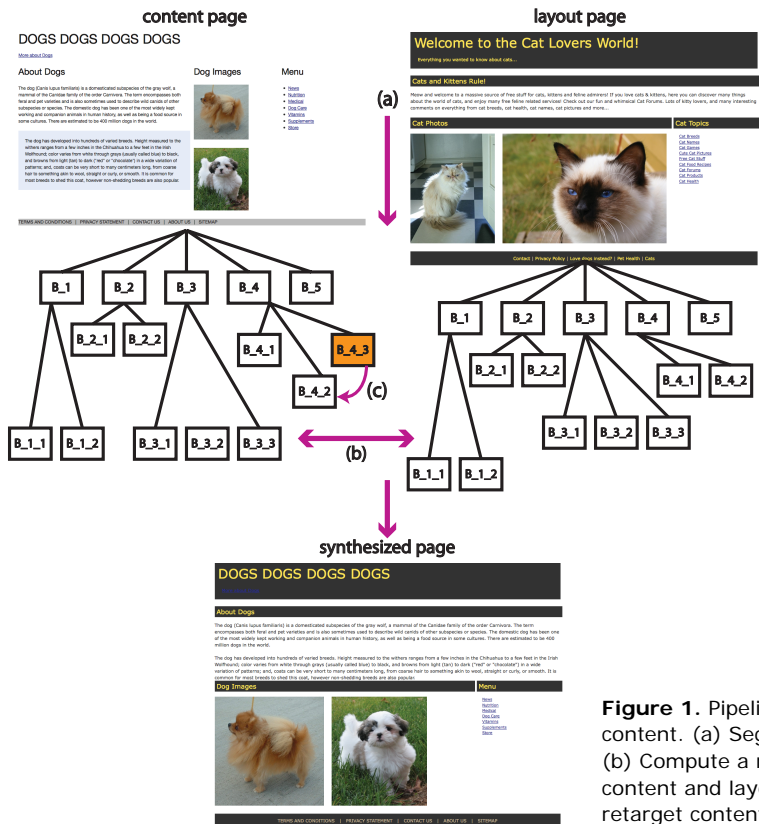


# Automatic Retargeting of Web Page Content



**Figure 1.** Pipeline for retargeting web page content. (a) Segment pages into perceptual trees. (b) Compute a reasonable mapping between content and layout trees. (c) Use mapping to retarget content onto the new layout.

**Ranjitha Kumar**  
 Stanford University HCI Group  
 353 Serra Mall  
 Stanford, CA 94305  
 ranju@stanford.edu

**Juho Kim**  
 Stanford University HCI Group  
 353 Serra Mall  
 Stanford, CA 94305  
 juhokim@stanford.edu

**Scott R Klemmer**  
 Stanford University HCI Group  
 353 Serra Mall  
 Stanford, CA 94305  
 srk@cs.stanford.edu

## Abstract

We present a novel technique for automatically retargeting content from one web page onto the layout of another. Web pages are decomposed into their perceptual hierarchical representations. We then use a structured-prediction algorithm to learn reasonable mappings between the perceptual trees. Using the mappings, we are able to merge the content of one page with the layout of another.

## Keywords

Web design, automatically generated alternatives

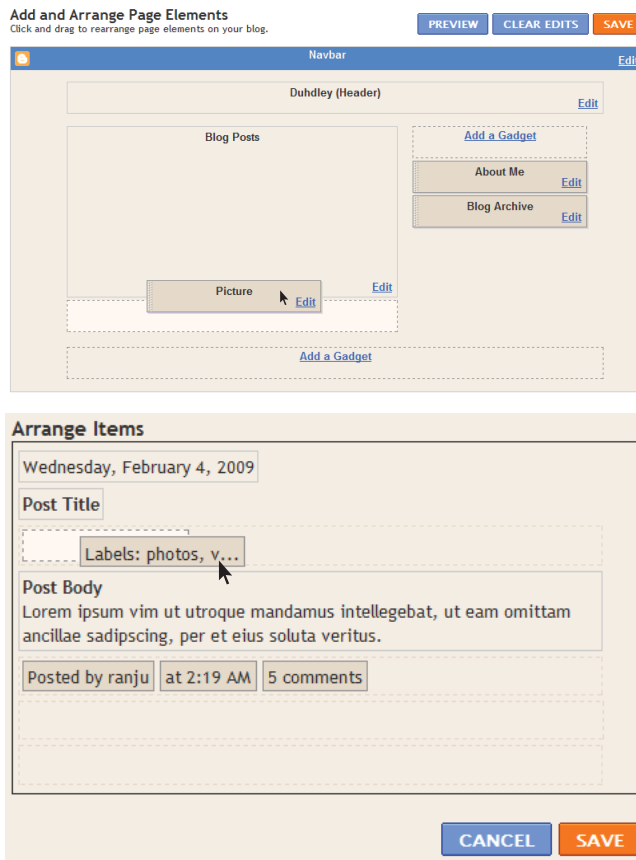
## ACM Classification Keywords

D.2.2 [Software Engineering]: Design Tools and Techniques.

