$

$$\alpha_i \geq -\frac{a_i}{Q_i} - 0.5, \quad \alpha_i \geq 0$$

We are now ready to formulate our problem as LP as follows. The linear objective is to minimize $\sum_i w_i \alpha_i$. The linear constraints are: i) signal \mathbf{s} must fall inside defined DCR according to (6); ii) a_i 's are unquantized transform coefficients of signal \mathbf{s} according (10); and iii) α_i 's, as shrinkage coefficients, obey inequalities in (14). Using algorithm in Fig. 2 to iteratively solve this LP, we will arrive at a signal \mathbf{s} with sparse representation in the transform domain.

4.3. Quantization Effects on DCR

Because non-zero quantized coefficients are rounded to the nearest integer, the rounding operation can force the computed signal \mathbf{s} to a quantized $Q(\mathbf{s})$ —one that is just as sparse as the intended \mathbf{s} —that is

²Note that we can reduce LP complexity by keeping all zero coefficients a_i 's of ground truth depth \mathbf{d}_{mid} at zero when searching for a *more* sparse signal \mathbf{s} , reducing the number of non-zero a_i 's that needs to be searched.

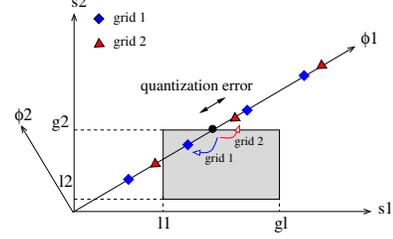


Fig. 3. Quantization Effects on Sparse Representation

outside DCR. This is particularly problematic when LP is formulated to find a sparse solution, since LP solvers like Simplex [10] typically find optimal solutions by iteratively searching among vertices of a polytope defined by the linear constraints of the problem, meaning the discovered solution is always at boundaries of DCR. As an illustration, consider Fig. 3, where an LP solver finds a 1-sparse solution (black dot) that is at the boundary of DCR (grey box). Rounding LP solution to the blue square using grid 1 below will keep the quantized solution inside DCR, while rounding the LP solution to the blue square above will force the quantized solution outside DCR.

To find a quantized sparse solution within DCR as much as possible, we perform the following heuristic after a sparse pre-quantized solution \mathbf{s} has been found by the iterative algorithm in Fig. 2. We perform one more LP, where the constraints are: i) the three linear constraints as described in previous LP formulation; and ii) shrinkage coefficients α_i 's that correspond to zero quantized coefficients, $\text{round}(a_i/Q_i) < 0.5$, in pre-quantized solution \mathbf{s} are kept at zero to maintain the same sparsity. The objective is to find a new pre-quantized solution \mathbf{s}' with same sparsity, such that it is furthest away from any DCR boundary possible. While such strategy does not guarantee new quantized $Q(\mathbf{s}')$ to be inside DCR (e.g., if grid 2 is used for quantization in Fig. 3, no sparse quantized solution is inside DCR), moving a solution \mathbf{s} to a solution \mathbf{s}' far from DCR boundaries does give \mathbf{s}' a higher chance of remaining inside DCR after quantization to $Q(\mathbf{s}')$. In practice, we compare quantized $Q(\mathbf{s})$ of original pre-quantized solution \mathbf{s} , and quantized $Q(\mathbf{s}')$ of the modified solution, \mathbf{s}' , and count the number of pixels in each that violate pixel DCR. We select one that has the fewer number of pixel DCR violations as our chosen sparse representation.

5. EXPERIMENTATION

5.1. Experimental Setup

To test the RD performance of sparse representations found using our algorithm, we used a free implementation of JPEG from Independent JPEG Group [11], cjpeg version 8a, for image compression of a depth map from image sequences teddy and bull [9]. In brief, cjpeg divides an image into 8×8 pixel blocks, performs 2D Discrete Cosine Transform (DCT), and quantizes the resulting DCT coefficients for run-length coding in a zigzag pattern.

cjpeg offers a *quality* setting, from 1 (low) to 100 (high), to adjust compression ratio of an image. *quality* is mapped to a *scale* parameter, which is used to scale a default quantization matrix Q for DCT coefficient quantization. In particular, the following procedure is performed to map *quality* setting to a quantization parameter $Q(i, j)$.

1. $scale := -2 * quality + 200$.
2. $Q(i, j) := \lfloor \frac{Q(i, j) * scale + 50}{100} \rfloor$.
3. $Q(i, j) := \max \{ 1, Q(i, j) \}$.

