SERBIAN JOURNAL OF ELECTRICAL ENGINEERING
Vol. 14, No. 2, June 2017, 177-197

UDC: 004.738.52:004.932; 004.4'273 **DOI: https://doi.org/10.2298/SJEE161010001T**

A Multi-Layered Image Format for the Web with an Adaptive Layer Selection Algorithm

Milan Tair[1], Aleksandar Mihajlović[1], Nikola Savanović[1], Marko Šarac[1]

Abstract: In this paper we present a proposed multi-layered image format for use on the web. The format implements an algorithm for selecting adequate layer images depending on the image container's surroundings and size. The layer selection depends on the weighted average brightness of the underlying web page background within the bounds of the image. The proposed image format supports multiple image layers with adjoined thresholds and activation conditions. Depending on these conditions and the underlying background, a layer's visibility will be adequately set. The selection algorithm takes into account the background brightness, each layer's adjoined threshold values, and other newly introduced layer conditions.

Keywords: Image format, Layers, Selection algorithm, Web, Responsive design.

1 Introduction

Ever since the early days of the Internet, programmers have been inventing different ways of including multimedia content into web pages. Because connection speeds used to be slow, image formats were designed with limitations in order to conserve space. This also helped reduce the impact of loading extra content into the page over the network. Some formats introduced colour indexing [1]. This provided a way to use a greater number of colours without restricting them to the 8-bit colour scheme. Other compression algorithms were designed on the Fast Fourier and Discrete Cosine Transformations [2]. Nevertheless, both of these formats still had major limitations despite providing smaller file sizes. Among these limitations was the lack of the alpha channel in the image's raster matrix. Indexed colour image formats, such as the Graphics Interchange Format (GIF), introduced binary transparency. This feature provided the ability to use limited transparency capabilities in image formats. This was the first step before the introduction of full alpha channel implementations.

The alpha channel is a feature available with certain image formats that derives from the method of encoding colour in binary format. These image

[1]Singidunum University, Danijelova 32, 11000 Belgrade, Serbia;
E-mails: milan.tair@gmail.com; mihajlovich@gmail.com; nsavanovic@singidunum.ac.rs; msarac@singidunum.ac.rs

formats add an additional byte or bytes for each pixel, where this additional byte is used to suggest the level of opacity for each pixel. This way, alongside the coloured layer, the image can carry another layer of information which suggests how opaque a certain part of the image's raster is. This layer is the alpha channel or the transparency layer. If an image is placed on top of another image, it could be made to seem translucent to a certain degree, depending on the values of pixels in the alpha channel for a certain region or the whole image.

Subsequent Web media formats, such as the PNG format, introduced four values per pixel. Three values are for the red, green and blue colour components and the fourth is the alpha channel [3]. All these formats share a common fact that they are all single layer image formats. Layered image formats exist and are commonly used in print and pre-press [4]. These formats have not been sufficiently used on the Web. This is because of the limited browser support [5] and inconsistent image layer rendering techniques. This results in unpredictable displaying across different browser engines [6]. Even formats with alpha channel support have the issue of contextual adaptability. Layered image formats are rendered in such a way that all layers are placed one on top of another. The result would be the same as in the case the image created within a pre-rendered single layer image, which is in term, stored with a multi-level alpha channel.

2 Problem Definition

The aim of this paper is to introduce a new image format. This format attempts to resolve an issue of low contrast between the image content and the background contrast. This problem occurs when images with alpha channels are placed on variable brightness backgrounds. Additionally, we have introduced new conditional triggers into the layer selection algorithm to take the image parent's size into account, in response to the growing demand for this kind of functionality in contemporary responsive design trends.

For instance, there is an image of a certain company's logo. The logo contains text in a custom type and a certain graphical component. The content of the image is predominantly in a darker colour scheme. This image implements alpha channel transparency. All pixels that do not contain logo text and graphical components are transparent. Such an image is placed over a predominantly bright background. In that case, the content would be easily distinguishable on the page. However, if the web page section on top of which the image was to be placed is predominantly dark, the image contents would be hard to distinguish. The content would be unrecognisable due to the low contrast between the background and the content. In such cases, designers would need to create a new logo with predominantly light text and graphical components, which is time-consuming and requires professional engagement.

Additionally, designers should be equipped with an additional document that describes the proper use of different colour versions of the logo. In this way, the contrast between the logo and the background can be high. Otherwise, the designer may choose to introduce an underlying bright wrapper that would exist within the image itself. This container would surround the main textual and graphical components of the logo. When such an image is placed on a dark background, the surrounding container would be visible, and the logo would still remain distinguishable. On the other hand, if such an image were to be placed on a bright background, the container would not increase the visibility of the logo, but would rather create a visual obstruction of the underlying background.

