Colouring Black and White Photos With Deep Learning

Jose Acitores Cortina *7007337* | Martijn Dekker *6013368* | Daniele Di Grandi *7035616* | Alex Vermeer *5717329* | Matthijs Wolters *5983592*

***Abstract*—The focus of this paper is to implement a functional algorithm capable of colourising black and white images. In order to do that different types of images are fed to the designed deep learning model which, through its architecture composed by several type of layers, including convolution, will be able to restore the colours of the desired image.**

I. Introduction

For this paper we aim to colour black and white images with a Convolutional Neural Network (henceforth CNN). Our aim was to improve on the results given by Satoshi Iizuka, Edgar Simo-Serra, Hiroshi Ishikawa in their 2016 paper *Let there be Color!: Joint End-to-end Learning of Global and Local Image Priorsfor Automatic Image Colorization with Simultaneous Classification* [1]. Even though current research in this topic is more focused on using General Adversarial Networks (henceforth GAN) to colour images, the initial focus of this paper was going to be analysing different colour spaces for representing the black and white images and thus changing the features needed to be learned by the models. Therefore, we chose a standard CNN model, as the complexity of the GAN models would increase the training time significantly. Furthermore, this specific paper continues to be cited today and the model described in it can still be considered a state of the art colourisation network, albeit based on a CNN architecture instead. In their paper they define a two stream CNN to colour the images. The two streams share weights in the intial part of the network, which tries to capture the low level features of the images. This low-level network is a sequence of convolutional layers and is connected to two other networks known as the mid-level features network and the global features network. This use of the global and mid-level feature networks is a novel idea in the area of colouring images. The idea is that the global features network tries to help the colourisation network with finding the context of the image, for example, "outside", "people". Another novel idea is the fusion layer, which combines the global and mid level streams, so as to embed the global features in the mid-level representation of the image. This fusion layer then feeds into a colourisation network that gives the image its colour.

Our own contribution is that we tried to implement a version of this network with only the information provided by the original paper, as there is no public code available for this paper, but only a pre-trained model ready to download [2]. Due to time constraints we were not able to test different colourspaces and had to restrict our results to the colourspace of the original paper. However, we have modified the model slightly and will train the network from scratch on a different dataset. In the original paper, the model was trained on the Places dataset [3]. We made use of the train 2014, validation 2014 and test 2015 image sets from the COCO-Images dataset [4]. This dataset is more object oriented rather than related to locations, so we were interested to see how the model would perform on this kind of data. This dataset did not contain classification labels, and as a consequence it was not possible for us to implement the classification network, added by the researchers, to help train the global features network. This classification network's is given a weight $alpha$ for the total loss calculation in the original paper, to determine it's influence on training the global features network. However, as the value of the parameter $alpha$ given by the researchers to this network was very small ($\frac{1}{300}$), we surmise this network would have little effect on the final results and will only increase training times.

II. Related Work

Early work on automatic image colourisation was focused on old physical materials that have degraded over time, such as old bleached film reels or oxidised paintings. These materials usually still contain colour, albeit not the original ones. These early works primarily used mathematical models for the colour restoration.

Pappas and Pitas [5] use one such mathematical model for digital painting restoration. Their method requires a human to restore sections of a painting manually, which the model uses as a reference for the colour palette of the entire painting. Their model would then use this reference using different distance and regression functions to restore the remaining sections of the painting.

Rizzi et al. [6] used a two phase algorithm for restoring bleached film reels, be it an originally coloured film, or a black-white film. Their algorithm works on the assumption that the film bleaches about equally for every pixel, but not necessarily for every colour channel. It thus readjusts the pixels related to the image as a whole and then uses the generated image to estimate what white and gray are in the image and finally adjust the tone using this information.

More recently, exploration of completely automatic image colourisation utilizing neural network models has attracted the attention of many researchers. These works aim to create a

model that can colour images and film correctly with as little manual work as possible.