## Introduction

Enumerating multiple alternatives is critical to good design [1] but is often tedious and time consuming. On the web, templates can be used to efficiently generate design alternatives. However, this limits designers to the corpus of pre-constructed templates. What if users could use *any* web page as a template? This paper presents a technique for borrowing the layout and style from any web page and applying it to your own.

The remainder of the paper is structured as follows. We describe the relationship of this paper to prior work. We



**Figure 2.** To edit layouts in Blogger, move around labeled content blocks.

then present the pipeline and algorithms used to synthesize alternatives. Finally, we discuss results and future work pertaining to building a graphical interface leveraging our technique to aid web design.

### Related Work

This work draws on two main areas of prior work: web design tools that aid in producing higher quality designs and techniques for generating design alternatives.

Template-based systems (*e.g.*, Blogger) operate on the principle that pre-existing content can aid design. These systems allow users to create many cookie-cutter web sites, but they greatly limit expressivity. Prior work has also explored using web pages with authentic content as starting templates. Lee *et al.*'s Adaptive Ideas [6] enables users to parametrically browse through a corpus of actual web pages, use the example pages as starting templates, and borrow style attributes (limited to font and

colors) from them. With Adaptive Ideas, users must manually adapt content to the example designs in the corpus. The system introduced here performs this step automatically. Ivory *et al.*'s system [4] uses learned statistical profiles of good sites to suggest improvements to existing designs to increase their quality. These changes, however, must be manually implemented by the user, and the system provides no

assistance for designers in the tabula rasa stage of development.

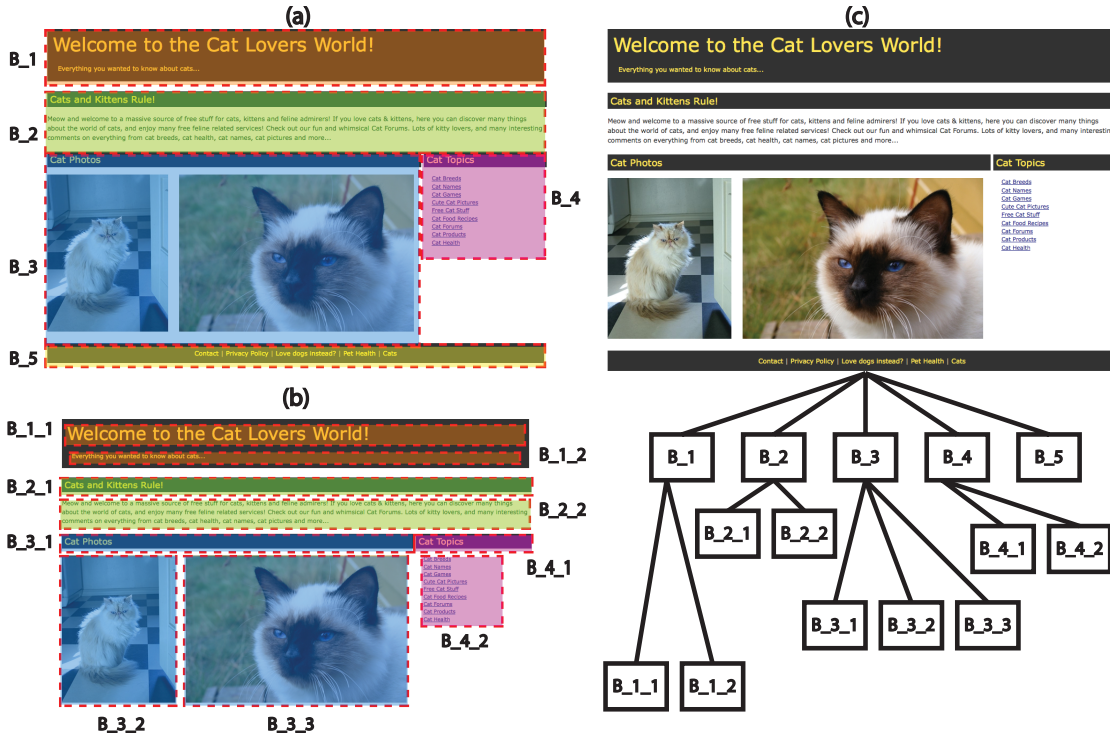
Prior work has explored two main approaches to generating alternatives. One is to represent the design as a set of parameters, which are manipulated to explore the space of alternatives [7, 9]. The other is to synthesize alternatives by adapting and merging past solutions to fit the current context: a common expert practice and the keystone of case-based design literature [5, 6]. Our framework provides an explicit mechanism for borrowing layouts and styles from other web pages to synthesize alternatives.