Also, companies often insist on a minimum clearance margin around the logo and other elements, which can be predefined and added to the file. These are all important parts in creating an automated file that will ensure a proper use of the corporate logo in accordance with all defined guidelines. Companies and organisations consider this issue a key element for ensuring proper brand identity and the most immediate representation of the company. A valuable corporate asset must be used consistently in adequate and approved forms to ensure corporate marketplace recognition. Thus, this issue is identified as one of the major concerns in the domain of marketing.

Fig. 1 – *Comparison of rendering a random company logo with opacity via alpha channel on light and dark backgrounds.*

As can be seen in Fig. 1, the logo is distinguishable on the light background, while it is hard to recognize it on dark backgrounds. The available image formats, either single or multi layered, do not provide a solution to this scenario. The current best practice is to create two images and load one or the other depending on the background.

The second scenario that can occur is when an image is displayed within a container that is smaller than the minimum size defined for that particular image. In such cases, Web developers usually request multiple versions of the logo image to be displayed for different parent's container sizes. They display different images on the web page depending on the available space. This is a

common occurrence in web pages designed with the responsive design principles, where different versions, commonly more and more simplified, are used for smaller display screen resolutions.

3 Proposed Solution

As explained in the previous section, current multi-layered image format implementations do not provide a method of having one or more layers algorithmically selected to be displayed depending on the underlying background and the space available within its parent container. The image format proposed herein attempts to solve that. We also describe a simple algorithm for deciding whether a certain layer should be rendered on the page.

The algorithm function takes a number of arguments for layer selection. Based on the ideas presented and implementation explained in our other paper [7] where we explore the responsive logo, related to the second usage scenario of the problem, we extend the selection algorithm. The algorithm now takes into account both the underlying background bounded by the image's coordinates and declared size and available space and size of the parent element within which it is contained on the page.

3.1 Image format description

The proposed image format is based on the PNG image format, the JSON (JavaScript Object Notation) [8] and the ZIP archive format [9]. The PNG format implements the transparency by using an additional value for mapping pixel opacity at a certain point in the image matrix.

Also, our format introduces layers with additional meta-data and image data. We describe each layer with an optional layer name and two mandatory brightness threshold values, as well as the minimum container size for which the layer should be available to the rendering engine.

The name of the layer exists for the purpose of the specialised image format editing tool that uses the name to indicate to the user to identify layers in the image format's container tree when a file is opened for editing. The layer name has no impact on the image’s rendering on the web page. The two brightness thresholds are the parameters used in the algorithm explained below. They can be set manually, by the user, or calculated automatically via the method that will be explained afterwards in Section 4.3 with the help of a certain use case.

The herein proposed algorithm for the layer selection procedure used these values. Image data for each layer consists of a complete PNG image with the same dimensions as the image. Image format packaging is based on a ZIP archive with a predefined file and directory structure. The root of the archive

contains a JSON formatted file manifest.json. The manifest contains plain text description of the package and settings for each layer.

Listing 1, available in the Appendix to this paper, provides the basic information about the image. Available information includes layer thresholds and smallest and largest allowed dimensions of each layer, as well as the priority of each layer in case that multiple layers meet the criteria for displaying within the image bounds. The rendering engine will render only the image with the largest priority value among multiple selected ones. The number of elements in the layers array determines the layer count. In addition, the use of the algorithm property defines which algorithm should be used for layer selection. Introduction of this property reflects the design decision for future implementation of alternative algorithms. The format version property identifies which renderer engine should handle image rendering. The newest implementation of the test rendering engine is versioned 0.2.0. Alongside the manifest file, each layer is stored as a PNG image. Layer images are stored in the root of the archive and are named layer_001.png, layer_002.png, layer_003.png etc. There can exist up to 999 layers. The enumerated part of the layer filename corresponds to the index in the manifest objects property layers, with left zero padding to three digits.

Another important element in the manifest is the properties object and its margins value. Currently, our rendering engine supports only the margin property to be set. This value represents an array that can be used to force the rendering engine to allow a clearance margin around the image content. Array values are margins listed in the following order: top, right, bottom and left, i.e. clockwise from the top margin, measuring in pixels.

In order to implement our solution, we needed to develop a method of analysing the image clipped from the page background content bounded by the image's rectangular bounds. There are many methods of analysing images and processing them in order to determine their properties. In this implementation, we have used an image-processing algorithm that proved to be adequate for this task. We have used a technique of comparing images based on their properties layer by layer and identifying image brightness as well as patterns in the image. Upon identifying patterns, we have clustered them into a weighting matrix used to weight the actual segment of the background in relation to the logo layer stored in the image format. This process is done for all images of same size ranges. This is because our image format supports layers of different targeted minimum and maximum display sizes.

3.2 Layer selection algorithm

In this proposed image format implementation, the drafted layer selection algorithm is relatively simple. It uses the aforementioned analysis algorithm.