Zhang et al. [7] attacked this problem by proposing a feed-forward pass in a CNN trained on over a million colour images and evaluating the performance by a "colourisation Turing test" while using, as a loss function, a classification loss with rebalanced rare classes. Larsson et al. [8] and Iizuka et al. [1] have developed similar systems, using large-scale data and CNNs, though they use different loss functions: an un-rebalanced classification loss and a regression loss, respectively. Moreover, Larsson et al. used a hypercolumn on a VGG network as their CNN's architecture. Iizuka et al. used a two-stream architecture in which they fuse global and local features and Zhang et al. used a single-stream, VGG-styled network with added depth and dilated convolutions.

Nazeri et al. [9] tackled this problem using a different approach: fully generalise the colourisation procedure using a conditional Deep Convolutional Generative Adversarial Network (DCGAN) and they compare the results between their model and traditional deep neural networks models. It is also worth mentioning the implementation of Richart et al. [10], where they proposed a Multi Layer Perceptron Neural Network to classify the colour of a gray level pixel without the need of heavy image processing algorithms, achieving some good, although not optimal, results.

III. Methodology

The model we use is inspired by the paper of Iizuka et al. [1], consists of a CNN divided in 4 main components: a low-level features network, a mid-level features network, a global features network, and a colourisation network. The output of the model is the chrominance of the image which is fused with the luminance in order to form the final output. CNNs are special cases of Neural Networks in which weights are "shared" spatially across an image, resulting in the effect of reducing the number of parameters needed for a layer and gaining a certain robustness to translation in the image. Typically, both the weights and the bias term are learnt through back-propagation. We will now proceed with the analysis of the CNN presented in the original paper, the architecture of which is also visible in Fig. 1, and by doing so, we will discuss our changes and why, in our opinion, they will improve the quality of that work. In all the feature networks in which it is applicable, a 3x3 convolution kernel and a padding of 1x1 are used to ensure the output is the same size as the input. Or a stride of 2x2 is used to halve the size. As a last remark, the activation function used in each layer that we will discuss, is the Rectified Linear Unit (ReLu) function, since it is the most effective function in order to solve the problem of the vanishing gradient and when performances of computational efficiency (such as training time) are important, as it is our case.

A. Shared low-level features

The first part of the network consists of a 6-layer CNN that extrapolates the low-level features directly from the input

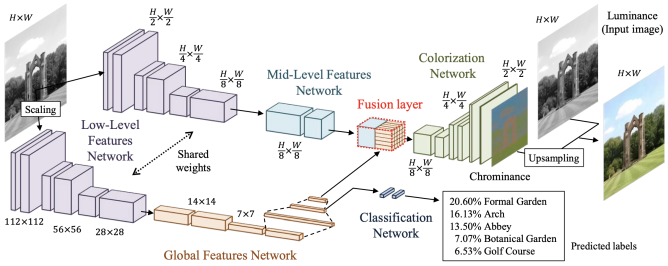


Figure 1: Original architecture. [1]

image. Both the mid-level and global feature sections of the network are preceded by this section. To ensure both streams continue with the same information, the parameters of layers in the low-level feature part are shared between both streams. This network is a fully convolutional network, thus, the output will be a scaled version of the input.

B. Global features

These features are obtained by processing the low-level features with 4 convolutional layers followed by 3 fully-connected layers, resulting in a 256-dimensional vector representation of the image. However, because of the nature of the linear layers in this network, it requires the input of the low-level features network in this stream to be of fixed size of 112×112 pixels. This limitation does not affect the full approach, as explained later in section III-D. The global features could play a fundamental role in establishing the context of the scene, by understanding for example, if an image is an outdoor or indoor image, as explained later in chapter III-D. Note that while the original paper has also a hidden classification layer and an output classification layer after the global features network, we decided to remove the classification layers in order to focus better on the implementation of the actual colourisation network part. In the original paper a baseline network, that consisted of a single stream of the model, was also tested. This was the low-level and mid-level stream of the model. In this baseline network they thus did not have the global features but also no fusion layer, as it was no longer necessary. It was our intention to test this model on a new dataset, with a model consisting of the both streams, but without the classification network, to see if it could still perform similar to the original paper.