### Synthesizing Alternatives

Automatically adapting content to different design layouts poses two significant technical challenges. First, given a pair of web pages, both must be segmented into their constituent perceptual blocks. Second, a mapping between the blocks of the two pages must be computed. Current template-based systems that support automatic layout changes require pages to be composed of labeled content blocks (Figure 2). Different template layouts are simply different arrangements of the same content blocks, making retargeting straightforward. We present a new method for merging content and layout between any two arbitrary HTML pages without imposing restrictions on content.

#### *Perceptual Tree Construction*

To merge content and layout of arbitrary HTML pages, it is necessary to codify the visual structure of web pages. We implemented a modified version of Cai *et al.*'s "Vision based Page Segmentation" (VIPS) algorithm [2], an iterative, top-down method for building a tree representation of a web page's content structure.



**Figure 3.** A web page segmented using our implementation of the VIPS algorithm [2]. (a) Blocks extracted after the first iteration. (b) Blocks extracted after the second iteration. (c) Resulting perceptual tree structure.

Each iteration of the algorithm has three phases. In the visual-block extraction phase, the system recursively traverses the DOM tree of the page and extracts DOM nodes which correspond to visually dominant blocks. The heuristics for determining perceptual importance are based on visual cues such as background color, size, and content type. In the next phase, vertical and horizontal separators are detected between the extracted blocks. These separators are assigned weights according to the visual difference between the blocks that they separate. In the final step, the extracted blocks are merged together into new blocks to form the hierarchical representation of the content

structure. Merging starts with the blocks separated by minimally-weighted separators and iterates until the blocks separated by the maximally-weighted separators are merged.

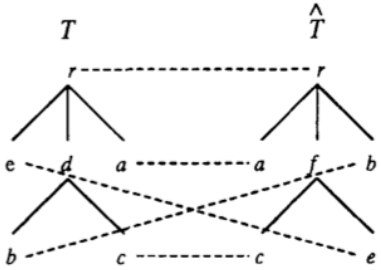
At the end of an iteration, the leaf nodes of the tree (*i.e.*, the extracted blocks from that iteration) are fed back into the next iteration of the algorithm unless they exceed a granularity threshold (Figure 3). It is important that both the page containing the content and the page containing the layout have been broken down to the same level of granularity for our mapping algorithm to perform well.

#### Tree Mapping

Given this perceptual tree representation of web pages, the mapping between the visual structures of a pair of web pages can be computed. In general, a mapping between two trees specifies a sequence of edit operations that should be applied to each node in order to transform one tree into the other (Figure 4).

Suppose we impose an ordering on the nodes of a given tree. Let  $T[i]$  represent the  $i$ th node of tree  $T$  in the given ordering. Formally, a mapping  $M$  from  $T$  to  $\hat{T}$  is a set of pairs of integers  $\{(i, j)\}$  satisfying the following constraints:

1.  $1 \leq i \leq |T|$ ,  $1 \leq j \leq |\hat{T}|$ , where  $|T|$  is the number of nodes in  $T$ .
2. For any pair of  $(i_1, j_1)$  and  $(i_2, j_2)$  in  $M$ :
  - $i_1 = i_2 \Leftrightarrow j_1 = j_2$  (one-to-one),



**Figure 4.** A mapping between two unordered trees,  $T$  and  $\hat{T}$  [8]. A dotted line from a node in  $T$  to a node in  $\hat{T}$  indicates a matching operation. A node in  $T$  and  $\hat{T}$  that is not touched by a dotted line should be added and deleted, respectively.

▪  $T[i_1]$  is an ancestor of  $T[i_2] \Leftrightarrow \hat{T}[j_1]$  is an ancestor of  $\hat{T}[j_2]$  (ancestor relationship preserved) [8].

