

# Resolution-sensitive document image analysis for document repurposing

Kathrin Berkner and Edward L. Schwartz  
Ricoh Innovations, Inc.  
2882 Sand Hill Rd., Suite 115  
Menlo Park, CA 94025  
{berkner, schwartz}@rii.ricoh.com

**Abstract**—The variety of displays used to browse and view images has created a need to adapt an image representation to the size constraints of a given display. In this paper a resolution-sensitive analysis of document images is performed that segments images into text and image regions using document layout analysis and JPEG 2000 header-based processing. Repurposing of the document for viewing on a given display is performed by automatic scaling, cropping, and rearranging of document segments to produce a pictorial document representation while maintaining readability of text and recognizability of image features.

## I. INTRODUCTION

With the ever increasing amount of digital images, optimizing the information content of an image is important. A traditional problem is to optimize for a bit-rate determined by storage or bandwidth constraints. Given a digital image and a bit-rate, the available bits are allocated to provide the best possible image viewed at full resolution, where 'best' is measured in some distortion metric. Similarly, the number of bits spent to achieve a particular distortion is a measure for the information contained in the image. Fundamental results in rate-distortion theory combined with Moore's Law driven availability of low cost computation have enabled the development of practical, sophisticated image coders over the last decade, such as JPEG 2000 [1]. Given state-of-art image coders and future trends of decreasing storage and bandwidth cost, bit-rate optimization improvements will most likely provide diminishing returns in the future.

In this paper, instead of bit-rate optimization, a different direction of optimizing the information content of an image is chosen, namely optimizing for viewing on a given display. Constraints come from display sizes, e.g. notebook, PDA and cellphone displays, or small-size screen areas in desktop applications. Given a fixed display size and a high resolution image, the image information needs to be allocated to the available display in order to convey information for an intended purpose. This problem is of particular importance for document images.

Compared to digital photographic images (e.g. natural scenes, portraits, frames from video), scanned document images have the distinct characteristics of high resolution and heterogeneous content that lead to important requirements for image processing algorithms.

Document images are typically scanned at print resolution, e.g. from 2550x3300 pixels for 300 dpi up to 20400x26400

pixels for 2400 dpi. Due to the large data size and range of dpi resolutions, algorithms have to be efficient for real-time processing as well as adaptive to changes in resolution.

Documents typically contain text and non-text areas. Non-text areas can be separated into several categories, e.g. photo, table, graphics, white-space, margins, background, etc. Different properties of the content categories require different processing. One well-known example is compression of text versus photo content. Keeping text readable is a challenge for quantization. For example, JPEG or JPEG 2000 artifacts make high frequency image regions more pleasing to the human eye, whereas quantization artifacts in text areas may destroy the purpose of those regions, namely displaying readable material. Solutions to this problem are provided by segmenting a document image and applying different compression schemes to the individual segments (e.g. [2]). In addition, text has structural characteristics that are different from those of generic photos. There are logical interpretations of groups of pixels as characters, words, lines, paragraphs, title, etc. including their hierarchical relationships. An important feature is also the reading order which may be language dependent. These logical characteristics of text affect algorithm design.

Traditionally, documents were created for a specific use by a target audience, e.g. a journal article was created for reading on paper by a scientific community. Digital documents now exist in an environment with almost unlimited storage capabilities, internet connections and new communication tools, such as cell phone cameras, powerpoint presentations, etc. This environment drives a new trend: Document images being used for a different purpose than they were originally designed for. Examples are display of documents at several resolutions, such as 600 dpi print resolution and 72 dpi monitor resolution, the use of document images for retrieval purposes, or using images of presentation slides to provide a link between logical and non-logical time-based digital data [3]. Following this trend, the need for new document processing is developing, focusing on *repurposing* of documents.

In this paper a specific repurposing application is targeted: Viewing documents — originally designed for being viewed printed on paper — on small size displays. Whereas existing approaches are designed mainly for web documents and utilize text-based analysis only ([4], [5], [6]), the techniques presented in the following sections will consider all the algorithmic

challenges introduced earlier in this section: text and image specific processing (Section II), repurposing step (Section III), efficiency (Section IV), and adaption to changes in document resolution (Section V). Examples of repurposing for various display sizes are given before the paper concludes in Section VI.