C. Mid-level features

This part of the network is relatively small, consisting of only two convolutional layers. It functions to capture mid-level features, as well as bring the number of channels down to 256, the same as the global features part outputs.

D. Fusing global and mid-level local features

In order to combine the two feature outputs with different sizes from both streams, a fusion layer is introduced to the network. The purpose of this layer is to embed the global features into the local features. The global features act as a context to the image as a whole. The local features can

(a) Low-Level Features network

Type	Kernel	Stride	Outputs
conv.	3×3	2×2	64
conv.	3×3	1×1	128
conv.	3×3	2×2	128
conv.	3×3	1×1	256
conv.	3×3	2×2	256
conv.	3×3	1×1	512

(b) Global Features network

Type	Kernel	Stride	Outputs
conv.	3×3	2×2	512
conv.	3×3	1×1	512
conv.	3×3	2×2	512
conv.	3×3	1×1	512
FC	-	-	1024
FC	-	-	512
FC	-	-	256

(c) Mid-Level features network

Type	Kernel	Stride	Outputs
conv.	3×3	1×1	512
conv.	3×3	1×1	256

(d) Colorization network

Type	Kernel	Stride	Outputs
fusion	-	-	256
conv.	3×3	1×1	128
upsample	-	-	128
conv.	3×3	1×1	64
conv.	3×3	1×1	64
upsample	-	-	64
conv.	3×3	1×1	32
output	3×3	1×1	2

Figure 2: Types and Parameters of Layers in each Network [1]

then use this information to adjust which colours to use. For example, if the image is an indoor image, the local features will be biased to not attempt to add sky or grass colours to the image, but instead add colours suitable for furniture, floors and ceilings.
This embedding process can be thought of as concatenating the global features with the local low-level features at each spatial location. The combined features are then passed through a single convolutional layer to compete the feature mapping.
As the global feature vector is in essence copied for every location in the output local feature stream, the embedding is independent of the input resolution of the local feature stream. Mathematically, the fusion layer can be described as:

$$\mathbf{y}_{u,v}^{fusion} = \sigma\left(\mathbf{b} + \mathbf{W}\begin{bmatrix}\mathbf{y}^{global} \\ \mathbf{y}_{u,v}^{mid}\end{bmatrix}\right)$$

Where $\mathbf{y}^{global} \in \mathbb{R}^{256}$ is the 256 dimensional vector resulting from the global features network and $\mathbf{y}_{u,v}^{mid} \in \mathbb{R}^{256}$ is the mid-level feature at u, v in the weight matrix. The resulting weight matrix $\mathbf{W}$ is of size 256-by-512.

E. Colourisation network

After the fusion layer, the resulting features are processed by a set of convolutional layers (5 in total) and upsampling layers (3 in total) which consists of upsampling the input using the nearest neighbour technique so that the output is twice as wide and twice as tall. In this network, convolutional layers and upsampling layers are used in an alternating facion. In the original paper, the output layer of the colourisation network consists of a convolutional layer with a sigmoid activation function that outputs the chrominance of the input grayscale image. In our case, rather than using the sigmoid activation function, we decided to change it to the ReLu function as well. We decided to do this because we believe it will make the model focus on more chrominance, rather than less, reducing training times for the model to start colouring the images. This output is then upscaled one final time, so its size matches the original image. This computed chrominance is then combined with the input luminance image to obtain the actual final coloured image. An overview of the layers in each network is displayed in Figure 2.

