Generative Adversarial Super-Resolution at the edge with knowledge distillation

GENERATIVE ADVERSARIAL SUPER-RESOLUTION AT THE EDGE WITH KNOWLEDGE DISTILLATION

Simone Angarano[1,2], **Francesco Salvetti**[1,2,3], **Mauro Martini**[1,2], **Marcello Chiaberge**[1,2]
[1]Department of Electronics and Telecommunications, Politecnico di Torino, Turin, Italy
[2]PIC4SeR PoliTo Interdepartmental Center for Service Robotics
[3]SmartData@PoliTo, Big Data and Data Science Laboratory
{name.surname}@polito.it

ABSTRACT

Single-Image Super-Resolution can support robotic tasks in environments where a reliable visual stream is required to monitor the mission, handle teleoperation or study relevant visual details. In this work, we propose an efficient Generative Adversarial Network model for real-time Super-Resolution, called EdgeSRGAN[1]. We adopt a tailored architecture of the original SRGAN and model quantization to boost the execution on CPU and Edge TPU devices, achieving up to 200 fps inference. We further optimize our model by distilling its knowledge to a smaller version of the network and obtain remarkable improvements compared to the standard training approach. Our experiments show that our fast and lightweight model preserves considerably satisfying image quality compared to heavier state-of-the-art models. Finally, we conduct experiments on image transmission with bandwidth degradation to highlight the advantages of the proposed system for mobile robotic applications.

1 Introduction

In the last decade, Deep Learning (DL) techniques have pervaded robotic systems and applications, drastically boosting automation in both perception [14, 73], navigation and control [51, 62] tasks. The development of Machine Learning driven algorithms is paving the way for advanced levels of autonomy for mobile robots, widely increasing the reliability of both unmanned aerial vehicles (UAV) and unmanned ground vehicles (UGV) [14]. Nonetheless, the adoption of mobile robots for mapping and exploration [38], search and rescue [16] or inspection [64, 63] missions in harsh unseen environments can provide substantial advantages and reduce the risks for human operators. In this context, the successful transmission of images acquired by the robot to the ground station often assumes a significant relevance to the task at hand, allowing the human operators to get real-time information, monitor the state of the mission, take critical planning decisions and analyze the scenario. Moreover, unknown outdoor environments may present unexpected extreme characteristics which still hinder the release of unmanned mobile robots in the complete absence of human supervision. Although novel DL-based autonomous navigation algorithms are currently under investigation in disparate outdoor contexts such as tunnel exploration [50, 56, 17], row-crops navigation [42, 1] and underwater [31, 4], complete or partial remote teleoperation remains the most reliable control strategy in uncertain scenarios. Indeed, irregular terrain, lighting conditions, and the loss of localization signal can lead navigation algorithms to fail. As a direct consequence of navigation errors, the robotic platform can get stuck in critical states where human intervention is required or preferred.

However, visual data transmission for robot teleoperation, monitoring, or online data processing requires a stable continuous stream of images, which may be drastically affected by poor bandwidth conditions due to the long distance of the robot or by constitutive factors of the specific environment. Besides this, UAVs and high-speed platforms require the pilot to receive the image stream at a high framerate to follow the vehicle's motion in non-line-of-sight situations. A straightforward but effective solution to mitigate poor bandwidth conditions and meet high-frequency transmission requirements is reducing the transmitted image's resolution. On the other hand, heavy image compression with massive loss of detail can compromise image usability.

To this end, we propose EdgeSRGAN, a novel deep learning model for Single-Image Super-Resolution (SISR) at the edge to handle the problem of efficient image transmission. Our intuition relies on a lightweight neural network

[1]Code available at https://github.com/PIC4SeR/EdgeSRGAN.