## II. RESOLUTION-SENSITIVE ANALYSIS OF DOCUMENT IMAGES

### A. Image-based document analysis

Creating a recognizable small-size image from an image-analysis point of view has the goal of identifying important segments of a high-resolution image and determining how much these segments can be reduced in size by scaling without losing the characteristics that made them important. Therefore, a resolution-sensitive localized importance measure is needed. In this paper, the newly standardized wavelet-based compression scheme JPEG 2000 [1] is used to extract such an importance measure.

The JPEG 2000 (J2K) image compression algorithm performs a wavelet transform on rectangular tiles and partitions the wavelet domain into groups of coefficients termed *code-blocks*. These code-blocks are encoded with an arithmetic coder and organized into one or more *layers*. Finally, all data are assembled into smaller unites called *packets*. *Packet headers* contain information such as number of bits allocated to each code-block. At low bit-rate, this information reflects the visual importance of a code-block. Due to the specific encoding scheme in the wavelet domain, a code-block is localized in the spatial and frequency domain. In [7] a *multiresolution bit distribution*  $B = B(r, (i, j))$  is derived from the packet headers. For each code-block location  $(i, j)$  at resolution  $r$  the value  $B(r, (i, j))$  denotes the number of bits spent by the encoder to encode the specific code-block area at resolution  $r$ . A Bayesian-type segmentation technique is applied to the bit distribution that yields a segmentation map grouping pixel blocks with similar *preferred resolution*

$$r^*(i, j) = \arg \min_r \sum_{\ell=r}^L w_{r,\ell} B(\ell, (i, j)), \quad (1)$$

where  $L$  denotes the maximal level of wavelet decomposition and  $w_{r,\ell}$  are specifically chosen weights (see also [7]). Figure 1 shows a multiresolution bit distribution for  $L = 5$  and a resolution-dependent segmentation map as the result of the segmentation process. Segmentation map values visualized with the same shade of gray have the same preferred resolution  $r^*$  from (1). The resolution-dependent segmentation map is computed from J2K headers only without any decoding of wavelet coefficients.

Next a connected component analysis is performed on the segmentation map. The connected components carry  $r^*$  as a *resolution attribute*. An *importance attribute* is also assigned to each component and is defined as the number of bits necessary to code the component area at resolution  $r^*$ . For the remainder of this paper, the components are called *segments*.

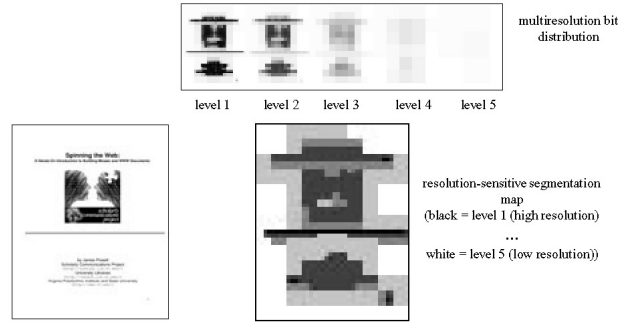


Fig. 1. Original document (left) and resolution-dependent segmentation map (right) derived from a multiresolution bit distribution (top).

The segments provide a coarse segmentation of a document into regions with different dominant frequencies. Due to the coarse code-block resolution and no specific text knowledge in the JPEG 2000 encoder, the segments may not completely capture text zones or align with them. On the other hand, image regions are well characterized.

### B. Text-based document analysis

From a text point of view, displaying text at a readable resolution is very important. A goal for creation of a meaningful small size image is to include as much important text as possible while keeping the text readable.

Initially, a document layout analysis is performed. Document layout analysis is typically part of state-of-the-art OCR systems (e.g. [8]). Bounding boxes for text zones and words are obtained from the layout analysis. This information provides a segmentation map for the text portions of a document including a hierarchy between segments and sub-segments. In this paper a text zone is called a *segment*. A sub-segment consists of individual words called *fragments*. A text segment is described by a rectangular bounding box. The aspect ratio of a segment can be changed by reflowing the fragments.