IV. Results

We will compare the results of our model with results of the model that was implemented by Iizuka et al., using the same images for comparison consistency. Since their true code is not available, we downloaded their pre-trained model [2] and used it with our dataset for comparisons reasons. First it should be noted that due to limitations regarding time and processing power our final model is not as complete as the one we compare it with. They trained their model for three weeks on a dataset with 2,327,958 training images, where all images with very little colour variance were removed. Our model was trained on 50,000 images during a timer period of 14 hours, while no such images were excluded. Another important difference to mention here, is that the datasets used to train the models are dissimilar, as this is part of our research. Moreover, our testing parameters were different: while Iizuka et al. used a batch size of 128 and 11 epochs, we used a batch size of 64 and 7 epochs. A batch size of 128 was tested but due to the fact that this model is particularly complex the hardware that was available to use was not sufficient to handle this in a similar way. The amount of epochs chosen was partially due to time constraints and also because during earlier testing of the model the loss converged within very few epochs. As such, we chose a number of epochs deemed to be sufficiently long. Furthermore, due to hardware constraints and the complexity of the model, it was not possible to compute the model on the GPU. Our results were computed on a laptop with an AMD Ryzen 7 4800H CPU and 16GB of RAM at 3200MHz. The images from the dataset were streamed from a 1TB SSD. A link to the implementation of our own code can be found in the reference [11] Therefore, it is reasonable to expect that our model will perform worse on part of the images. Nevertheless, our testing gives a good insight into the potential of the CNN that we used for the model.

A. Results using different colourspaces

We wanted to see how the CNN, as proposed by Iizuka et al., could perform on different colourspaces and different ways of removing colour from images as a way of representing different camera specifics and image degradation over time. We adapted the CNN structure to output three streams for red, green and blue. Then provided the network with 8 different

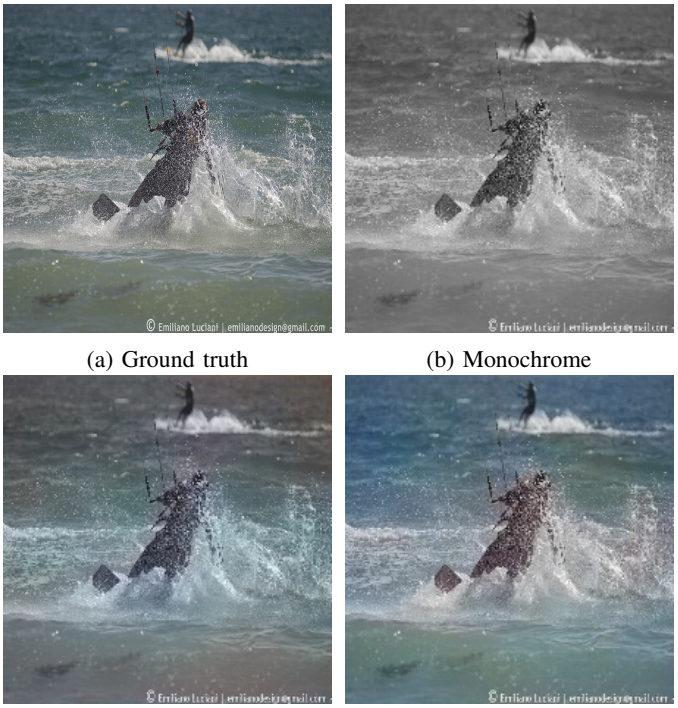
methods for the colour conversion into black and white. However, after training, we quickly found that this model was not capable of correctly colourising the images. Most images received very little colour or only one colour stream for the whole image, as seen in figure 3. Because of this setback we decided to focus more on the L*a*b colourspace.

