Kernel Aware Resampler

Michael Bernasconi[1,2] Abdelaziz Djelouah[2] Farnood Salehi[2] Markus Gross[1,2] Christopher Schroers[2]
[1]ETH Zürich [2]DisneyResearch|Studios
michael.bernasconi@inf.ethz.ch, abdelaziz.djelouah@disney.com

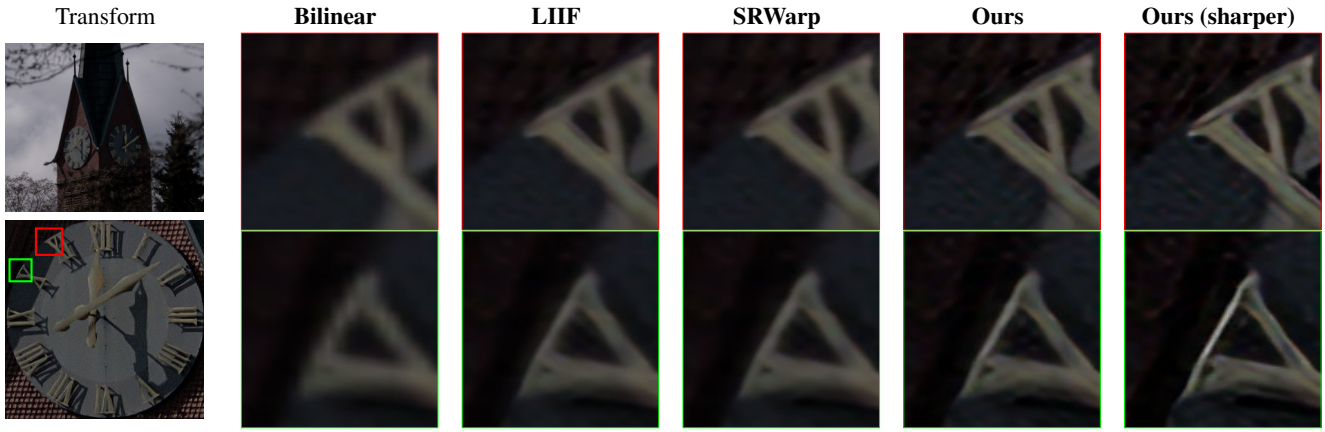

Figure 1. Comparison of our method with LIIF and SRWarp when rectifying a natural image. As the kernel map is unknown in this case it is estimated automatically for our method. Additionally our results can be further sharpened.

Abstract

Deep learning based methods for super-resolution have become state-of-the-art and outperform traditional approaches by a significant margin. From the initial models designed for fixed integer scaling factors (e.g. $\times 2$ or $\times 4$), efforts were made to explore different directions such as modeling blur kernels or addressing non-integer scaling factors. However, existing works do not provide a sound framework to handle them jointly. In this paper we propose a framework for generic image resampling that not only addresses all the above mentioned issues but extends the sets of possible transforms from upscaling to generic transforms. A key aspect to unlock these capabilities is the faithful modeling of image warping and changes of the sampling rate during the training data preparation. This allows a localized representation of the implicit image degradation that takes into account the reconstruction kernel, the local geometric distortion and the anti-aliasing kernel. Using this spatially variant degradation map as conditioning for our resampling model, we can address with the same model both global transformations, such as upscaling or rotation, and locally varying transformations such lens distortion or undistortion. Another important contribution is the automatic estimation of the degradation map in this more complex resampling setting (i.e. blind image resampling). Finally, we show that state-of-the-art results can be achieved by predicting kernels to apply on the input image instead of direct color prediction. This renders our model applicable for different types of data not seen during the training such as normals.

1. Introduction

Thanks to recent advances in deep learning based super-resolution which allow to infer impressive high frequency details from low resolution inputs, it has become possible to bridge the gap between content and display resolution without noticeable degradation in quality. This is beneficial in different contexts and, among other things, has enabled new visual effects production workflows to operate in 2K while still ultimately delivering at 4K resolution by performing a 2x upscale just before final delivery.

However, super-resolution is not the only image transformation that can occur in typical visual effects pipelines, and it is very common to perform additional tasks such as image rectification, retargeting, lens (un)distortion or image

warping. All these transformations require more complex image resampling solutions. Even the simple case of lens undistortion corresponds to a more complex type of resampling — which might locally upscale or downscale — that existing super-resolution methods do not support. As a result, one has to fall back to traditional interpolation based resampling approaches which can result in a noticeable and unnecessary loss in quality.

To the best of our knowledge, there is only one learning based method that considers more complex resamplings [15]. However, this approach has two drawbacks: On the one hand it is not optimally suited for real world content that might suffer from different kinds of implicit degradations from different blur kernels. On the other hand the solution seems more complex than needed due to the multi-scale warping and blending strategy.

In this paper we propose a framework for generic neural image resampling that is lean and better applicable to real world scenarios through handling implicit degradations. To achieve this, we build upon fundamental concepts of signal processing and decompose the resampling process into different stages, namely reconstruction, geometric distortion, and anti-aliasing. With this, we are able to create proper training examples to better handle and interactively control spatially variant degradation maps that are expected in image resampling. In addition to this, we design our approach to be able to predict kernels instead of directly outputting color values which makes the model more robust and enables consistent resampling of other channels, such as normals. Figure 1 illustrates this with a complex example: the transformation consists of image rectification and an increase in image resolution. This is an image that was not downscaled and the blur kernel is unknown. Our method automatically estimates the degradation map and produces sharper results than existing methods. Additionally it's possible to directly create outputs at different sharpness levels. This is the first time such applications are possible in image resampling.