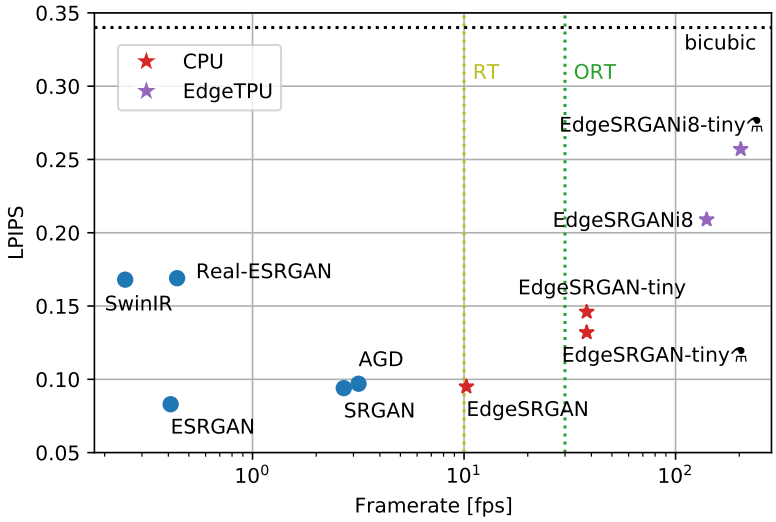

Figure 1: LPIPS [68] results (lower is better) on Set5 [7] vs framerate (80×60 input) of different visual-oriented SISR methods for $\times 4$ upsampling. Real-time (RT) and over-real-time (ORT) framerates are marked as references. Our models, marked with $\star$, reach real-time performance with a competitive perceptual similarity index on the CPU. Edge TPU models can further increase inference speed far beyond real-time, still outperforming the bicubic baseline.

allowing us to send low-resolution images at a high transmission rate with scarce bandwidth and then reconstruct the high-resolution image on the pilot's mobile device. Moreover, the successful spread of edge-AI in different engineering applications [12, 5, 36] has shown encouraging results in moving the execution of DL models on ultra-low power embedded devices. Hence, we propose an edge-AI computationally efficient Super Resolution neural network to provide fast inference on CPUs and Edge TPU devices. To this aim, we adopt several optimization steps to boost the performance of our model while minimizing the quality drop. We refine the architecture of the original SRGAN [30] to speed up inference and perform model quantization. Nonetheless, we experiment with a teacher-student knowledge distillation technique for SISR to further enhance the reconstructed image of our tiny model. We take inspiration from the work of [21] and obtain a remarkable improvement for all the considered metrics.

We perform experiments to validate the proposed methodology under multiple perspectives: numerical and qualitative analysis of our model reconstructed images and inference efficiency on both CPU and Edge TPU devices. As an example, as shown in Fig. 1, EdgeSRGAN achieves real-time performance with a competitive perceptual similarity index compared with other visual-oriented SISR methods. Moreover, we test the performance of our system for robotic applications. In particular, we focus on image transmission for teleoperation in case of bandwidth degradation, also performing tests with the popular robotic middleware ROS2.

The rest of the paper is organized as follows. In Section 2, we introduce the research landscape of Super-Resolution (SR), starting from the general background and then deepening the discussion towards robotic applications of SR and efficient SR methods presented in previous works. In Section 3, we describe the Super-Resolution problem and our methodological steps to obtain an Edge AI implementation for real-time performances. In Section 4, we propose a wide range of experiments to validate the proposed methodology, analyzing the results obtained for inference speed and output image quality and characterizing the advantages of our approach for robotic applications in limited-bandwidth conditions. Finally, in Section 5, we summarize the overall study with conclusive remarks and suggest some potential future work directions.

2 Related Works

2.1 Single-Image Super-Resolution

Single-Image Super-Resolution, also referred to as super-sampling or image restoration, aims at reconstructing a high-resolution (HR) image starting from a single low-resolution (LR) input image, trying to preserve details and the information conceived by the image. Therefore SISR, together with image denoising, is an ill-posed underdetermined

inverse problem since a multiplicity of possible solutions exist given an input low-resolution image. Recently, learning-based methods have rapidly reached state-of-the-art performance and are universally recognized as the most popular approach for Super-Resolution. Such approaches rely on learning common patterns from multiple LR-HR pairs in a supervised fashion. SRCNN [15] was the first example of a CNN applied to single-image super-resolution in literature. It has been followed by multiple methods applying standard deep learning methodologies to SISR, such as residual learning [29, 35], dense connections [71], residual feature distillation [37], attention [70, 13, 45], self-attention, and transformers [9, 11, 34]. All these works focus on content-based SR, in which the objective is to reconstruct an image with high pixel fidelity, and the training is based on a content loss, such as mean square error or mean absolute error.