Next, a resolution and an importance attribute are assigned to each segment. A resolution attribute is derived from the minimal scaling factor that still keeps the text at a readable character size. The minimal readable character size is determined empirically for a given display device. In the experiments in this paper, a 72 dpi CRT monitor was used and a minimal readable character size of 6 pixels was determined. For each segment a resolution attribute is defined as the ratio of the *Minimal Readable Character Size (MRCS)* and the *Maximum Font Size* in a segment (MFS). An importance attribute is derived as a product of font-size and position of the text segment on the page. The position parameter  $w_{pos}$  puts more weighting on segments placed in the middle of the top of a page than on segments placed on the bottom of a page. Further details on the text attributes can be found in [9]. Table I summarizes resolution and importance attributes for image and text segments. While layout analysis provides generally

TABLE I

RESOLUTION AND IMPORTANCE ATTRIBUTES FOR DOCUMENT SEGMENTS

type(S)	resolution	importance
image	$1/r^*(S)$	$\sum_{(i,j) \in S} \sum_{r=r^*(S)}^L B(r, (i, j))$
text	$\frac{MRC S}{MFS(S)}$	$MFS(S) \cdot w_{pos}(S)$

a good segmentation into text zones, image regions are either not recognized or if recognized are not further characterized with respect to resolution and importance properties.

### C. Merging text- and image-based analysis results

The image- and text-based analysis results must be merged. To achieve this goal two steps are necessary. In a first step the different image- and text-based segmentation maps have to be combined into one segmentation map. In a second step the attributes associated with each image or text segment have to be unified. Before these steps are performed the reference grids for text segments (pixels) and image segments (code-blocks) have to be normalized.

Merging of segmentation maps is done by checking each image segment for significant overlap with text segments. In case of significant overlap, the text segments are subtracted from the image segment, and separating the remaining segments into new connected components. These components do not have any significant text content anymore.

Merging attributes by carrying the attributes from the individual analyses over to the merged segments is possible for the resolution attributes, but not for the importance attributes. Since importance attributes for image and text segments were measured via incompatible metrics (see Table I), it is necessary to define a unified importance measure for the segments in the merged segmentation map that reflects the importance of a single segment with respect to the entire collection of segments. The multiresolution bit distribution derived from the J2K header data is available for all regions in the image and is chosen as the base element for a unified importance measure. The importance value  $T(S) = MFS(S) \cdot w_{pos}(S)$  from the text-based analysis is added as a bonus to the bit measure of text segments. The unified importance measure is defined as

$$I(S) = \alpha \cdot \frac{I_{\text{bounding\_box}}(S, r^*(S))}{N_1} + \beta \cdot \frac{T(S)}{N_2}, \quad (2)$$

where  $N_1$  is the total entropy of the document image,  $N_2$  is the image size in pixels, and  $I_{\text{bounding\_box}}(S, r^*(S))$  measures the bit content inside the bounding box around the segment  $S$  at resolution  $r^*(S)$ . The parameters  $\alpha$  and  $\beta$  are introduced to allow flexible weighting of text and image regions. In this paper  $\alpha = 0$  and  $\beta = 1$  if  $\text{type}(S) = \text{text}$ , and  $\alpha = 1$  and  $\beta = 0$  if  $\text{type}(S) = \text{image}$ . Explicit details on the merging processing can be found in [9].

## III. RESOLUTION-SENSITIVE LAYOUT AS OPTIMIZATION PROBLEM

After creation of a list of text and image segments with importance and resolution attributes the next step is to lay those segments out in a display of a given size. This size may be determined by a specific application, e.g. a PDA display, or a thumbnail screen area.

In order to fit document segments in the final display three operations are allowed to be performed on the segments: scaling, cropping and positioning with respect to the reference location. This *repurposing step* includes a possible modification of the original image. Modifications can be made to resolution, inclusion, and layout of segments.

The quality of a possible layout as a combination of scaling, cropping and positioning of a collection of segments  $\{S_i\}$  is measured by a cost function  $Cost = Cost(\{S_i\}, \{s, B, p\})$ , where  $s$  is the amount of scaling,  $B$  is the set discarded in the cropping operation, and  $p$  denotes the positioning. This cost function determines the distortion of the composited image from the original document. The best fit is that vector of triplets  $\{(s_i^*, B_i^*, p_i^*)\}$  that is a feasible solution and results in the smallest distortion, i.e. results in the lowest cost

$$\{(s_i^*, B_i^*, p_i^*)\} = \arg \min_{\{s, B, p\}} Cost(\{S_i\}, \{s, B, p\}). \quad (3)$$

At this point the question of what cost function to choose becomes important. The cost should increase if more scaling has to be applied, i.e. with decreasing scale factor  $s$ . The cost should also increase if more pixels are discarded in the cropping operation, i.e. when the size of the set  $B$  increases. A change of relative positions from the original should also be penalized. The cost function used in this paper is of the following type

$$Cost(S) = 1/(\text{importance\_of\_}S\text{\_after\_scale\_and\_crop}) + \text{change\_in\_positioning\_of\_}S. \quad (4)$$

This layout problem is similar to the classical floor-plan problem [10], but differs since it allows cropping of segments in addition to scaling. A related problem is the layout for video key frames on paper as in [11], where the size of video frames is allowed to vary, but is determined by the shot-importance measure for key frames. The image size in [11] does not depend on the resolution properties of the image.