(a) Ground truth (b) Run 1 (c) Run 2

Figure 3: Result using multiple methods for turning monochrome

*B. Results on the L*a*b colourspace*

When we trained our model using the L*a*b colourspace we did get better results. Although the images are not exactly like the original, there are still some images that are close to the original and still look natural. In Fig. 4 we can see that our

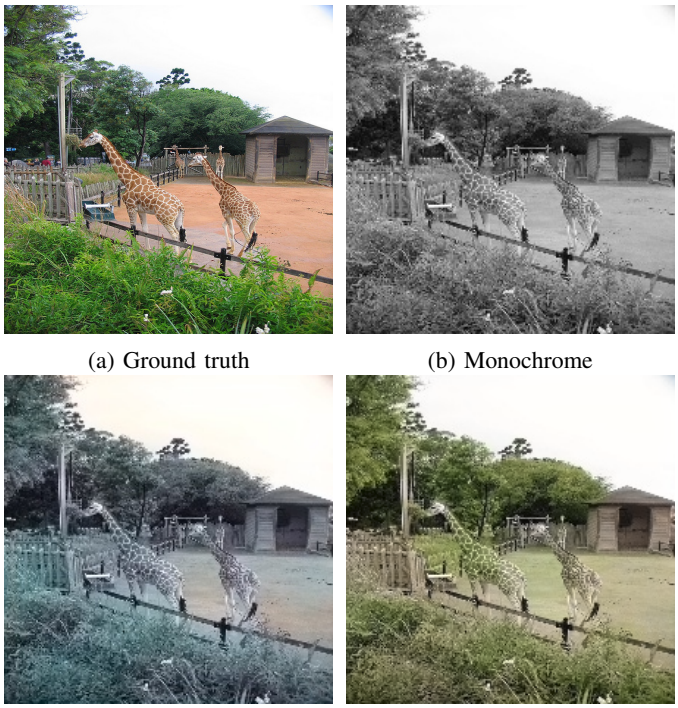


(a) Ground truth (b) Monochrome

(c) Our result (d) Iizuka et al.

Figure 4: Result using L*a*b

model gives the water a lighter colour than the water in the ground truth, but the water still looks natural. It also succeeds in keeping the white and black parts of the image the correct colour. The model does, however, fail at colourising the whole image, it seems to only colourise the center of the image. The model made by Iizuka et al. performs better, it colourises the whole image, but also seems to give the water a lighter colour than the ground truth.



(a) Ground truth (b) Monochrome

(c) Our result (d) Iizuka et al.

Figure 5: Result using L*a*b

In Fig. 5 we can see that both models perform sub-par. Our model gives almost everything a blue-ish colour while their model performs reasonably well except for the giraffes that are also green. This is probably because neither model has seen enough pictures of giraffes to correctly colour them. However, it is interesting to note that the ground colour, although there isn't much of it, is more correct in our result. This means that either our dataset contains more pictures with orange-ish ground colour, or that the more trained the model is, the more it becomes biased to colour outside ground green to mimic grass colour.

In Fig. 6 we can see that their model performs significantly better. Although the grass is greener than the ground truth (as earlier said, the model is probably biased to do that), as well as the plane having more dull colours than the ground truth. The overall picture looks very much the same. With our model, on the other hand, the sky turns orange and the grass and trees gain a blueish green colour. We also noticed that in several of our other results the sky seems to get coloured orange. We suspect that this is due to the limited number of pictures with a clear blue sky in the training data.

In Fig. 7 we see that their model performs slightly better yet again. However, both models turned the green banner purple. Another limitation of our model can be see here, namely the uncoloured man. This is probably due to a lack of training images containing a clearly visible and close by

(a) Ground truth

(b) Monochrome

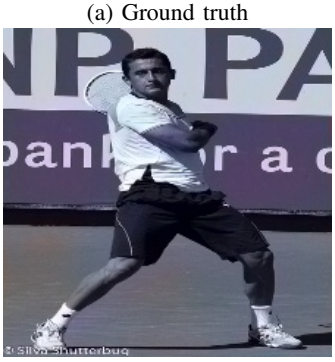
(c) Our result

(d) Iizuka et al.