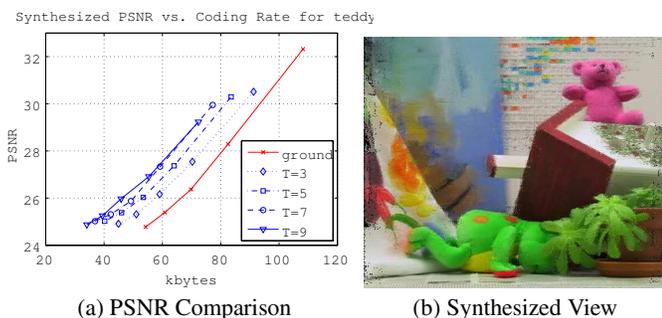


Fig. 4. PSNR Comparison between proposed Sparse Representations and Ground Truth Depth Maps, and Synthesized View for $T = 7$, $quality = 70$ for *teddy*.

$$4. Q(i, j) := \min \{ 32767, Q(i, j) \}.$$

For a given *quality* setting, we fed the corresponding quantization matrix \mathbf{Q} into our sparse representation search algorithm as previously discussed, together with a defined DCR (for a given threshold T). The computed sparse representation is compressed using *cjpeg* at this *quality* setting, and uncompressed back to pixel domain for view synthesis of the middle image \mathbf{I}'_{mid} as described in Section 3. *quality* setting from 50 to 90 and threshold T ranging from 3 to 9 for *teddy* and from 1 to 5 for *bull* were used in our experiment. Computed sparse representations were compared against the ground truth depth map \mathbf{d}_{mid} compressed using *cjpeg* at the same *quality* settings.

5.2. Experimental Results

For the *teddy* sequence, in Fig. 4(a) we see the rate-distortion (RD) performance—quality of synthesized view \mathbf{I}'_{mid} in Peak Signal-to-noise Ratio (PSNR) versus size of the compressed depth map—of our computed sparse representations for different threshold value T and the ground truth depth map \mathbf{d}_{mid} . We see that as T increased, RD performance of our sparse solutions improved. In particular, at 70 kbytes, our sparse solution at $T = 7$ offered more than 2.5dB gain over \mathbf{d}_{mid} . We see also that as T increased from 7 to 9, the RD curve is almost exactly the same. This shows the law of diminishing return: the increase in freedom to explore sparser representations in a larger DCR no longer offers a strictly beneficial RD gain, given the cost in synthesized distortion has become significant.

In Fig. 4(b), we see synthesized view \mathbf{I}'_{mid} of *teddy* when sparse solution is found using $T = 7$ and compressed at *quality* = 70. As shown, there are no annoying visual artifacts unique to our use of sparse representations that were not already present when ground truth depth maps were used.

In Fig. 5(a), we see the RD performance of our discovered sparse representations and the ground truth depth signal for the *bull* sequence. Because the texture maps are inherently more complex than *teddy*, DCR for given threshold value tends to be smaller, and the PSNR gain of our scheme over ground truth is not as large; maximum PSNR gain in this case is about 1.5dB. The synthesized view for $T = 5$ and *quality* = 70 is shown in Fig. 5(b). Again, we notice no disturbing visual artifacts unique to our sparse representations.

6. CONCLUSION

In this paper, observing that depth maps are for view synthesis only and not for direct observations by users, we first constructed don't

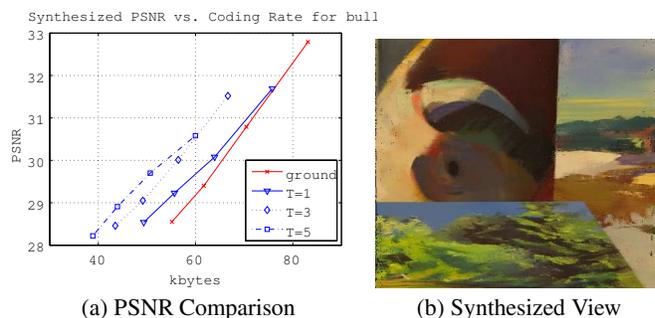


Fig. 5. PSNR Comparison between proposed Sparse Representations and Ground Truth Depth Maps, and Synthesized View for $T = 5$, $quality = 70$ for *bull*.

care region (DCR), where one can manipulate depth values within DCR without causing severe distortion in the synthesized view. Exploiting the freedom to pick any signal within DCR, we seek sparse representations in the transform domain for coding gain via iterative weighted l_1 minimization. Our scheme achieved up to 2.5dB gain over compression of the ground truth depth map in JPEG.

7. ACKNOWLEDGMENT

The authors thank Junichi Ishida for his assistance in collecting experimental data for this paper.

8. REFERENCES

- [1] T. Fujii and M. Tanimoto, "Free viewpoint TV system based on ray-space representation," in *Proceedings of SPIE*, 2002, vol. 4864, p. 175.
- [2] 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.
- [3] 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.
- [4] 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.
- [5] 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.
- [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] "2006 stereo datasets," <http://vision.middlebury.edu/stereo/data/scenes2006/>.
- [10] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Dover, 1998.
- [11] "Independent JPEG group," <http://www.ijp.org/>.