### A. Greedy algorithm

Solving the optimization problem as stated in (3) in general is very complex. Therefore, a greedy approach is chosen to trade off complexity versus optimization accuracy. The greedy approach fits the segments one by one into the available space. The following presumptions are applied.

- 1) Fitting order is segment with largest unified importance measure  $I$  from (2) first.
- 2) Use of the cropping operation is restricted to text segments only and removes fragments following the inverse reading order.
- 3) Cropping and scaling are mutually exclusive.

- 4) Relation between importance and scaling is  $importance\_after\_scaling = s \cdot I$ .
- 5) The relation between importance values of a segment  $S$  and its fragments  $F_1, \dots, F_m$  is  $I(S) = \sum_{k=1}^m I(F_k)$ .
- 6) Cropping and scaling are penalized equally.
- 7) For image and text segments a minimal scaling factor is set by  $s_{min}$ . A lower bound for cropping is given by requiring the importance of a cropped segment to be larger than a threshold  $\epsilon$ . In this paper, the bounds are set to  $s_{min} = \min(\frac{width(C)}{width(D)}, \frac{height(C)}{height(D)}) = s_c$  ( $D$  = original document,  $C$  = display canvas) and  $\epsilon = 0.3$ .

With these presumptions, the notation used to define the cost function is the following. A segment  $S$  is given by a geometric area  $A$  and an intensity function  $f_A$ , i.e.  $S = (A, f_A)$ . Its anchor point in the original document is denoted by  $q(S)$ . Scaling is defined as multiplication with a scalar, cropping of the area  $A$  by discarding the set  $B$  as  $A \setminus B$ . The position of an area  $A$  with respect to a reference point  $X$  is denoted by  $p(A, X)$ . A placed segment  $S$  with respect to the display canvas  $C$ , i.e.  $p(s(A \setminus B, f_A \setminus B), q(C))$  is denoted by  $\Lambda(S)$ . The metric  $d$  measures the Euclidean distance. Using these notations, the cost function is defined as follows.

$$Cost(S_i, s, B, p) = \begin{cases} \infty & \text{if (i), (ii), or (iii) is invalid;} \\ [(I(S_i) - I((A_i \setminus B, f_{A_i \setminus B}))) \cdot s \cdot I(S_i)]^{-1} + \\ d[s_c \cdot q(A_i \setminus B), p(s \cdot q((A_i \setminus B), \Lambda(S_{\sigma(k-1)})))] & \text{otherwise} \end{cases} \quad (5)$$

where  $S_{\sigma(k-1)}$  is the last segment that was placed before the current segment  $S_i$ , and conditions (i), (ii), (iii) are the feasibility conditions:

- (i)  $s > s_{min}$  (allowed scaling),
- (ii)  $I((A_i \setminus B, f_{A_i \setminus B})) > \epsilon$  (allowed cropping),
- (iii)  $[\bigcup_{k < i} \Lambda(S_{\sigma(k-1)})] \cap \Lambda(S_i) = \emptyset$  (available white space).

#### IV. IMPLEMENTATION UTILIZING J2K FEATURES

An image is compressed using several layers. The first layer contains image data compressed at 0.2 bpp and forms a *thin layer*. The other layers contain remaining desired image bits. For 300 dpi images five resolution levels were chosen for encoding, 600 dpi images were encoded with six levels.

Packet header data are extracted using a *packet header reader*. For 300 dpi images that reader extract headers for all J2K resolution levels one to five. For 600 dpi images only header data for the coarsest five out of six levels are extracted. That way the extracted information for 300 and 600 dpi images should be similar, and the following analysis step should have similar complexity for the two different dpi resolutions.

The image-based analysis program performs processing on luminance data only. Therefore, the complexity for color and grayscale images should be similar.