In parallel, other works proposed Generative Adversarial Networks (GAN) [19] for SISR to aim at reconstructing visually pleasing images. In this case, the focus is not on pixel values but perceptual indexes that try to reflect how humans perceive image quality. This is usually implemented using perceptual losses and adversarial training and is referred to as visual-based SR. SRGAN [30] first proposed adversarial training and was later followed by other works [35, 18, 59]. With robotic image transmission as a target application in mind, in this work, we particularly focus on visual-based SR, aiming to reconstruct visually pleasing images to be used by human operators for real-time teleoperation and monitoring.

2.2 Efficient Methods for Single-Image Super-Resolution

In recent years, efficient deep neural networks for SR have been proposed to reduce the number of parameters while keeping high-quality performances [33]. However, most of the proposed architectural solutions are designed for content-based training, which aims to minimize the difference between the high-resolution image and the network output. Among them, [52] proposed a thin, simple model which handles SR as a bilinear upsampling residual compensation. Despite the high-quality images obtained, this approach has high inference latency due to the double prediction required. Diversely, [44] entirely based their study to target Edge-AI chips, proposing an ultra-tiny model composed of one layer only.

As already stated, we prefer GAN-based SR to enhance the visual appearance of produced images for robotic applications. However, successful studies of efficient GANs are very rare in the literature. Recently, knowledge distillation (KD) emerged as a promising option to compress deep models and GANs too [2, 20]. KD was originally born in 2015 with the visionary work of [25], where a teacher-student framework was introduced as a knowledge transfer mechanism. More recent works evolved such concept in disparate variants: FitNets [49] introduced the idea of involving also intermediate representations in the distillation process, Attention Transfer (AT) [65] proposes an attention-based distillation, and Activation Boundaries (AB) [24] interestingly focuses on the distilled transfer of activation boundaries formed by hidden neurons, further advanced in [23]. Specifically considering KD application in SR, Feature Affinity KD (FAKD) [21] uses intermediate features affinity distillation for PSNR-focused SR. We found this approach a good starting point also for GAN-based SR. Diversely, [69] investigates a progressive knowledge distillation method for data-free training. Besides KD, [18] recently proposed an Automated Machine Learning (Auto-ML) framework to search for optimal neural model structure, and filter pruning has been used as another optimization technique [32].

Differently from previous works, our model optimization for edge-SR is composed of three main steps: first, an edge-oriented architectural definition is performed; then, we leverage teacher-student knowledge distillation to further reduce the dimension of our model; lastly, we perform TensorFlow Lite (TFLite) conversion and quantization to shift the network execution to CPUs and Edge TPUs with maximum inference speed.

2.3 Super-Resolution for Robotic Applications

SISR has been recently proposed in a few robotic applications where a high level of detail is beneficial to support the specific task. Research on the indoor teleoperation of mobile robots mainly focuses on improving user experience, combining Deep Learning methods with Virtual Reality [66, 22, 55], but neglecting the potential bottleneck caused by connectivity degradation in harsh conditions. Differently, a great effort has been devoted to SISR for underwater robotics perception [47, 27], effectively tackling the problem of high-quality image acquisition under the sea for accurate object and species detection. Besides autonomous navigation applications, interesting contexts are robotic surgery [58, 8] and medical robots research [41], where SISR can provide substantial advantages improving the visibility and increasing the level of detail required for delicate high-precision movements of the surgeon. Similarly, a detailed image acquired by a robot is needed for monitoring and inspection purposes. For example, [6] uses a Super-Resolution model to enhance the online crack detection and in-situ analysis of bridge weaknesses. Nonetheless, no relevant works proposed so far have identified Super-Resolution as an efficient solution for image transmission to support robot teleoperation and exploration of unknown environments in bandwidth-degraded conditions.