Then, the reward of a mapping  $M$  can be defined to be the sum of all individual rewards associated with each edit operation:

$$\gamma(M) = \sum_{(i,j) \in M} \gamma(t_1[i] \rightarrow t_2[j]) + \sum_{\{i | \nexists j \text{ s.t. } (i,j) \in M\}} \gamma(t_1[i] \rightarrow \lambda) + \sum_{\{j | \nexists i \text{ s.t. } (i,j) \in M\}} \gamma(\lambda \rightarrow t_2[j]),$$

where  $\gamma(t_1[i] \rightarrow \lambda)$  and  $\gamma(\lambda \rightarrow t_2[j])$  represent insertion and deletion edit operations, respectively.

#### Predicting Mappings

Given this formalized notion of mapping between trees, we use the generalized perceptron algorithm for structured-prediction [3] to predict reasonable mappings between the visual structures of web pages. For this supervised learning algorithm, a training example is given by a pair  $(x^{(i)}, y^{(i)})$ , where the input variable  $x^{(i)}$  represents a pair of perceptual trees and the target variable  $y^{(i)}$  represents the mapping between them.

To predict good mappings, the algorithm learns how to weight (reward) shared features between a pair of perceptual blocks. Using these learned weights, the optimal mapping (that is, the mapping with the highest reward) can be calculated for a given tree pair. More formally, the algorithm learns a rewards-weight vector  $\vec{w}$  such that  $y^{(i)} = \arg \max_y \vec{w}^T F(x^{(i)}, y)$ , where  $F(x^{(i)}, y)$  is a feature-counts vector containing the number of

appearances of each feature in a given mapping. The feature-counts vector we use has the form

$$F = \begin{pmatrix} \text{\# of inserts and deletes} \\ \text{\# of matchings with matching tags} \\ \text{\# of matchings with similar content} \\ \text{\# of matchings with similar dimensions} \\ \text{\# of matchings with similar positioning on page} \\ \text{\# of matchings meeting none of the above criteria} \end{pmatrix}.$$

Each iteration of stochastic gradient ascent executes the following sequence of operations until  $\vec{w}$  converges:

- Pick a random training example:  $(x^{(i)}, y^{(i)})$ .
- Compute  $\hat{y} = \arg \max_y w^{(t)T} F(x^{(i)}, y)$ , where  $t$  is the iteration variable.
- Update  $\vec{w}^{(t+1)} = \vec{w}^{(t)} + \alpha^{(t)} (F(x^{(i)}, y^{(i)}) - F(x^{(i)}, \hat{y}))$ , where  $\alpha^{(t)} = 1/\sqrt{t+1}$  is the learning rate.

In each iteration, the optimal mapping  $\hat{y}$  is computed for the given tree pair  $x^{(i)}$ , using the current rewards-weight vector  $w^{(t)}$ . The algorithm for computing the optimal mapping between two unordered trees is NP-complete. However, if we bound the number of leaves in even one tree, the exhaustive search algorithm runs in polynomial time [8]. The leaves of our trees correspond to the smallest perceptual blocks that are returned by our VIPS implementation. By controlling segmentation granularity, we generate trees that have fewer than 20 leaf nodes, giving a reasonable bound on the running time of the algorithm. Dynamic program-

Scott R Klemmer

Assistant Professor of Computer Science



Engagement and awareness... In the process with their education design... collaboration and creativity...

(a)

ming solutions exist but introduce mapping constraints that greatly restrict the flexibility in choosing mappings between trees [10]. Furthermore, once we have learned a reward function that allows us to predict good mappings between perceptual trees, we can employ heuristic algorithms to predict mappings [8] instead of relying on exhaustive search.

We are still in the process of generating training data to produce better predictions. Manually constructing mappings between tree pairs is very tedious; therefore, we are designing a graphical interface to aid in the process. With such an interface, we can even crowd-source the task.

CV-style resume for Scott R Klemmer, listing education (MIT), research (HCI Group), and teaching (MIT courses like Introduction to Human-Computer Interaction).

Three columns of text describing the HCI Group's work on interaction between humans and computers, the design of interactive systems, and the design of user interfaces.

(b)

(c)

Preliminary tests done with only a few mappings in the training set are promising. The reward weights corresponding to the various features are reasonable: the weight associated with "insertions and deletions" is negative, while the weights associated with matched pairs with "matching html tags", "similar content", and "similar dimensions" are large and positive.

Synthesizing a new page Given a mapping between two page trees, we can synthesize a new page in which the content from one page has been adapted to the layout of another. Since all the graphi-

cal content of a page is contained within the leaf nodes of its tree, we consider only the image of the leaf nodes of the content page under the computed mapping. If these nodes are all mapped to blocks in the layout tree, we are finished. If some nodes are not mapped, we must determine approximate mappings in order to ensure that we transfer all of the content from our current design. We accomplish this by collapsing the unmatched blocks into their siblings or ancestors, depending upon which of these blocks have the best mappings; see Figure 1(c). The synthesized pages shown in Figure 5 were constructed using this method.