Results of text- and image-based analysis are merged, and the merged output is sent to the layout module which performs repurposing. At this point in the processing path, the dimensions of the final display size are necessary. The

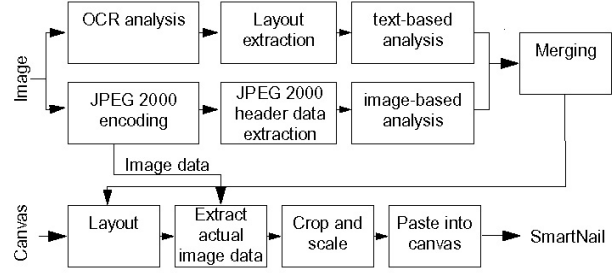


Fig. 2. System diagram showing complete SmartNail architecture.

computation of the final layout is performed on rectangular box description with attributes only. No access to image data is necessary at this point.

The first time actual image data are required is in the final composition step, that includes cropping and scaling of image data and pasting the results onto the final canvas. Scaling can be done in part by decoding of specific code blocks at specific resolutions of the J2K code stream. In case additional fine-scaling is required, a pixel-based scaling routine, e.g. *pnmscale* has to be applied. Cropping can be performed by region decoding or by a pixel-based cropping tool.

As a final step, pasting performed by a pixel-based pasting tool, is performed to place the cropped and scaled image into the final canvas. A diagram showing the composition of the individual modules into an entire system is shown in Figure 2.

#### V. RESULTS

This section starts with an example in order to explain the different analysis techniques and the layout as repurposing step derived in the previous sections. The example document from Figure 1 is being reformatted for various display sizes. Figure 3 shows the results applying the greedy layout algorithm optimizing the cost function from (5) for various display sizes. The final images are called *SmartNails*<sup>TM</sup> in contrast to traditionally scaled thumbnails. Reflow as a result of repositioning of text fragments and cropping of text lines by removal of fragments is clearly visible. While the analysis steps are the same for all SmartNails, the layout step incorporates the different display sizes. Text reflow based on bounding boxes of layout analysis was already introduced in [4] with a final layout based on reading order only, but not on combinations of scaling, cropping, and positioning of the text segments.

Table II shows relative computational effort in percentages for the different modules for three different document images at 300 and 600 dpi. Repurposing to fit a target size of 130x160 pixels was performed in the layout module. Compared to the effort of other modules, the layout modul, which performs the actual repurposing task, is efficient (0% means that the complexity was below 0.5%). Thumbnails and SmartNails of several tested images are shown in Figure 4. Table III shows a comparison between run times for 300 and 600 dpi images measured on a Sun Ultra 5 work station. Starting with

TABLE II

RUN TIME OF INDIVIDUAL MODULES FOR 300 AND 600 DPI IN PERCENT

	header reading (%)	image analysis (%)	text analysis (%)	merge (%)	layout (%) (repurposing)	image crop+scale+paste (%)
300 dpi						
im1	12	33	10	3	0	42
im2	3	22	18	4	1	52
im3	11	38	6	2	0	43
600 dpi						
im1	21	16	8	3	1	51
im2	16	15	15	3	0	51
im3	20	20	7	2	1	50

TABLE III

RUN TIMES FOR TRADITIONAL THUMBNAIL CREATION BY PIXEL-BASED SCALING AND J2K DECODING, AND SMARTNAIL CREATION (IN SECONDS)

	thumbnails				SmartNails	
	pixel-based scaling		J2K decoding		300 dpi	600 dpi
	300 dpi	600 dpi	300 dpi	600 dpi		
im1	1.56	6.15	0.26	0.63	3.47	4.22
im2	1.66	6.27	0.39	0.83	5.11	6.32
im3	1.56	5.92	0.49	0.70	4.51	5.20

J2K encoded data and layout analysis data, run times were measured using a research implementation that has not been fully optimized. The effort for 600 dpi resolution is only a little larger than that for 300 dpi resolution. For comparison the run times for two different ways of creation of traditional thumbnails are shown. One method applies a generic pixel-based scaling method (*pnmscale*) to the original image. The other method decodes that lower resolution image from the J2K codestream that is closest to the final thumbnail resolution and performs additional fine scaling with a pixel-based method.

The expected behavior  $run\_times\_for\_pixel\_based\_scaling \sim resolution^2$  is confirmed in the first two columns of Table III. Also expected is that using the low-resolution extraction feature of the J2K code stream has significantly better run time performance for simple thumbnail creation (column 3 and 4) than pixel-based scaling.