First, it finds the average background-brightness-intensity, which is used to determine which layer to render. This decision depends on the defined image layer thresholds and the underlying background. In addition, the final layer selection depends on the layer priority in case multiple layers are initially selected in the first phase of selection.

The min and max brightness properties stand for the smallest and the largest percentage of the background brightness. If the background brightness is between these values, the layer is rendered. Threshold values are inclusive. If there is a layer that has the minimum brightness set to 0.0 (0%) and the maximum brightness of 1.0 (100%), than that particular layer will be shown regardless of the calculated background brightness intensity, provided that the layer's priority is greater than other selected layers' priorities and that the bounds of the image have a maximum size of 165 with 85 pixels (per concrete given example). As per the manifest example given in Listing 1 for the second layer threshold values, the situation is different. That particular layer will be rendered only if the underlying background brightness intensity is between 0% and 30% and if the maximum size of the bounding rectangle for the image is within the defined range. The value 0% is the darkest intensity in the colour scheme. The value 30% is a point, which is at one tenth of the total distance between the darkest and the brightest brightness intensity. More precisely, this layer is set to be displayed only when the image is placed on top of dark backgrounds. Likewise, the third and fourth layer will be evaluated if the size of the targeted bounds for the image is greater than or equal to 166 by 86 pixels and if their calculated underlying background picture intensities are within the range specified for each layer.

As seen in the aforementioned example, the third and the fourth layers do not have the maximum_size property object set. Because of this, the rendering engine will allow these layers to be evaluated for any size of the image target bound size that is greater than the size specified in the minimum_size property object. This differs from the first and second layer which must must include the maximum_size property object in order to avoid multiple layer selection in the first phase of the layer selection process.

Therefore, the conclusion is that the minimum required set of properties of each layer object in the manifest's layers property includes the following: minimum_brightness, maximum_brightness, minimum_size and the priority. As mentioned earlier, the name property is optional and is used to easily identify the layer in the multilayer image format editor software.

The new property called priority, added in the second iteration of the layer selection algorithm's development, is an unsigned, positive integer value starting from zero. The layer with the priority value of zero is the least significant in the order of selection when multiple layers are initially identified

as potential layer candidates for rendering on screen. Before the second and the final phase of the selection process can commence, the first phase must complete successfully with at least one layer selected. The first procedure begins with the calculation of the intensity of the part of the page's background underlying the image and bounded by the targeted position and size of the image on the page.

The procedure for calculating the background intensity according to the second version of the algorithm, implemented for testing this proposal, can be described with the following steps:

- Finding the average opacity of each point in the mask matrix for each alpha channel point of image layers. Equation (1) defines each layer's alpha channel as a matrix of real values of width and height w and h. Each point in the matrix is a value between 0 and 1.

$$L_i = \mathbb{R}^{w \times h}, \quad i = 1...n, \; n = 2...99 . \tag{1}$$

Equation (2) defines the method of determining values for average opacities of each point in a mask matrix L_a with the same size as layer matrices L_i to L_n.

$$L_a = \frac{\prod_{i=1}^{n} L_i}{256n}, \quad L_a = \mathbb{R}^{w \times h} . \tag{2}$$

- Capturing the rendered web page content as a raster image, bounded by the size and position of the image on screen, is done using a screen capturing technique based on third party software [10]. Currently, web browsers do not support native methods for capturing images of the rendered web page as displayed to the user, due to potential security issues than can arise by allowing this. To that end, external software was required for the implementation of this phase in the experimental stage of this proposal's development and testing.

- Creating a new matrix with each point representing the grey-scale value of each point in the captured raster image of the background. This matrix will be referred to as L_g in the rest of the document.

- Creating a matrix which is the weighted background-worth-matrix calculated by the following equation.

$$L_b = \frac{L_g}{L_a} . \tag{3}$$

- After the weighted background worth matrix, its mean is calculated (4) and used as the average background intensity value. We use the mean of this matrix instead of the grey-scale background matrix. We value segments under transparent regions more than those under opaque regions of the image layer. This is because regions under transparent pixels yield more visibility of the underlying background's pixels, thus resulting in more interference of those particular regions of the background with the logo shown in the evaluated layer.

$$a = \overline{L}_b . \tag{4}$$

- All layers whose minimum brightness thresholds are less than or equal to and maximum brightness thresholds are greater than or equal to the calculated value are selected to be rendered within the image bounds. In this process, layers whose minimum and maximum sizes do not include the size of the targeted bounding rectangle for the image are skipped. This results in greater efficiency, since those particular layers would not be selected by the end of the first phase.

- We call the last step in the selection algorithm the second phase. In this phase, evaluated layers are sorted in the descending order according to their adjoined priority property value. The first in the resulting sequence takes precedence, and is selected for rendering.

After the selection procedure, the selected layer is rendered. The earliest rendering engine that we have developed used to render multiple layers one on top of another. However, in subsequent testing, we have determined that this is hard to control in practice.