Conclusions

This work-in-progress introduced a novel technique for automatically retargeting the content of a web page onto the layout of any other web page. This technique allows users to quickly realize new alternative designs for their web pages without restrictions on content. We have observed that our algorithmic pipeline produces reasonable results when mappings are manually defined, and the results of the mapping predictor trained with only a few examples are likewise encouraging. We are currently collecting more training data to produce better predictions, which we hope will result in a fully automated retargeting pipeline.

After that, the next step is to build a graphical interface which leverages this technique. Since there could be many reasonable ways to actually retarget content onto different layouts, presenting the user with a gallery of options would allow for greater customization flexibility. By extending this pipeline, we could also support borrowing of style and layout separately. We plan to deploy this tool as a web application to do large-scale



**(b)**

the hybrid debate

Can your choice of car change the world we live in? What would be the knock on effects of a global switch to hybrid cars? Join the debate!

Editor's overview

What do you think?

Can your choice of car change the world we live in? What would be the knock on effects of a global switch to hybrid cars? Join the debate!

Change display style:

Environment Politics & Energy Business & Economy Families & Lifestyle Urban Planning

Specialist Views

Phil Sivell Why the UK needs a new policy

Bob Geldof Why low-carbon living should be encouraged but not imposed

Chris Vernon Can hybrids make a difference to the way we live?

Neil Spiller The car has a future

Roger Kemp Where will the hybrid live in the future?

Meet the specialists | Hybrid research | Lexus Hybrid

Terms of use | Privacy policy

**(c)**

Scott R Klemmer

What do you think?

Can your choice of car change the world we live in? What would be the knock on effects of a global switch to hybrid cars? Join the debate!

Assistant Professor of Computer Science

Research

Finding Inspiration for the Future in Our Past

Integrating Physical and Digital Interactions on Walls

Iterative Design and Evaluation of an Event Architecture for Parallel Paper Publishing

Design As Exploration: Creating Interface Alternatives through Parallel Authoring and Runtime Tuning

Hacking, Mashing, Gluing: Understanding Opportunistic Design

Meet the specialists | Hybrid research | Lexus Hybrid

Terms of use | Privacy policy

empirical studies on how these methods would effect the quality of designs produced.

**References**

[1] Buxton, B. *Sketching User Experiences: Getting the Design Right and the Right Design*. Morgan Kaufmann, San Francisco, CA, 2007.

[2] Cai, D., Yu, S., Wen, J., and Ma, W. VIPs: A Vision-based Page Segmentation Algorithm. *Microsoft Technical Report*, MSR-TR-2003-79, 2003.

[3] Collins, M. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *Proceedings of the Acl-02 Conference on Empirical Methods in Natural Language Processing - Volume 10 Annual Meeting of the ACL*. (2002), 1-8.

[4] Ivory, M. Y. and Hearst, M. A. Statistical profiles of highly-rated web sites. In *Proc. CHI 2002*, ACM Press (2002), 367-374.

[5] Kolodner, J.L., and Wills, L.M. Case-Based Creative Design. In *AAAI Spring Symposium on AI and Creativity*. Stanford, CA. (1993), 50-57

[6] Lee, B., Klemmer, S. R., Srivastava, S., and Brafman, R. Adaptive Interfaces for Supporting Design by Example. *Stanford Tech Report*, CSTR 2007-16, 2007. (<http://hci.stanford.edu/cstr/>)

[7] Marks, J. *et al.* Design galleries: a general approach to setting parameters for computer graphics and animation. In *Proc. SIGGRAPH 1997, ACM Press/Addison-Wesley Publishing Co., New York, NY* (1997), 389-400.

[8] Shasha, D., Wang, J. T. L., Zhang, K. Exact and approximate algorithms for unordered tree matching. *IEEE Trans. Systems, Man, and Cybernetics* 24, 4 (1994), 668-678.

[9] Terry, M. and Mynatt, E. D. 2002. Side views: persistent, on-demand previews for open-ended tasks. In *Proc. UIST 2002*, ACM Press (2002), 71-80.

[10] Zhang, K. A Constrained Edit Distance Between Unordered Labeled Trees. *Algorithmica* 15, 3 (1996), 205-222.

**Figure 5.** Two synthesis results (continued from previous page). Mappings were manually defined. (a) Content page, (b) layout pages, (c) synthesized pages. **Zoom for more detail.**