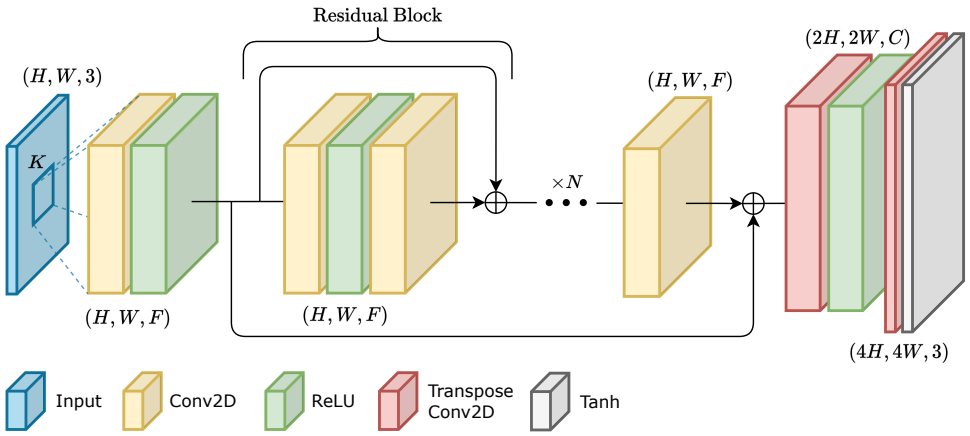

Figure 2: EdgeSRGAN Generator Architecture.

3 Methodology

In this section, we introduce all the components of the proposed methodology. As explained in Section 1, we choose to use an adversarial approach to obtain an optimal balance between pixel-wise fidelity and perceptual quality. For this reason, we take inspiration from three of the most popular GAN-based solutions for SISR: SRGAN [30], ESRGAN [61], and AGD [18]. The proposed method aims to obtain a real-time SISR model (EdgeSRGAN) with minimal performance drop compared to state-of-the-art solutions. For this reason, we mix successful literature practices with computationally-efficient elements to obtain a lightweight architecture. Then, we design the network training procedure to leverage a combination of pixel-wise loss, perceptual loss, and adversarial loss. To further optimize the inference time, we apply knowledge distillation to transfer the performance of EdgeSRGAN to an even smaller model (EdgeSRGAN-tiny). Furthermore, we study the effect of quantization on the network's latency and accuracy. Finally, we propose an additional inference-time network interpolation feature to allow real-time balancing between pixel-wise precision and photo-realistic textures.

3.1 Network Architecture

As previously done by [61], we take the original design of SRGAN and propose some changes to both the architecture and training procedure. However, in our case, the modifications seek efficiency as well as performance. To obtain a lighter architecture, we reduce the depth of the model by using only $N = 8$ Residual Blocks instead of the original 16. In particular, we use simple residuals instead of the Residual-in-Residual Dense Blocks (RRDB) proposed by [61] as they are less computationally demanding. For the same reason, we change PReLU activation functions into basic ReLU. We also remove Batch Normalization to allow the model for better convergence without generating artifacts [61]. Finally, we use Transpose Convolution for the upsampling head instead of Sub-pixel Convolution [53]. Despite its popularity and effectiveness, Sub-pixel Convolution is computationally demanding due to the Pixel Shuffling operation, which rearranges feature channels spatially. We choose instead to trade some performance for efficiency and apply Transpose Convolutions taking precautions to avoid problems such as checkerboard artifacts [46]. The complete EdgeSRGAN architecture is described in Fig. 2. The adopted discriminator model is the same used in [30, 61], as it serves only training purposes and is not needed at inference time. Its architecture is described in Fig. 3.