Multiple layers, rendered one on top of another can sometimes be wrongly selected due to the selection algorithms weight tolerance, resulting in the rendering of two layers that should not naturally be shown at the same time. In the current implementation, only one layer is selected by the end of the second phase and it is rendered within the targeted bounds.

3.3 Active adaptation

The currently proposed image format implementation does not support active background monitoring, nor does it include the means of actively calculating the average background-brightness-intensity. Continuous selection of layers is not possible at present.

Future implementations may provide such a mechanism at greater cost of resources such as memory and processor time. However, we have successfully tested the rendering engine with the implementation of the resize event listener added to the image's container element. With this, we have successfully tested

and confirmed the possibility of using our proposed image format to support the responsive web design principles. This could, in general, help mostly with the implementation of logos on pages designed for a variety of target platforms and screen sizes.

The introduction of digital portable devices calls for complete and continuous processes of adaptation of a range of visual brand elements for different marketing initiative.

Digital platforms such as smart phones and social media encourage logo adaptation. This is required in order to achieve better flexibility and for the brand to remain visible. Thus, responsive logo design must be applied carefully in favour of enhancing the brand image and identity [11].

Google also announced that responsive design would become a notable ranking factor in the Search engine results page (SERPs). Because of this, all companies on the market that wish to gain and/or retain a competitive advantage take responsive design as an extremely important development objective.

As Ofcom's Eighth International Communications Market Report shows [12], smart phones and tablet devices are equally important as a desktop and a laptop. This means that having a consistent strategy across all user platforms can be extremely effective. Such trend is so prevalent that Google began boosting ratings of mobile-friendly sites. This ranking applies for searches made from mobile devices.

Due to such reasons, we believe that our decision to implement a responsive component in our proposed image format is justified. One of important parts of properly displaying an adequate image from a list of embedded layers, not only on the size, but also on the orientation and rotation of the image's container element within the web page.

3.4 Predefined image rotation

The proposed image format takes into account the fact that the image can be rotated using CSS3 styling properties. If the rotate property is not applied or is set to 0.0 the rotation procedure is not executed. If the remainder of division of the rotation angle with 360º were to be greater than 0 and less than 360º, the rotation mechanism should be executed. This mechanism follows the following outlined procedure:

- After the L_a matrix is calculated, its weighed centroid is calculated as explained in (5) and (6). Equation (5) defines the method of determining the x coordinate of the matrix and (5) defines the method of determining the y coordinate of the alpha channel mask matrix.

$$c_x = \frac{\sum_{i=1}^{n} x_i L_{a_i}}{\sum_{i=1}^{n} L_{a_i}}, \tag{5}$$

$$c_y = \frac{\sum_{i=1}^{n} y_i L_{a_i}}{\sum_{i=1}^{n} L_{a_i}}. \tag{6}$$

- When the coordinates of the weighted alpha channel centre are found, a new matrix is created. The new matrix width and height are calculated after applying rotation. The rotation is done around the calculated coordinates. New bounding box coordinates are the minimum and maximum x and y coordinates after applying the original corner point transformation and organising them into new coordinate vector points. A simple coordinate transformation function is given in (7) for the x coordinate of a point. The same method is used for the y coordinate

$$x_i = x_0 + (x - x_0)\sin\alpha + (y - y_0)\cos\alpha \tag{7}$$

- In (7), x_0 and y_0 are coordinates of the point around which the image is being rotated and α is the angle.
- After the new bounds are calculated, the layer selection process can continue from (1). All layer matrices are rotated and fit into the new bounds. Finally, one layer will be selected by sorting them by the priority property value in the descending order and choosing the one with the largest priority value.

As explained in this section, the main feature of the proposed image format is the ability to select one layer, which is most adequate for rendering on screen for given conditions. This would enhance the visibility of the main content of the image, depending on its surroundings. Designers can choose which logo is optimal for different backgrounds. They can set the logo which is optimised for bright backgrounds into a layer set to be rendered when the image is placed on a bright background.

Likewise, they can set the logo optimised for dark backgrounds into a layer set to be rendered when the image is placed on a dark background. This way, one image package can contain two completely separate images. Each layer contains an optimised logo for dark and light backgrounds. This method can be used to create one image for more than two situations for different grades of background brightness.

Also, as explained above, the new feature of the proposed image format is its ability to adapt to changes of available space that occur through the implementation of responsive design on web pages. Even though RWD is mostly employed on the web, modern mobile applications use the design pattern for responsive user interfaces [13]. These differ from the concept of RWD used when rendering pages on mobile devices. Even so, responsive designs targeting mobile devices continue to gain popularity [14].

4 Use Case

In the case of the image containing a company logo shown in Fig. 1, at least one solution can be found for resolving the low contrast issue, shown on the right. One method relays on adding a permanently visible surrounding logo holder that would be visible on all backgrounds. This method would be the only applicable in case of PNG images that are the most used image formats on the web, which fully support the alpha-channel [15]. In this section, two methods of resolving the aforementioned issue are given.