Finally, we are able to show that our approach is able to beat the state-of-the-art despite its lean design allowing higher quality processing in parts of the visual effects pipeline that until now could not benefit from advances in deep learning.

2. Related Work

Resampling is one of the classic problems of signal processing and a large body of theory exists around it. Given the importance of images as a *type of signal*, image resampling is well studied and documented. For a review of key notions and solutions we refer to the work of [4]. Previous methods in computer graphics such as [22] have built upon this theoretical foundation. Our paper leverages such fundamental considerations in the case of neural based image resampling.

We start our review of related works with image super-resolution, which is one of the main sub-tasks in resampling. Here there is no geometric transformation of the image, but simply an increased sampling rate. By focusing on a simpler problem, works in this category pushed the limits of state-of-the-art by proposing new models and training strategies. These contributions are either already integrated in our model or represent an interesting future step for improved quality. Dong *et al.* [5] were among the first authors to propose using deep convolutional neural networks for image super-resolution. Since then the field has seen significant progress and we can mention, among other things, increased speed and application to video [14], adversarial training [8], and residual dense networks [17, 21]. We can note that any important progress in deep learning had an impact on model choices for super-resolution, as such recent works explored channel attention [20], transformer models [9, 19], diffusion models [13] and finally normalizing flows [10]. There is another new direction taken by recent works that explore frequency representation in the generator [12] or in the loss function [6].

Modeling the implicit blur kernel in the image for deep learning based super-resolution, as proposed by Zhang *et al.* [18], is one step toward a more generic setting. This blur kernel conditioned generator idea is leveraged by several blind super-resolution methods, differing mainly in their kernel estimation strategy [3, 11].

Even in the simple upscaling setting, most existing works are limited to integer scaling. Hu *et al.* [7] note that extending super-resolution to arbitrary scaling factors needs to address the arbitrarily large state for output pixels given possible scaling factors and offsets with respect to the closest input pixels. They solve this issue by using a separate weight prediction model that estimates weights to be applied on low resolution features according to position and scaling information of the output pixel. Chen et al. [2] showed that a local ensemble strategy was sufficient to extend possible scaling factors beyond the training range. Most recently [16] extend these ideas to address independent scaling factors for width and height.

To the best of our knowledge [15] is the only work that addresses image resampling using deep learning. They present some interesting ideas, namely by explicitly using the Jacobian of the local transform to modify pixel offsets before weight prediction. However the overall solution is overly complex in particular due to the multi-scale warping and blending strategy. Our proposed resampling layer is simpler, takes into account the local distortion more naturally by using the linearized Jacobian matrix and achieves better results. More importantly we take care to accurately model the implicit resampling operation to be undone which is a generalization of the blur kernel conditioning strategy

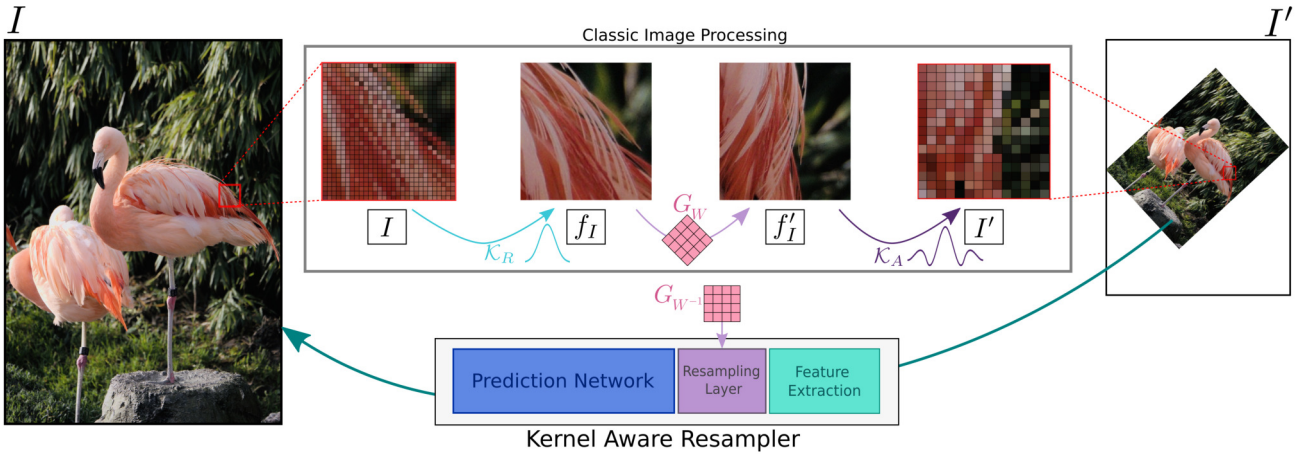

Figure 2. Preliminaries on the image resampling problem. Given an input image I, the classic approach for resampling starts with a conversion to the continuous image f_I using a reconstruction kernel $\mathcal{K}_R$. The continuous image is then warped according to W. To avoid aliasing artifacts, the continuous warped image f'_I is filtered before sampling. In the case of decreasing sampling rate, classic filtering techniques are favored, however increasing the sampling rate corresponds to an inverse problem and a deep learning method has a strong advantage.

used in SR.

