Abstract

The latest trend in advertisement is to display video logos in a context devoid of ads such as movies or television shows where the product is featured in the storyline of the show. Such embedded marketing strategies are considered distractions and are often the cause of decreasing viewing pleasure for audiences. This paper presents a method for automatically detecting and removing logos from digital images given the logo of interest. During the logo detection phase, the initial logo boundary is localized using Speed Up Robust Features (SURF) technique and the logo is completely removed after further refinement by active contour based on a Mumford-Shah segmentation method. In the image filling part, we use exemplar-based inpainting to fill in the removed logo location in a visually plausible way. The combination of these three methods allows robust detection and removal of the target region without manual selection.

1 Introduction

For several years, major companies have used product placement through television and movies to advertise their products to the general public. A primary method of advertising has been the use of logos and trademarks specific to each company. However, given the increasing influence of television and other media, the constant presence of advertising has been found to be distracting. We suggest that the removal of these advertising logos could enhance the viewing experience. While most product placement is intentional, there exist cases where a logo’s presence is not foreseen leading to an unfortunate lawsuit from the company. Our solution helps in both types of situations.

In order to accurately remove the logos of interest from a target image, we divide our task into two main phases: logo detection and image filling for the removed portion. For the logo detection portion, we apply Speed-Up Robust Features (SURF) that accurately locates a logo while being invariant to scale and rotation of the logo in the image. After the logo of interest is located with SURF, we mark this region to be removed with the Region-based Active Contour method. Then, the resulting image is passed into the Exemplar-based Inpainting Algorithm which preserves the structural portion of the removed region based on its surroundings.

We’ve developed our work in the following fashion - section 2 gives detailed description of the two methods, SURF and Region-based Active Contour, that are used in logo detection process. In section 3, Exemplar based Inpainting is presented to achieve image filling after the removal process. Sections 4 details our experimental results and comparisons between our results and the latest method of logo detection and removal by [1]. Finally we conclude our work in section 5.
Figure 1: Overall flow of our logo detection/removal algorithm.

## 2 Logo Detection Algorithm

Our proposed logo detection algorithm is composed of two steps as indicated in Fig. 1. To detect logo, we first localize the general position using Speeded Up Robust Features. The general shape of the logo is extracted from the detected matching feature points from SURF. The naive model region is refined in accordance with active contour technique based on a Mumford-Shah segmentation. These steps are described in more detail below.

### 2.1 Speed Up Robust Features

The first part of our task is to locate a given logo in a field image. In order to locate the logo in the field image, we must perform object detection. We accomplish this by using a detector and descriptor called SURF (Speeded-Up Robust Features) [4]. The algorithm first detects interest points at various locations in each of the images. These interest points can be at places such as corners, edges, and blobs. SURF uses a fast Hessian detector to detect interest points within the images. The Hessian matrix is a matrix containing the second-order partial derivative information of an image at a certain point. An interest point represents an instance where the determinant of the Hessian matrix is large leading to a local maximum. SURF approximates the Hessian matrix using Gaussian box filters and convolves them with the corresponding integral image at that location. An integral image is the sum of all the pixels within a specified rectangular region of the input image.

The algorithm then attempts to find interest points at different scales. When an image is represented at different scales it is called a scale space. The scale space is constructed by upsampling the 9 by 9 Gaussian box filters. The input image is then convolved with these upsampled image and the interest points are extracted. Rotation invariance is then ensured through the use of Haar wavelets.

Once the interest points have been computed for the target image, we determine the corresponding interest points in the input logo and proceed to match them. We notice that the majority of the interest points in the left image of Fig. 2 are clustered around the logo; however, in the right image of figure there exist interest points that are nowhere near the desired logo. Therefore, we compute the Euclidean distance of each interest point to the centroid of each image. If the distance of any interest point to the centroid in the target image exceeds the maximum distance from an interest point in the logo image to its centroid, then that interest point in the target image is removed.

Removing the extraneous interest points, we construct a rectangular mask to represent the area that needs to be filled in using the inpainter. This rectangular region is constructed by spanning the
Figure 2: The images on the left shows how our modifications to SURF remove all of the interest points that are too far from the centroid we get interest points clustered around the target logo. The images on the right shows what happens when these erroneous interest points are not removed. In this case, the erroneous interest point is indicated by the green arrow.

minimum and maximum coordinate values in both the horizontal and vertical directions. This region is shown in Fig. 3

Figure 3: The left image is the mask that represents the removed apple logo. The right image is the mask that represents the removed Mercedes Benz logo.

2.2 Region-based Active Contour

One of the important aspects of computer vision is finding the appropriate boundary in images. In our method of model region refinement, we use region based active contours. The region based active contour method first assumes initial curve (in our case nave square from SURF output) and define contour energies based on regional properties. The key assumption here is that when the curve is placed on the object border, the image will be partitioned into two or more distinctive regions whose properties are easily characterized [2].