4.1 Predefined image rotation

The first method uses the proposed image format with two layers. The topmost layer contains the company's logo as seen in Fig. 1. The second layer contains the company's logo with a drawing of a certain shape designed to fit the logo visually, drawn with a solid fill around it. These two images are stored within two independent layers. The first layer is set to be shown when the background lightness is above 60%, while the second layer is set to be shown when the background lightness is less than 60%. Depending on the underlying background, the appropriate layer shall be selected for rendering. The result of the rendering is shown in Fig. 2 for both the light underlying background and the dark underlying background.

Fig. 2 – *Comparison of rendering a random company logo with opacity via alpha channel on light and dark backgrounds with the use of the proposed image format with two layers preset for different underlying background brightness.*

As can be seen in Fig. 2, the rendering shown on the right is a result of the second layer being shown in the finally rendered image. This is because the background brightness is less than 60%. The image on the left shows the same logo without the second layer shown. This is because its underlying background, although coloured, is not within the defined threshold range of 0% to 60% for that layer.

4.2 The selective content method

This image format also provides a derived use. Initially, this use was not considered. Later on, it was recognised as a potentially good practice and is now considered the recommended method. It is based directly on the previously explained method, but the second layer's image content is actually a completely redesigned logo, designed to perfectly fit darker backgrounds without having to add abstract and disruptive elements, such as a coloured shape around the original logo. It is characterised by using two or more different image contents for each layer. In addition, layers have threshold values set in such a way so that no more than one layer is shown at one time for one background brightness intensity.

For example, this selective content method allows the designer to define two modes of the same logo. Each logo can be optimised for two selected background brightness ranges [16]. In the example given in this use case, the original company logo has another version adapted for dark backgrounds. The image is packed with two layers, and each of them contains the full logo. Thresholds are set to 0.0 and 0.59999999 for the first layer, which contains the dark background optimised logo, and 0.6 and 1.0 for the second layer, which contains the light background optimised logo. The resulting rendering of an image packed in that way is shown in Fig. 3.

Fig. 3 – *Comparison of rendering a random company logo with opacity via alpha channel on light and dark backgrounds with the use of the proposed image format with two selectively displayed independent content layers.*

The method used to render the logo shown in Fig. 3, can be used for creating a range of more than two logos. Each logo can be fitted for more than

two background brightness intensity ranges. By designing such an image, almost all scenarios can be accounted for. This way, inadequate contrast issues are resolved with just one file loaded into the web page. This is especially useful if the web page design changes themes for each user according to the personalised design [17].

Additionally, using the available responsive adaptability of the format, many more multiples of the logo can fit within the package for a wide variety of display scenarios.

4.3 Comparing layer images and identifying their properties

Each layer image is matched against the background and the appropriate thresholds are identified. Thresholds are selected by identifying dissimilarities between the image and the background. Proper thresholds are identified by measuring the recognisability and similarity of positions of certain peaks.

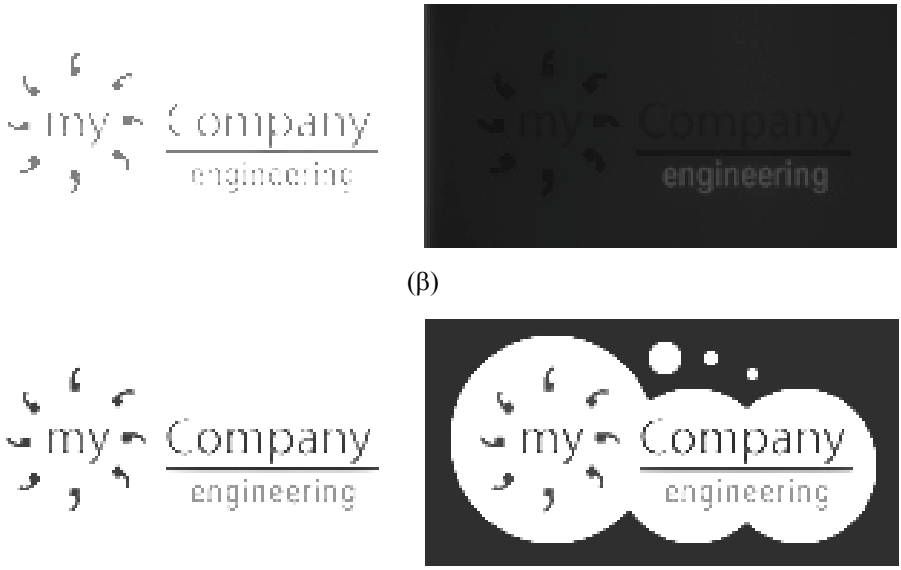

Fig. 4 – *Comparing peaks* (β) *and* (β′) *in order to identify patterns within the image resulting from overlaying a layer image over the target background.*