Finally, besides addressing a wider set of transforms, we also have the possibility to resample other types of channels that can be available with the image (e.g. normals, alpha, depth, etc.) even if these are not seen during training. This is feasible by offering the option of image resampling through kernel prediction similar to some existing denoising methods [1]. To the best of our knowledge this is first time such applications are possible in image resampling.

3. Preliminaries

In this section we discuss some of the theory around image resampling and motivate using a learned approach in certain settings. Our goal in resampling is warping the image using a given mapping (or warp) W which maps coordinates in the input image to coordinates in the output image. Let us for a moment assume that the input image is a continuous function which may be evaluated at any 2D coordinates (x, y). In that case warping the image can be achieved using simple function concatenation. That is

$$f_{I'}(x, y) = f_I(W^{-1}(x, y)), \tag{1}$$

where f_I and $f_{I'}$ are the continuous input and output images respectively.

In case the input image is a discrete set of pixels, warping becomes more complex. Concretely we have two issues. First, the mapping $W^{-1}(x, y)$ does not in general result in coordinates that directly correspond to a pixel in the input image. Due to this some sort of interpolation is required to evaluate the expression $I'(x, y) = I(W^{-1}(x, y))$. Second, in case the sampling rate of the output image is lower than that of the input image, anti-aliasing needs to be applied to avoid artifacts.

To address both issues classic image processing formulates the resampling process of discrete images as illustrated in Figure 2. The input image I is first converted to the continuous image f_I using the reconstruction kernel $\mathcal{K}_R$. The continuous image is then warped according to W. To avoid aliasing artifacts an anti-aliasing kernel $\mathcal{K}_A$ is applied to the continuous output image f'_I. Finally, f'_I is sampled to create the discrete output image I'. In addition to illustrating these steps, Figure 2 also shows how the change of sampling rate plays a key role in the image we obtain. When the resampling operation destroys information then classic filtering techniques are sufficient. However if we try to increase the sampling rate (e.g. Super-resolution), we have an inverse problem and using a deep learning model has a significant impact on the result. The next sections detail how we solve this inverse problem.

4. Learned Image Resampling

Our objective is to address the generic image resampling problem in the case where the sampling rate increases. The notion of increased sampling rate is loosely defined here and mostly corresponds to the settings where the output image resolution is not significantly reduced. We start by describing the process that is used to generate training pairs with different degradations. After that we present the proposed resampling model, discussing in particular how information about the degradation and local distortion can be provided to the model.

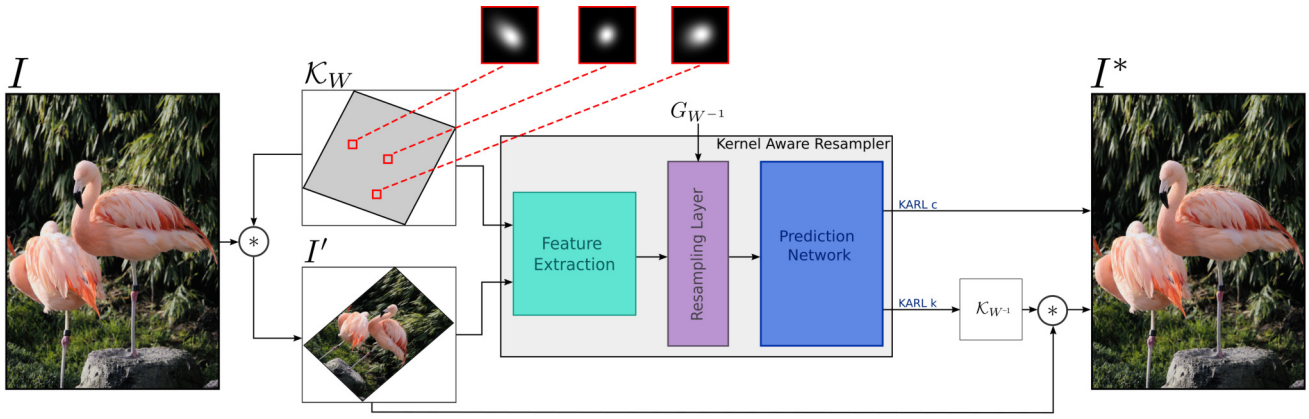

Figure 3. Overview of our proposed kernel aware resampler. First, features are extracted from the input image I' and the degradation map $\mathcal{K}_W$. Second, the extracted features are resampled using our resampling layer. Third, from the resampled features the resampling kernels or the output colors are predicted. In case the resampling kernels are predicted they are applied to the input image to produce the output. During training the degradation map $\mathcal{K}_W$ is applied on the ground truth image I to get the low resolution image I'.

4.1. Training Data Generation

We start by noting that modeling the implicit blur kernel in the input images was a key aspect for a kernel aware super-resolution model that can be applied on real-word data. This relatively simple task in the super-resolution setup, where even locally varying blur kernels are straightforward to model [3], becomes challenging in the case of generic image resampling due to the more complex transformation.