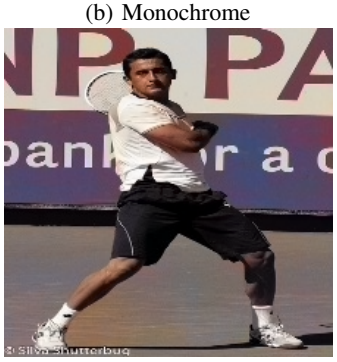
Figure 6: Result using L*a*b

(a) Ground truth

(b) Monochrome



(c) Our result



(d) Iizuka et al.

Figure 7: Result using L*a*b

human. Besides showing the poor performance, this image also shows the ambiguity of colourising. It is not unnatural for the banner in the background to be purple as well as the ground to be orange, because it is a common colour for tennis courts. Figure 8 showcases some more interesting results.

V. Conclusion

Based on the result of using the different colourspaces and colour conversion methods we can conclude that our model, using the CNN as proposed by Iizuka et al. and our limited dataset does not perform well enough to correctly colourise images. This CNN might perform better with a lot more training, but due to our limitations regarding time and processing power we were not able to test this properly. We do suspect that, without any modifications to the CNN, it will be difficult to get good results, because the methods for the colour conversion can give very different results for the same starting colour.

Our testing using the L*a*b colourspace had better results. When the limitations regarding time and computational power are taken into account, and the results of our model are compared to those of Iizuka et al., we can see that it performs reasonably well. We therefore conclude that there is reason to believe that the classification output part of the CNN is not necessary to colourise black and white images in a natural manner with results close to the original.

We do not think that this CNN is capable of colourising images 100% correctly. This is because the conversion from RGB images into L*a*b images and then only using the luminance as input will always cause some loss of information. We think that with more training and a bigger dataset, our model, without the classification part, could perform as well as that of Iizuka et al.

VI. Future Works

A. Metadata usage for colour accuracy

A line of future work consists of using additional information of the image in order to recolour the image in a more accurate way. Saving and accessing information like location, date or time of the day can help to understand important elements such as lighting or the prevalent colours in the region where the original image was taken. Therefore this would lead to a more loyal representation of the ground truth colours.

B. Key-frame based video recolourisation

In many occasions, when it comes to recolouring videos, there is a common problem of a large magnitude of images. The problem emerges due to the common practice of re-colouring every frame of the video separately, and as specified before, there is a phenomena where several colours can appear similar or the same in grayscale and therefore create ambiguity. If these ambiguous colours are considered different from one frame to another, this would make a terrible and inconsistent video. The aim of this branch of future work is to bring consistency to videos by recolouring one or a few specific key frames for every scene. An automatic system would then base the ambiguous colours on the chosen key-frames.

C. Specialised models

Projecting the different algorithms into a specific field would make them considerably better when it comes to that specific topic, this would allow the user to achieve a high-end result without the need of a high-complexity model which would take weeks of training. In many scenarios the pictures to colour back are indeed similar when it comes to the scenario, type of gray scaling and/or lighting, making the idea of specialising a viable option for these cases.

References

[1] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Let there be color!: Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification," *ACM Transactions on Graphics (Proc. of SIGGRAPH 2016)*, vol. 35, no. 4, pp. 110:1–110:11, 2016.

[2] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Let there be color!: Automatic colorization of grayscale images," https://github.com/satoshiiizuka/siggraph2016_colorization, 2016, [Online; accessed 15 January 2020].

[3] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[4] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2015.

[5] M. Pappas and I. Pitas, "Digital color restoration of old paintings," *IEEE Transactions on Image Processing*, vol. 9, no. 2, pp. 291–294, Feb 2000.

[6] A. Rizzi, C. Gatta, C. Slanzi, G. Ciocca, and R. Schettini, "Unsupervised color film restoration using adaptive color equalization," in *Visual Information and Information Systems*, S. Bres and R. Laurini, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1–12.