Parameters that are used for comparison are colours, textures, contours and peaks. Multiple parameters are chosen for comparison to yield favourable results. We have compared frequencies for each colour j in the background β and the overlaid image β′ based on the coherent pixels for the same colour rounded by the following formula (8).

$$\mathrm{dist}(\beta,\beta') = \sum_{j=1}^{J}\left(\left|\frac{\alpha_j - a'_j}{a_j + a'_j + 1}\right| + \left|\frac{\beta_j - \beta'_j}{\beta_j + \beta'_j + 1}\right|\right). \tag{8}$$

In (8) α and β are coherent and incoherent pixel colour histograms, respectively. The presented method was used to calculate the distance between colours.

First, we used a monochromatic version of the resulting image with different brightness levels (grey and white), which were automatically iterated through to create all possible toeholds. In such images, we looked for negligible and regular variations of the distance from the midpoint colour value (grey) and recorded all repetitions. With these results, we formed a matrix with rows and columns representing all shades of the image and the number of identified peaks. By iterating through this image, thresholds with the maximum number of smallest coherent peaks where identified. Finding peaks allows for finding the outline of the shape of the rendered image. After putting the logo over the background, we use the following method:

$$K = \sum_{a=0}^{N-1}\sum_{b=0}^{N-1}(a-b)^2 g(a,b). \tag{9}$$

In (9) *a* and *b* are indices of the formed matrix and g is the matrix. Value *K* is the resulting average colour intensity of the entire matrix.

When we utilised these methods to analyse images for all thresholds in the matrix and their corresponding calculated *K* values, we found the number of pixels of the matching *K* value, thus finding all peaks. These peaks are counted for each threshold. For logos overlaid on over a pure white background, the following peak representations are found:

Fig. 5 – *Comparison of the number of identified peaks at two neighbouring threshold values for the logos over a white background.*

Fig. 5 shows the peaks for threshold values 45 and 46. The number of peaks at threshold 45 is 1, which is also the minimum useful threshold value, where the image on the right has a larger number of peaks for the threshold value 46.

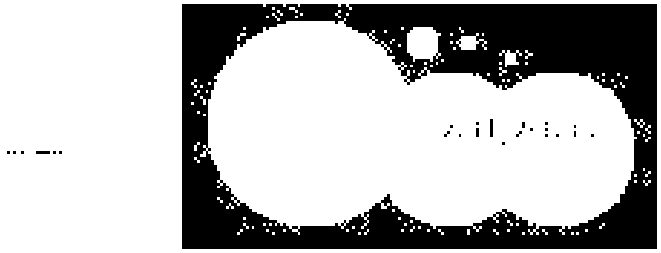

Fig. 6 – *Comparison of the number of identified peaks at two neighbouring threshold values for logos over a white background.*

Fig. 6 shows the peaks for threshold value 48. At this threshold value, the first image on the left shows a smaller number of peaks, while the image on the right has more peaks. However, the image has become unusable because the background had covered the entire surrounding of the logo.

When the designer does not input specific thresholds or when he or she wants the application to suggest proper thresholds for the newly designed and packed image, this method is used. Essentially, we attempt to find the minimum threshold that yields the maximum number of peaks, which are usable. In this way, we identify which of the layers (images) should be used for a predefined set of backgrounds partitioned based on the total number of layers packed in the image of this format. For example, if the image has two layers, we test layers against the white and the black background. If there are three layers, we test them against black, mid-way grey and white background.

4.4 An additional test case

To illustrate the functionality of the selection algorithm explained herein, we have created an additional test case to demonstrate the universality of the algorithm's effectiveness in determining values for brightness thresholds for each two layers of a new image. Thresholds are suggested to the user packing the logo into the image format container. These thresholds can be calculated automatically, as is the case in this situation. However, the user still retains the right to manually correct them.

These two versions of the logo are processed as explained in Section 4.3 with the resulting peak identification shown in Fig. 8. This example is for the white background against which both versions of the logo are tested.

In this particular result, the number of identified peaks for the version targeting dark backgrounds (left) was 6, while the number of identified peaks targeting light backgrounds (right) was 21. When we remove the remnants of the underlying original image to visualise only the peaks, we get the image shown in Fig. 9.

Fig. 7 – *Comparison of a random logo with opacity via alpha channel on light and dark backgrounds*

Fig. 8 – *Comparison of the number of identified peaks at two neighbouring threshold values for logos over a white background*

Fig. 9 – *Comparison of the number of identified peaks at two neighbouring threshold values for the logos over a white background.*

As displayed in Fig. 9, the version on the left with a smaller number of peaks is better suited for light backgrounds than the one on the right. This conclusion matches the obvious reasoning, as the logo on the left was originally designed for lighter backgrounds while the one on the right was intended for darker backgrounds.