If we detail the equations corresponding to the description provided in the previous section

$$I' = \mathcal{K}_A * f_{I'} \tag{2}$$
$$= \mathcal{K}_A * (f_I \circ W) \tag{3}$$
$$= \mathcal{K}_A * ((\mathcal{K}_R * I) \circ W) \tag{4}$$

we can note that it is possible to combine the reconstruction kernel $\mathcal{K}_R$ and the anti-aliasing kernel $\mathcal{K}_A$ into a single kernel $\mathcal{K}_W$ which can be applied to the input image directly. The warping function will simply define the input image location where the kernel is applied as well as the offsets used for the kernel weight computations. By modeling the downscaling process this way we are able to generate a complex, spatially varying kernel map $\mathcal{K}_W$ by combining two simple, non spatially varying kernels $\mathcal{K}_R$ and $\mathcal{K}_A$ with a warp W. We note that we have a differentiable CUDA implementation of the kernel combination and application, which is required for our automatic degradation estimation. We provide additional details in the supplementary material.

4.2. Kernel Aware Resampler

An overview of the proposed image resampling model is illustrated in Figure 3. The left side of the figure illustrates the input data preparation. In the previous subsection, we have seen that sampling a lower resolution image can be parameterized with the reconstruction kernel $\mathcal{K}_R$, the warp W and the anti-aliasing kernel $\mathcal{K}_A$. Putting these together results in the resampling kernel map $\mathcal{K}_W$ which can be applied on the input image I to obtain the transformed lower resolution I'. This kernel map $\mathcal{K}_W$ is the key additional input for the resampling model.

We now have all the data needed for our kernel aware resampler. Given the input image I' the objective is to resample it according to the warp W. We simply express the warp as a 2D map of the same resolution as I, indicating the sampling position in I' for every output pixel location. We refer to this map as the warp grid $G_{W^{-1}}$. This is the most flexible option as it allows a wide range of transforms.

To reduce it's dimensionality the kernel map $\mathcal{K}_W$ is first encoded by a small MLP. Then, the input image I' and the encoded kernel map are concatenated and further processed by the ProSR network introduced in [17].

After that the resampling layer uses the warp grid $G_{W^{-1}}$ to resample the extracted features. Note that W^{-1} maps coordinates in the output image to their location in the input image. The resampling layer produces two outputs. First, for each sampling location in $G_{W^{-1}}$ it gathers the extracted features in a 3×3 neighborhood around the closest feature. Second, it computes geometric information about the warp for each sampling location in $G_{W^{-1}}$. The geometric information consists of the offset to the closest feature and the local Jacobian matrix. Further details are discussed in Section 4.3. After the resampling layer, the obtained features, now of same resolution as the target image are processed by the prediction network (see Section 4.4).

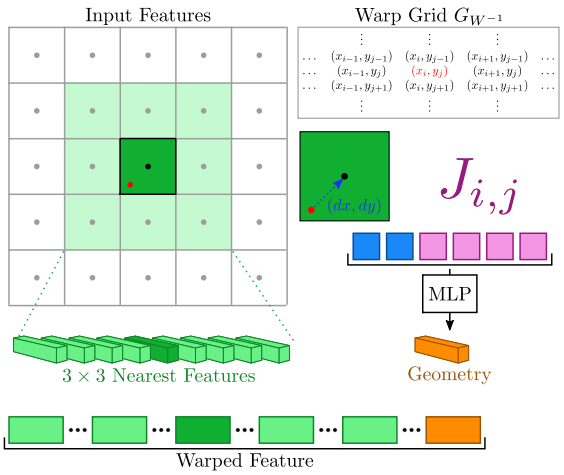

Figure 4. Details of the Resampling Layer. The warp grid $G_{W^{-1}}$ indicates the sampling position for every output pixel. The resampling layer outputs for every pixel the closest 3×3 features as well as geometric information. Geometric information consists of the sampling offsets d_x, d_y and the Jacobian matrix.

4.3. Resampling Layer

The resampling layer warps the extracted features according to $G_{W^{-1}}$. As illustrated in Figure 4, $G_{W^{-1}}$ provides for each output pixel its corresponding coordinate in the input image. The warping is done by looking up the nearest feature for each output location. The features in a 3×3 neighborhood around the nearest feature are then concatenated along the channel dimension to produce the the warped feature.

Additionally, geometric information about the warp is computed for each warped feature. The geometric information consists of the *sampling offsets* and the *Jacobian*. The sampling offset is the vector pointing from an output pixels coordinate in the input image to the coordinate of the nearest feature. This gives us a 2D vector (d_x, d_y) which describes the warped features sub-pixel location in the input image. To provide information about the local distortion, the Jacobian is computed using simple central differencing. It produces a 2×2 matrix describing the local deformation for each output location. The sampling offsets and the flattened local Jacobians are concatenated along the channel dimension and fed to a small MLP. Finally, the result of this MLP is concatenated with the warped features along the channel dimension.