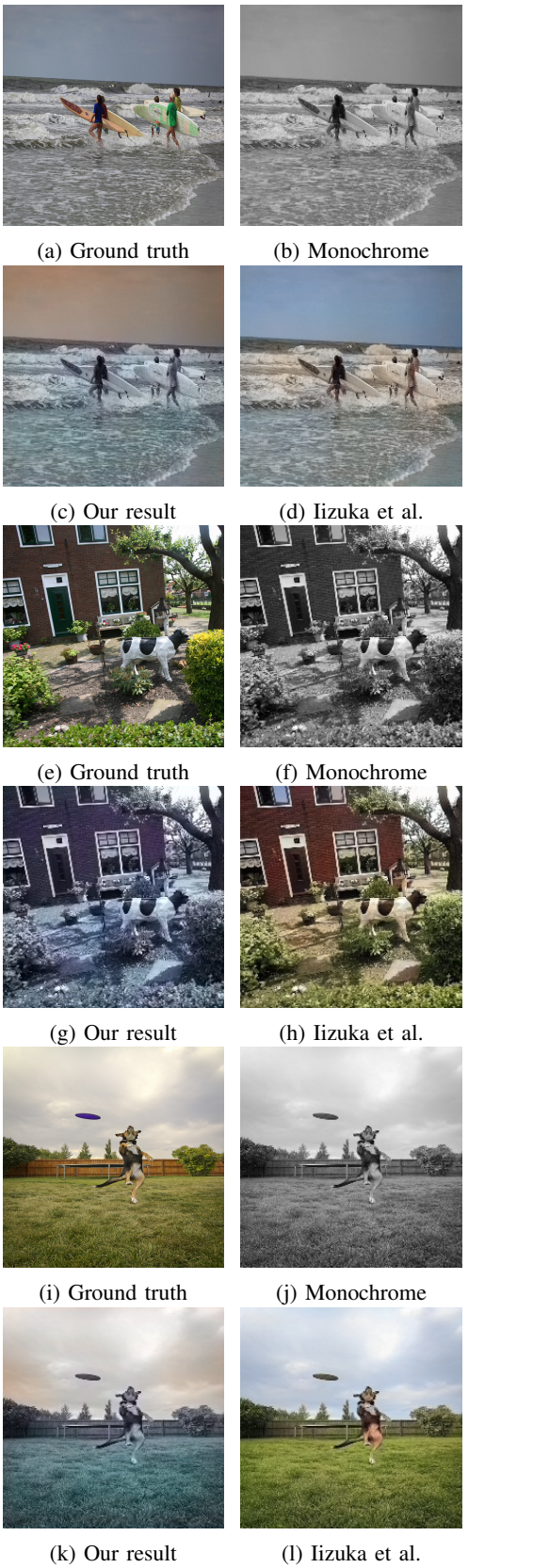
[7] R. Zhang, P. Isola, and A. A. Efros, *Colorful Image Colorization*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016.

[8] G. Larsson, M. Maire, and G. Shakhnarovich, *Learning Representations for Automatic Colorization*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016.

[9] K. Nazeri, E. Ng, and M. Ebrahimi, *Image Colorization Using Generative Adversarial Networks*, F. J. Perales and J. Kittler, Eds. Cham: Springer International Publishing, 2018.

[10] M. Richart, J. Visca, and J. Baliosian, "Image colorization with neural networks," pp. 55–60, 2017.

[11] A. Vermeer, D. Di Grandi, J. Acitores, M. Dekker, and M. Wolters, "Pattern recognition image colourization," https://git.science.uu.nl/a.vermeer/pattern-recognition-image-colourization, 2020, [Online; accessed 29 January 2020].



(a) Ground truth (b) Monochrome

(c) Our result (d) Iizuka et al.

(e) Ground truth (f) Monochrome

(g) Our result (h) Iizuka et al.

(i) Ground truth (j) Monochrome

(k) Our result (l) Iizuka et al.

Figure 8: Comparisons between our model and the model by Izuka et al.