Whereas the numbers for the SmartNail algorithms at 300 dpi resolution are still larger than the pixel-based scaling, the increase with resolution is not as steep as for pixel-based scaling. The crossover point seems to be at 600 dpi images where SmartNail algorithms and pixel-based scaling are in the same range.

## VI. CONCLUSIONS

In this paper the image processing application of repurposing document images for viewing on a constrained display is discussed. A resolution-sensitive analysis combines image-based multiresolution analysis derived from header data of a JPEG 2000 code stream with text-based document layout analysis information. A layout step allows for simple repurposing of documents by recompositing important document segments at appropriate resolutions.

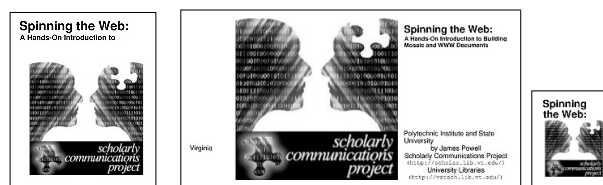


Fig. 3. Example for repurposing the document in Figure 1 for display sizes: 130x160 (left), 400x200 (middle), and 80x100 (right) pixels.

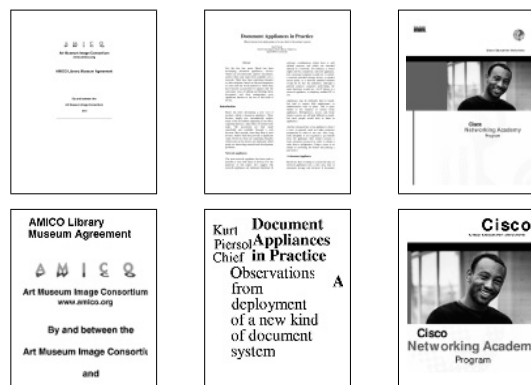


Fig. 4. Example images im1, im2, im3 (left to right): Smartnails for display size 130x160 (bottom) in comparison with thumbnails (top).

Low complexity of the proposed algorithms utilizing JPEG 2000 features allows acceptable processing time of print resolution images at various dpi while considering document specific text and image characteristics.

## REFERENCES

- [1] ITU-T Rec. T.800—ISO/IEC 15444-1:2000, *Information Technology – JPEG2000 Image Coding System*.
- [2] D. Mukherjee, C. Chrysafis, and A. Said, “JPEG-Matched MRC Compression of Compound Documents,” in *Proc. IEEE Int. Conf. Image Processing – ICIP ’02*, 2002, vol. 3, pp. 73–76.
- [3] B. Erol, J. J. Hull, and D.-S. Lee, “Linking Multimedia Presentations with their Symbolic Source Documents: Algorithm and Applications,” in *Proc. ACM MM*, 2003.
- [4] T. Breuel, W. Janssen, K. Popat, and H. S. Baird, “Paper to PDA,” in *Proc. Int. Conf. Pattern Recognition - ICPR 02*, 2002, pp. 476–479.
- [5] Alam et al., “Web page Summarization for Handheld Devices: A Natural Language Approach,” in *Proc. Int. Conf. Document Analysis and Recognition – ICAR ’03*, 2003, vol. 2, pp. 1153–1157.
- [6] A. Woodruff, A. Faulring, R. Rosenholtz, J. Morrison, and P. Pirolli, “Using Thumbnails to Search the Web,” in *Proc. of SIGCHI’01*, 2001.
- [7] R. Neelamani and K. Berkner, “Adaptive Representation of JPEG 2000 Images using Header-based Processing,” in *Proc. IEEE Int. Conf. Image Processing – ICIP ’02*, 2002, vol. 1, pp. 381–384.
- [8] G. Nagy, “Twenty years of document image analysis in pami,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 38–62, January 2000.
- [9] K. Berkner, E. L. Schwartz, and C. Marle, “SmartNails - Image and Display Dependent Thumbnails,” in *Proc. SPIE Int. Soc. Opt. Eng.*, 2004, to appear.
- [10] R. D. Meller and K.-Y. Gau, “The facility layout problem: Recent and emerging trends and perspectives,” *Journal of Manufacturing Systems*, vol. 15, no. 5, pp. 351–366, 1996.
- [11] S. Uchiyase and J. Foote, “Summarizing video using a shot importance measure and a framepacking algorithm,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing – ICASSP ’99*, 1999, pp. 3041–3044.