4.4. Prediction Layer

After the resampling layer, we obtain a feature map of same resolution as the target. The prediction network either produces the output image colors directly or kernels to be applied to the input image. A small MLP is used for this final prediction. In the case of kernel prediction, the network produces a 5×5 kernel for each output location. This 5×5 kernel is then applied to the 5×5 neighborhood around the closest pixel in the input image. Note that per output location only a single kernel is predicted. This kernel is then applied to all channels in the input image, which allows the method to generalize to input channels that were not seen during training such as alpha, depth, normals, etc.

5. Kernel Map Estimation

The method we have described so far requires three inputs — the warp W, the input image I' and the kernel map $\mathcal{K}_W$. In a practical application, however, $\mathcal{K}_W$ is unknown and a method for estimating it is needed.

SR methods that are conditioned on the blur kernel have faced a similar problem [3]. Because the warping process described in section 3 is fully differentiable, we propose to adopt a similar strategy for blind image resampling. We start by noting that providing the incorrect degradation map $\mathcal{K}_W$ to the image resampling model produces images that are either blurry or contain artifacts such as ringing. Building on this observation we train a simple neural network model which predicts the difference between the output with a random degradation map $\mathcal{K}_{\hat{W}}$ and the result with the correct kernel $\mathcal{K}_{W_{\text{GT}}}$:

$$\mathcal{F}_E(\mathcal{F}, \mathcal{K}_W, I') = |\mathcal{F}(I', \mathcal{K}_{W_{\text{GT}}}) - \mathcal{F}(I', \mathcal{K}_W)|, \quad (5)$$

where $\mathcal{F}_E$ is the error prediction model and $\mathcal{F}$ our resampler.

Once the error prediction model is trained, it can be used at test time. We propose to modify the optimization problem originally solved in [3] for estimating the degradation map. To give more control over the resampled image to the user, we add an additional loss term to the estimation procedure. This additional loss term rewards larger kernels $\mathcal{K}_R$ and $\mathcal{K}_A$, which results in sharper images after resampling. The kernel optimization can be written as

$$\mathcal{K}_W{}^* = \arg\min_{\mathcal{K}_W} \mathcal{F}_E(\mathcal{F}, \mathcal{K}_W, I') - \alpha|\mathcal{K}_R| - \beta|\mathcal{K}_A| \ , \quad (6)$$

where $|\mathcal{K}_R|$ and $|\mathcal{K}_A|$ refer to the size of $\mathcal{K}_R$ and $\mathcal{K}_A$ respectively, and $\alpha \geq 0$ and $\beta \geq 0$ are adjustable parameters. More details about the estimator's architecture and its training procedure can be found in the supplemental material.

6. Experimental Results

We compare our method against state-of-the-art SR and resampling methods. For all experiments our method was trained on the DIV2K dataset consisting of 800 high resolution training images. We compare two different versions of our method: direct color prediction and kernel prediction. Our models are trained in a fully supervised way: First, we sample a random projective transform W which has local

	bilinear	none	so	so + J
$P_{\times 2}$	25.44	24.77	30.66	**30.95**
$P_{\times 3}$	23.36	22.68	27.14	**27.25**

Table 1. Effect of providing geometric information to the model measured in PSNR. All models are trained on projective transforms using a fixed reconstruction and anti-aliasing kernel. The second column refers to our method without access to any geometric information. The third and fourth column show our method's performance when it is given access respectively to the sampling offsets, and both the sampling offsets and the local Jacobian.

scaling factors in the range $[1, 4]$. Second, we sample two random kernels $\mathcal{K}_R$ and $\mathcal{K}_A$. Third, we sample a random high resolution image I. The kernel map $\mathcal{K}_W$ and the input image I' are then created as described in section 4.1.

6.1. Ablation Study

In this ablation study we evaluate the two key contributions of the proposed resampling model: the parametrization of the geometric transform in the model and the degradation map provided along with the image.

Local Distortion. As detailed in section 4.3 the resampling layer, in addition to warping the features, computes additional geometric information about the warp. To analyze the effect of this information we train 2 other models: one where no geometric information is provided and another where only the sampling offsets are given. For this experiment the models are trained on projective transforms using a bilinear reconstruction kernel and a bicubic anti-aliasing kernel. As the kernels are not varied during training no degradation map is provided to the models. The models are evaluated on two fixed sets of projective transforms. One with an average local scaling factor of 2 ($P_{\times 2}$), the other with an average local scaling factor of 3 ($P_{\times 3}$). Results of this evaluation are provided in Table 1. Here simple bilinear interpolation serves as a baseline. Access to geometric information helps producing better results, in particular adding the sampling offsets induces a large boost in performance. Providing the Jacobian yields a smaller but consistent improvement and is the parametrization we use.

Degradation Map. The second aspect of this ablation study is to understand the importance of providing information about the degradation process to the method. We train three models on projective transforms. The first is trained using a fixed reconstruction kernel (bilinear) and a fixed anti-aliasing kernel (bicubic). The second and third model are trained with varying reconstruction and anti-aliasing kernels but only the third model has access to the degradation map.