The basic idea of the algorithm could be explained based on famous flows by Chan-Vese [3]. First, assume that the image is formed by two separations, foreground $u_0^f$ and background $u_0^b$, based on
the mean intensities of the two regions. Assume further that the model region to be detected is represented by the region with $u_i$. Then we can formulate the energy function as following:

$$E = \int_{\Omega} (I - u)^2 dA + \int_{\Omega^c} (I - v)^2 dA$$

where $\Omega$ and $\Omega^c$ represent the interior and exterior of the curve respectively, and $u$ and $v$ represent the mean image intensities over $\Omega$ and $\Omega^c$. The first term of the energy function represents the internal energy and the second term represents the external energy. As shown in Fig. 4, the energy function is minimized when the mean intensities, $u$ and $v$ are most accurately approximated in $\Omega$ and $\Omega^c$. Since the energy function takes account to the global image data, the refinement can be robust to the noise and curve placement.

Figure 4: Four possibilities of curve formation

3 Image Filling Algorithm

3.1 Exemplar-based Inpainting

To fill the holes created by the previous blocks of the system, we use a method of inpainting called exemplar based inpainting. This method is developed by Microsoft Research in [1]. There are many methods considered but this method has been the best in terms of preserving structure in the image. One can simply use linear inpainting, but this causes a great loss of detail. A secondary method we looked at was described in [5]. This method is shown to be somewhat successful, but the version discussed in the Microsoft Research paper is shown to do better.

Exemplar-based inpainting works in a fairly simple iterative way. We specify the region that we want to fill, and then proceed to fill it based on a priority based system. The priority is computed based on two components. The first is a confidence term. This term is found by looking at the current patch we wish to fill and calculating the percentage of pixels in the patch that we already know from the filled region. The second term is called the data term. This term is more complex to calculate and understand. The point of this term is to capture the structure information present in the image. Throughout reconstruction, we keep track of the gradients of the image. We need to keep both the x direction gradient and the y direction gradient in order to gain insight into the true structure. We also keep track of the "fill front", that is, the gradients of the fill region. To compute the data term at a pixel location on the fill front, we simply take the dot product of the two terms. The insight into why this makes sense is as follows, we want to first fill the patches where the fill front is perpendicular to the gradient direction of the image at the fill front. Points that satisfy this are ones that have a great deal of structure information we wish to propagate.

The major improvement that we see in this method over previous inpainting results is that we have prioritized filling in such a way that we will make sure the structure is propagated before anything else. This is somewhat similar to how people would tend to fill an image if you asked them to draw what was removed from an image. The greatest information you have comes from the structure around the image. After we compute the confidence and data term of each patch on the fill front, we multiply the terms for each patch, and choose the one that resulted in the highest total as our
Figure 5: This figure shows the process taken to determine data term of the current patch centered at pixel $p$, $\psi_p$. We compute the normal to the fill front $n_p$ and the isophote at point $p$ given by $\nabla I_p$ for all pixels on the fill front. Similarly, we can compute the confidence term of the same patch by summing up the number of terms inside of the patch that are a part of $\phi$ and dividing by the total number of pixels in the patch.

We patch to fill. From this patch, we move around the image pixels that we know and compare the patch centered at each of these pixels with the pixels that are already filled from the current patch. When we find the patch that is results in the least MSE from the patch we wish to fill, we copy this patch to the current location we wish to fill. We then proceed to do this over and over again until we have no region left to fill.

3.2 Improvements to Method

There are a few improvements made to the inpainting method described in the paper, and the results of the version with changes are be shown below. The first issue that we notice is that the current method is not actually detecting gradients in an image very well. The method works in the grayscale domain, and this gives small edge results when the color image would actually have very large gradients. This is changed by computing the color difference between points instead of the gradient

Figure 6: Comparison of the results before and after improvements to the method. The figure on the left is the improved version, and the one on the far right is the old version. As you can see, our version completed the house and shoreline much more convincingly.
of the image. The second issue with the previous implementation is that the patch size is constrained to be the same. Crimisi suggests that the patch size is correlated with the texture resolution which means that images with very fine texture should have a small patch size, and images with course texture should have a larger patch size. This is implemented by downsizing and upsizing the image and detecting the MSE of the image and its reconstructed form given this change. A simple threshold is found and it works reasonably well to decide the correct patch size. The third design choice is that we wanted to give more preference to the data term than [1]. It is somewhat intuitive to fill an image based on the data term that contains the structure portions of the image rather than using the confidence term. This is implemented by adding a small constant to the confidence term to provide a small boost in data term priority.

4 Experimental Results

![Figure 7: Examples of result images](image)

We have implemented several steps of a complete system to accomplish automatic logo detection and removal in media images. Fig. 7 shows examples of resulting images. The left side of the figure displays the original input images and the right side shows the resulting image after the complete process.

We can easily see that our automatic logo detection and removal system correctly finds the model regions and inpaints them consistently with the surroundings. It is very interesting to observe that in the second resulting image our system not only correctly removes the logo but also fills in the very thin line that goes through the logo. Solely considering the resulting images, it is almost impossible to detect any residue from the original logo.

5 Conclusions and Future Work

In this paper, we have presented an automatic logo detection and removal which uses a combination of three methods: SURF, Region-based Active Contour and Exemplar-based Inpainting. Our scheme is robust enough to handle random logo detection and removal automatically. Most importantly, our automatic system performs better than the inpainting system done by Microsoft Research Group which manually selects the region of interest.

Future work will involve further improvements on individual steps of the system to develop more robust detection and removal system. Additionally, we would like to study automatic or adaptive setting for deformations of the patch sizes in inpainting to boost system speed. This will be a key factor in branching out our system to take video data.
References