5 Discussion

During the initial phase of developing software used for testing methods of implementation of our proposed image format, we have come across certain issues. These issues derive from security measures implemented in modern web browsers. These issues impose limitations to the possibility of wide implementation of our format natively in web browsers. Among these issues is the inability to capture the image of the rendered web page using native JavaScript in modern web browsers. As explained in the paper, we require the image of the rendered web page in order to crop out the region bounded by the image container so that the selection algorithm can determine the adequate layer to be rendered in the image container.

As we were not able to implement the acquisition of the underlying background for analysis using native JavaScript, we had to do it using third party software that operates outside the browser window. This method is impractical for production use, but was acceptable for proving the advantages of the concept.

A possible solution was to use the feature available in in Mozilla's Firefox web browser where the CSS property -moz-element is used to set the background of another element to the rendered image of the element specified as the argument of the -moz-element property. This is the vendor specific implementation of the CSS element property [18].

By creating an element whose content can be captured as a raster image and setting its background to the rendered image of the body element through this CSS property, we were able to acquire the image for analysis. Since other browsers do not support this property, we quickly abandoned the idea of using this method due to its limited browser support. Therefore, we considered it necessary to mention it and discourage attempts of underlying background acquisition using this method. Instead, we have decided to use third party software as previously explained herein. Based on our research, at the time of writing this paper and conducting testing and development of our proposed

image format test case engine, the most acceptable method was by using the PhantomJS [19] JavaScript library's Screen Capture functionality [20].

Using this library, we were able to acquire a rendered web page image in a raster form that was then analysed in a separate software component. A segment of the picture is cropped from the web page screenshot taking into account the rotation and scaling of the image's container element on the web page at its absolute position within the page. We used a simple program written in Java to convert the raster image into a two-dimensional matrix and to analyse it in order to determine which embedded layer is most adequate for rendering. After the program uses the selection algorithm to select the adequate layer, it extracts it from the archive and triggers a JavaScript function that loads it into the designated container element from a local path. Because the layer image is stored in a natively supported PNG format, the rest of the browser handles the rest of the rendering process natively.

Another issue that we have come across is the scenario when the web page is loaded with the HTTP Access Control Origin Headers [21] set within the specific domain or array of domains. In this scenario, a selected image is stored locally and thus referenced via a local path that does not match any of the domains set in the HTTP Origin Header list and cannot be displayed due to browser enforced restrictions. The authors shall attempt to solve this isue in their future work.

6 Conclusion

As described herein, the proposed solution helps solve issues that occur with image formats used on the web. The key problem solved by the proposed image format is the insufficient content to background contrast distance and the use of an adequate image based on its targeted screen size. It occurs on web sites which change the design based on the user personalisation philosophy. This is especially true in the domain of colour and contrasts of the surrounding context. The low contrast issue is currently solved by creating multiple files. Also, responsive web design plays a key role in additionally complicating means of selecting the appropriate images for different scenarios.

Traditionally, appropriate files are loaded through CSS depending on the known background colour. This solution is flawed in cases when the background is not a known colour. In such cases, the yielding visual impression, derived from its combination with the image is not predictable. Such cases may occur from backgrounds comprised of photographs etc. The solution detailed in

this paper presented a way of using the proposed image format. By supporting multiple layers with additional meta-data comprised of threshold values, the rendering engine determines whether a layer should be displayed or not. In addition, three related scenarios were presented that illustrate situations where a random company logo is displayed on bright and dark surfaces.

Scenarios with a common PNG image format and the proposed image format are shown. The new image format is presented by using two suggested design techniques to help one image file adapt to both background surfaces better. With both techniques, designers can set additional meta-data that would ensure proper use of the logo in accordance with the predefined rules and guidelines. This would result in a decrease of improper use of corporate logos by third parties, because companies depend on their logo being used in compliance with their guides or branding as one of the key elements of marketing. Moreover, using this proposed image format will allow companies to include multiple versions of their logos to be chosen adequately for rendering, not only based on the surface on which they are placed, but also depending on the size of the available space. It will allow the logo to be displayed in one of multiple modes, ranging from the most simple to the most elaborate logo designs when the display surface is sufficiently spacious.

7 Future Work

The current image format implementation uses third party software to fully implement the concept, as previously explained. This software renders the entire web page in a headless web page rendering engine and creates a raster image screenshot. It operates externally, outside the web browser frame and is therefore not appropriate for use in production conditions.

Development of a method for obtaining the underlying background will be explored in future research even though there is no pending implementation or announced plan to support such functionality in future versions of the modern web browser. In addition, future studies will explore additional methods for implementing support for scalable vector graphics image layers. Another important part of our future work will be to identify or develop a better method of identifying optimal layer threshold values and to support more CSS transformations, aside from scaling and rotation, which the implementation currently supports.