For the evaluation we choose two fixed sets of projective transforms. One with an average local scaling factor of 2 ($P_{\times 2}$), the other with an average local scaling factor of 3 ($P_{\times 3}$). We fix the reconstruction kernel to bilinear and choose three bicubic kernels of different sizes (small, medium, large) for the anti-aliasing kernel. The medium size kernel corresponds to the one that is used to train the model in the second column.

	bilinear	fixed deg (m)	variable deg	deg aware
$P_{\times 2}$ s	26.27	25.70	29.61	**29.96**
$P_{\times 2}$ m	25.44	**30.95**	29.69	30.33
$P_{\times 2}$ l	23.40	24.71	24.98	**27.86**
$P_{\times 3}$ s	23.89	22.48	26.15	**26.39**
$P_{\times 3}$ m	23.36	**27.25**	26.35	26.81
$P_{\times 3}$ l	21.70	22.65	22.96	**25.38**

Table 2. Effect of providing the degradation map to the model measured in PSNR. All models are trained on projective transforms. The second column refers to our model trained with fixed reconstruction and anti-aliasing kernels. The third and fourth column refer to our method trained with varying reconstruction and anti-aliasing kernels, respectively without and with access to the degradation map.

	bilinear	LIIF	SRWarp	KARL c	KARL k	KARL c est. deg	KARL k est. deg
$\times 1.5$	25.25	27.29	27.37	30.43	**30.85**	28.01	28.41
$\times 2.0, 1.5$	24.75	26.23	26.28	29.14	**29.43**	27.70	27.64
$\times 2.0$	24.37	25.53	25.49	28.56	**28.64**	27.13	27.39
$\times 2.5, 2.0$	23.32	24.22	24.27	27.36	**27.46**	25.93	26.59
$\times 2.5$	22.64	23.40	23.46	26.57	**26.64**	25.52	26.00
$\times 3.0$	21.70	22.17	22.23	25.37	**25.38**	24.23	24.86
$\times 3.5$	21.05	21.34	21.36	24.33	**24.40**	23.27	24.03
$\times 4.0$	20.79	20.97	20.88	**23.48**	**23.48**	22.00	22.13
$P_{\times 2}$	25.06	25.70	25.76	28.56	**28.78**	27.06	27.02
$P_{\times 3}$	22.77	22.45	22.70	25.53	**25.59**	24.38	24.50

Table 3. Comparison of our method to LIIF and SRWarp on a range of different upscaling and projective transformation tasks (using PSNR). Both the direct color prediction variant (KARL c) and the kernel pediction variant (KARL k) of our method are evaluated, once by providing the kernel map used to generate the low resolution images, and once by estimating it (est. deg).

Results in Table 2 show that the model trained using the medium kernel performs well when evaluated on the same settings, however, performance drops significantly on other types of blur kernels. Training a model with varying kernels (column 3) makes it more robust but it performs overall worse. Column 4 shows the results when providing the degradation map, it performs better overall and even reaches competitive results against the more specialized model on the medium size kernel.

6.2. Quantitative Results

To the best of our knowledge only SRWarp [15] targets the image resampling problem. It's the only method against which a full quantitative comparison can be run. Since arbitrary super-resolution is also part of the tasks of image resampling, we additionally compare our model against the state-of-the-art super-resolution model for non integer scaling factors LIIF [2]. LIIF can also, without major modifications, be used for projective transforms.

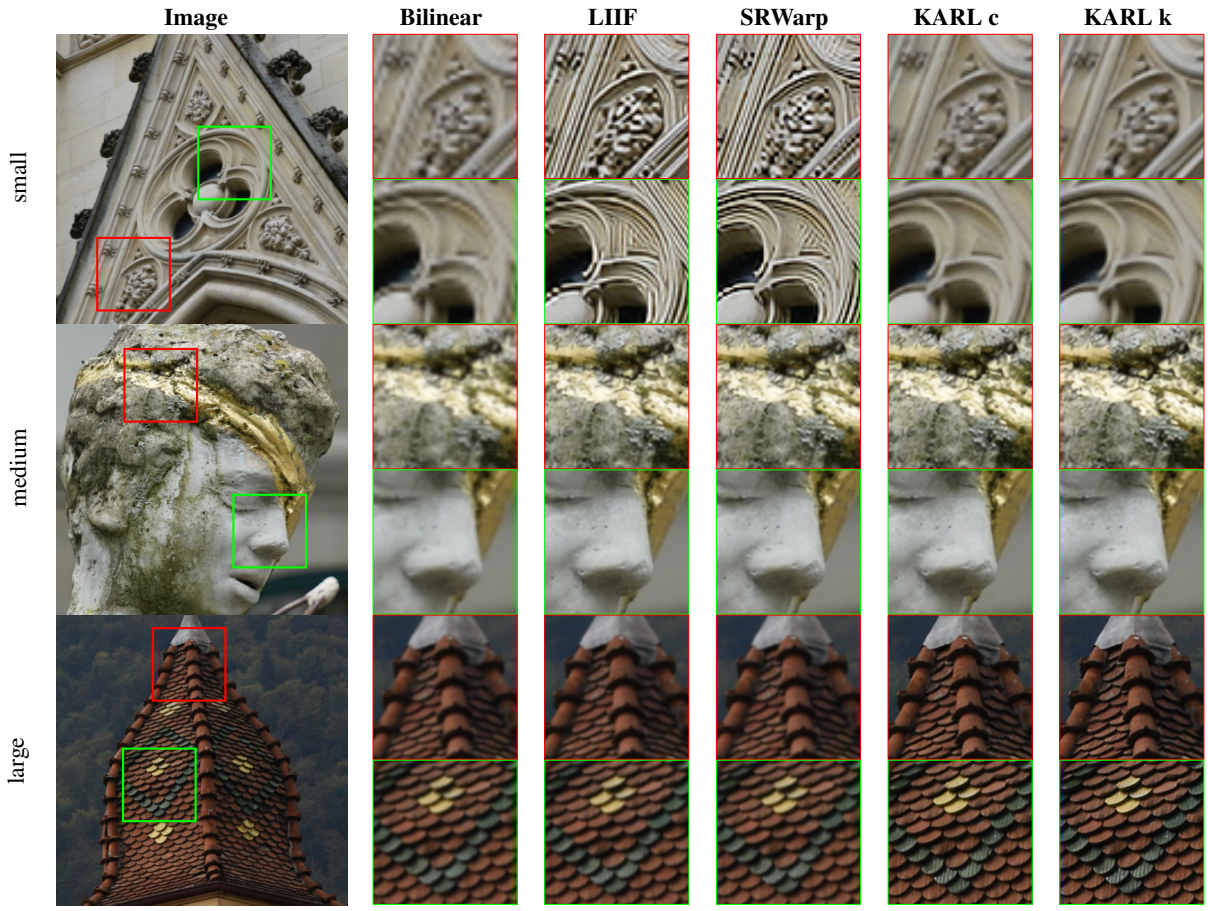

Figure 5. Comparison of our method against LIIF and SRWarp for x2 upscaling. The first row shows the results on an image that was downscaled using a small Lanczos kernel. The image in the middle row was downscaled using standard bicubic interpolation. This corresponds to the setting that LIIF and SRWarp were trained for. The last row shows the results for an image that was downscaled using a large Gaussian kernel.

All methods are evaluated on standard SR as well as projective transformations. For SR we choose a number of different scaling factors between $\times 1.5$ and $\times 4$ and a range of different reconstruction kernels. For the projective transformations we choose two sets of transforms with average local scaling factors of 2 and 3. For projective transformations we choose a range of different anti-aliasing kernels. The reconstruction kernel is fixed to bilinear.

The evaluation is done on 100 crops of size 512×512 from DIV2Ks test set (one crop per image). The crops were specifically chosen to be hard examples. A hard example is defined as one where simple bilinear upscaling produces a large error. We evaluate two versions of our method. One that predicts the output color directly (KARL c) and one that predicts a kernel to be applied to the input image (KARL k). Each method is evaluated using the correct kernel map K_W as well as using an estimated kernel map (est. deg).

Results are shown in table 3: for each type of transform the results over a wide range of different kernels are averaged. We can see that, even when the kernel map is estimated, our method manages to outperform both LIIF and SRWarp in every test case. When the correct kernel map is provided our method's performance is further improved by 1-1.5dB. A full breakdown of table 3 into the different kernels that were used can be found in the supplemental material. SRWarp and LIIF are only able to narrowly outperform our method when evaluated on the kernel that they were trained on (bicubic m).

	LIIF	SRWarp	KARLc	KARLk
Rectification	0.68 s	1.49 s	0.10 s	0.10 s
SR ($\times 2$)	0.98 s	1.08 s	0.11 s	0.10 s

Table 4. Runtime comparison on an Nvidia RTXA6000

Computational Cost Table 4 shows our methods inference time compared to SRWarp and LIIF. Due to our much leaner architecture our method enjoys significantly reduced inference times. Estimating the degradation requires and

Figure 6. Comparison on real image rectification. The unknown kernel maps are estimated for the results generated by our method. We show the results both when predicting the output color directly (KARL c) and when predicting a 5×5 kernel that is applied to the input image (KARL k). Both methods produce visually more appealing results than LIIF and SRWarp.

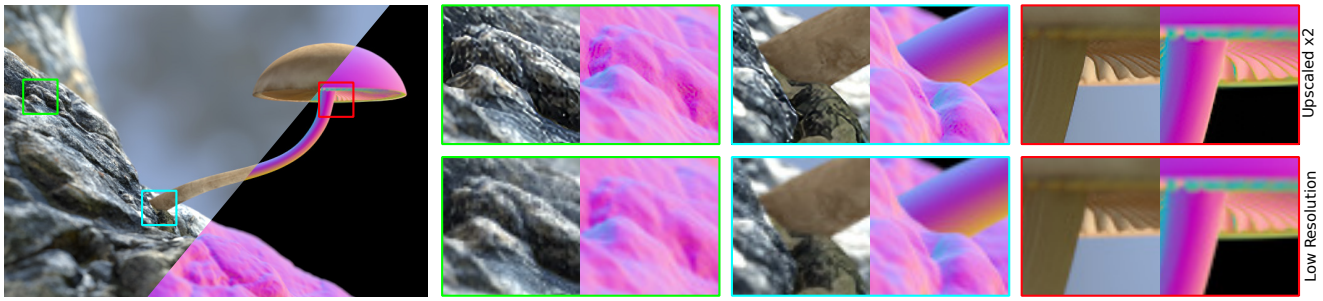

Figure 7. Upscaling a rendered image with normal information ($\times 2$). Our method predicts kernels based on the input image colors. The predicted kernels are then applied to the colors and the normals to produce the upscaled result.

additional $\sim 30s$ and $\sim 110s$ for the SR and rectification test respectively. This time is, however, amortized when working on images with shared degradation maps like video sequences.