8 Appendix

```
{
  "format_version": "0.2.0",
  "author_name": "Forename Surname",
  "made_with": "DEMO image editor",
  "use_algorithm": "default-2",
  "properties": {
    "margins" : [ 5, 5, 5, 5 ]
  },
  "layers": [
    {
      "name" : "Small dark logo for any BG at <= 165x85",
      "minimum_brightness" : 0.0,
      "maximum_brightness" : 1.0,
      "minimum_size": {
        "width": 0,
        "height": 0
      },
      "maximum_size": {
        "width": 165,
        "height": 85
      },
      "priority": "3"
    },
    {
      "name" : "Small bright for dark BGs at <= 165x85",
      "minimum_brightness" : 0.0,
      "maximum_brightness" : 0.3,
      "minimum_size": {
        "width": 0,
        "height": 0
      },
      "maximum_size": {
        "width": 165,
        "height": 85
      },
      "priority": "2"
    },
    {
      "name" : "Large dark logo for any BG at >= 166x86",
      "minimum_brightness" : 0.0,
      "maximum_brightness" : 1.0,
      "minimum_size": {
        "width": 166,
        "height": 86
      },
      "priority": "1"
    },
    {
      "name" : "Large bright for dark BGs at >= 166x86",
      "minimum_brightness" : 0.0,
      "maximum_brightness" : 0.3,
      "minimum_size": {
        "width": 166,
        "height": 86
      },
      "priority": "0"
    }
  ]
}
```

Listing 1 – *Sample content of the manifest.json file in the archive root.*

9 References

[1] J. Miano: Compressed Image File Formats: JPEG, PNG, GIF, XBM, BMP, Addison-Wesley, NY, USA, 1999.

[2] G.A. King: Understanding and Designing Computer Networks, Butterworth-Heinemann, Oxford, UK, 1995.

[3] S. Powers: Painting the Web, O'Reilly Media, Cambridge, UK, 2008.

[4] M. Gatter: Getting it Right in Print: Digital Pre-press for Graphic Designers, Laurence King Publishing, London, UK, 2005.

[5] Wikipedia: Comparison of Web Browsers – Image Format Support. Available at: https://en.wikipedia.org/wiki/Comparison_of_web_browsers#Image_format_support.

[6] R.I. Chang, Y. Yen, T.Y. Hsu: An XML-Based Comic Image Compression, 9th Pacific Rim Conference on Multimedia, Tainan, Taiwan, 09-13 Dec. 2008, pp. 563 – 572.

[7] A. Mihajlović, J. Gajić, J. Stanković, M. Tair: The Importance of Responsive Logo Design Across a Wide Range of Devices on the Web, International Scientific Conference on ICT and E-Business Related Research, Belgrade, Serbia, 22 April 2016, pp. 50 – 55.

[8] Ecma International: ECMAScript Language Specification, June 2011. Available at: http://ecma-international.org/ecma-262/5.1/.

[9] S. Saxena: A First Course in Computers: Based on Windows Xp & Office, Vikas Publishing House, New Delhi, India, 2009.

[10] N. Sofer: SiteShoter, 2008. Available at: nirsoft.net/utils/web_site_screenshot.html.

[11] M. Kelly: Analysing the Complex Relationship between Logo and Brand, Place Branding and Public Diplomacy, Vol. 13, No. 1, Feb. 2017, pp. 18 – 33.

[12] Ofcom Research: Communications Market Report 2016, Aug. 2016. Available at: https://www.ofcom.org.uk/research-and-data/cmr/cmr16

[13] J. Lehtimaki: Smashing Android UI: Responsive Android UI and Design Patterns for Android Phones and Tablets, John Wiley and Sons, West Sussex, UK, 2012.

[14] Google Trends, 29 March 2016. Available at: https://trends.google.com/trends/explore?q=%22responsive%20web%20design%22.

[15] Web Technology Surveys: Usage of Image File Formats for Websites, 2016. Available at: w3techs.com/technologies/overview/image_format/all.

[16] S.J. Koyani, R.W. Bailey, J.R. Nall: Research-based Web Design and Usability Guidelines, Computer Psychology, Washington, DC, USA, 2004.

[17] I.H. Ting, H.J. Wu: Web Mining Applications in E-Commerce and E-Services, Springer, Chennai, India, 2009.

[18] Mozilla: CSS Element Property, Aug. 2016. Available at: https://developer.mozilla.org/en-US/docs/Web/CSS/element.

[19] V. Slobodin: GitHub - ariya/phantomjs: Scriptable Headless WebKit, 2016. Available at: https://github.com/ariya/phantomjs.

[20] V. Slobodin: Screen Capture, 2016. Available at: http://phantomjs.org/screen-capture.html.

[21] A. Barth: The Web Origin Concept, IETF RFC 6454, Dec. 2011. Available at: https://tools.ietf.org/html/rfc6454.