6.3. Qualitative Results

Super-resolution. Figure 5 shows a comparison of our method against LIIF and SRWarp for the case of $\times 2$ upscaling. Here high resolution images were downscaled using blur kernels of different sizes. We can see that both LIIF and SRWarp perform well on the bicubic (medium) example but are not able to generalize to the other cases. Our method produces high quality results in all situations.

Image Rectification. Figure 1 shows an example where a natural image is rectified. As the kernel map is unknown it is estimated for our method. The KARL column shows the results of our method with degradation map estimation. Our approach produces a visually more appealing and sharper result than LIIF and SRWarp. The sharpness can be further increased and the result is shown in the last column. Figure 6 shows another rectification example of a natural image. Here we show the results of our method when predicting colors directly (KARL c) and when predicting a kernel to be applied to the input image (KARL k). Both versions produce a visually more appealing and sharper result than LIIF and SRWarp.

Rendered content. Rendered content often contains additional channels such as normals. Our methods kernel prediction variant naturally generalizes to types of data that were not seen during training. This is illustrated in figure 7, where we upscale the colors and the normals of a rendered image. In this example we feed the input image (RGB channels) into our model which produces a kernel map. This kernel map is then applied to input image to produce the upscaled image. Then, the same kernel map is applied to the input normals to produce the upscaled normals.

7. Conclusion

In this paper we have presented a deep learning framework for kernel aware image resampling, that carefully models the image warping process. Using a spatially variant degradation map as conditioning for the resampling model, we can address with the same model both global transformations, such as upscaling or rotation, and locally varying transformations such as removing lens distortion. We are also able to automatically estimate a suitable conditioning for the model to produce sharp results on complex transformations on images with unknown distortions.

Finally, we also introduce a variant that predicts kernels instead of colors. We show that this variant generalizes to image data types unseen during training, and produces best results on standard resampling tasks. Future works could improve the degradation estimation process, reducing the gap in performance between the estimated degradation and the correct one.

References

[1] Steve Bako, Thijs Vogels, Brian McWilliams, Mark Meyer, Jan Novák, Alex Harvill, Pradeep Sen, Tony DeRose, and Fabrice Rousselle. Kernel-predicting convolutional networks for denoising monte carlo renderings. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2017)*, 36(4):97:1–97:14, 2017. 3

[2] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8628–8638, 2021. 2, 6

[3] Victor Cornillère, Abdelaziz Djelouah, Wang Yifan, Olga Sorkine-Hornung, and Christopher Schroers. Blind image super resolution with spatially variant degradations. *SIGGRAPH ASIA*, 2019. 2, 4, 5

[4] Neil Anthony Dodgson. Image resampling. Technical report, University of Cambridge, Computer Laboratory, 1992. 2

[5] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015. 2

[6] Dario Fuoli, Luc Van Gool, and Radu Timofte. Fourier space losses for efficient perceptual image super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2360–2369, 2021. 2

[7] Xuecai Hu, Haoyuan Mu, Xiangyu Zhang, Zilei Wang, Tieniu Tan, and Jian Sun. Meta-sr: A magnification-arbitrary network for super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1575–1584, 2019. 2

[8] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017. 2

[9] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1833–1844, 2021. 2

[10] Jingyun Liang, Andreas Lugmayr, Kai Zhang, Martin Danelljan, Luc Van Gool, and Radu Timofte. Hierarchical conditional flow: A unified framework for image super-resolution and image rescaling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4076–4085, 2021. 2

[11] Jingyun Liang, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Mutual affine network for spatially variant kernel estimation in blind image super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4096–4105, 2021. 2

[12] Salma Abdel Magid, Yulun Zhang, Donglai Wei, Won-Dong Jang, Zudi Lin, Yun Fu, and Hanspeter Pfister. Dynamic high-pass filtering and multi-spectral attention for image super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4288–4297, 2021. 2

[13] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *arXiv preprint arXiv:2104.07636*, 2021. 2

[14] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016. 2

[15] Sanghyun Son and Kyoung Mu Lee. Srwarp: Generalized image super-resolution under arbitrary transformation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7782–7791, 2021. 2, 6

[16] Longguang Wang, Yingqian Wang, Zaiping Lin, Jungang Yang, Wei An, and Yulan Guo. Learning a single network for scale-arbitrary super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4801–4810, 2021. 2

[17] Yifan Wang, Federico Perazzi, Brian McWilliams, Alexander Sorkine-Hornung, Olga Sorkine-Hornung, and Christopher Schroers. A fully progressive approach to single-image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 864–873, 2018. 2, 4

[18] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Learning a single convolutional super-resolution network for multiple degradations. In *CVPR*, 2018. 2

[19] Pengchuan Zhang, Xiyang Dai, Jianwei Yang, Bin Xiao, Lu Yuan, Lei Zhang, and Jianfeng Gao. Multi-scale vision longformer: A new vision transformer for high-resolution image encoding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2998–3008, 2021. 2

[20] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 286–301, 2018. 2

[21] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2472–2481, 2018. 2

[22] M. Zwicker, H. Pfister, J. van Baar, and M. Gross. Ewa splatting. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):223–238, 2002. 2