3.2 Training Methodology

The training procedure is divided into two sections, as it is common practice in generative adversarial SISR. The first part consists of classic supervised training using pixel-wise loss. In this way, we help the generator to avoid local minima and generate visually pleasing results in the subsequent adversarial training. We use the mean absolute error (MAE) loss for the optimization as it has recently proven to bring better convergence than mean squared error (MSE) [72, 35, 70, 61].

$$L_{\text{MSE}} = \sum_{i=1}^{B} ||y_i^{\text{HR}} - y_i^{\text{SR}}||_1 \tag{1}$$

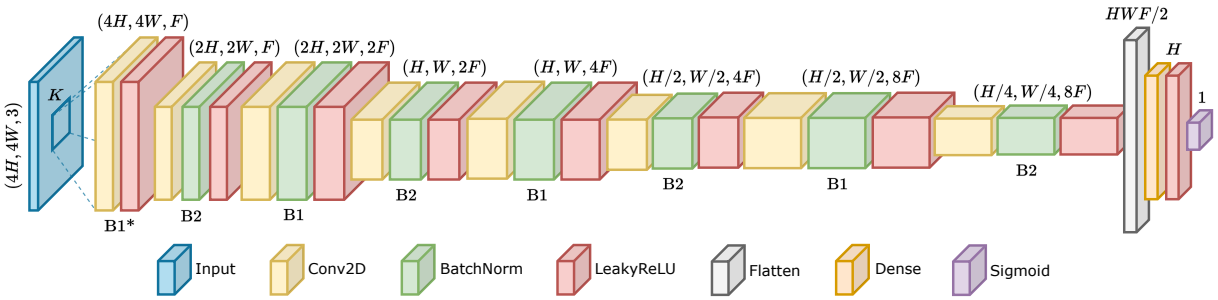

Figure 3: EdgeSRGAN Discriminator Architecture. The model progressively reduces the spatial dimensions of the image by alternating blocks with strides 1 (B1) and 2 (B2). The first block (marked with *) does not apply batch normalization.

where y^{HR} is the ground-truth high resolution image, y^{SR} is the output of the generator, and B is the batch size. We use the Peak Signal-to-Noise Ratio (PSNR) metric to validate the model.

In the second phase, the resulting model is fine-tuned in an adversarial fashion, optimizing a loss that takes into account adversarial loss and perceptual loss. As presented in [30], the generator G training loss can be formulated as

$$L_G = L_G^P + \xi L_G^A + \eta L_{\text{MSE}}. \tag{2}$$

L_G^P is the perceptual VGG54 as the euclidean distance between the feature representations of a reconstructed image SR and the reference image HR. The features are extracted using the VGG19 network [54] pre-trained on ImageNet:

$$L_G^P = \sum_{i=1}^{B} ||\phi(y_i^{\text{HR}}) - \phi(y_i^{\text{SR}})||_2 \tag{3}$$

where ϕ is the perceptual model VGG. L_G^A is the adversarial generator loss, defined as

$$L_G^A = -\log(D(y_{\text{SR}})) \tag{4}$$

where D is the discriminator. Using this loss, the generator tries to fool the discriminator by generating images that are indistinguishable from the real HR ones. ξ and η are used to balance the weight of different loss components. The weights of the discriminator D are optimized using a symmetrical adversarial loss, which tends to correctly discriminate HR and SR images.

$$L_D = \log(D(y_{\text{SR}})) - \log(D(y_{\text{HR}})) \tag{5}$$

We optimize both models simultaneously, without alternating weight updates like in most seminal works on GANs. The overall training methodology is summarized in Fig. 4 summarizes the overall training methodology.

3.3 Knowledge Distillation

As mentioned in Section 2, Knowledge Distillation (KD) has gained increasing interest in deep learning for its ability to transfer knowledge from bigger models to simpler ones efficiently. In particular, KD has been applied in some SISR works to compress the texture reconstruction capability of cumbersome models and obtain efficient real-time networks. However, to the best of our knowledge, KD has never been applied to GAN SISR models. For this reason, we adapt an existing technique developed for SISR called Feature Affinity-based Knowledge Distillation (FAKD) [21] to the GAN training approach. The FAKD methodology transfers second-order statistical info to the student by aligning feature affinity matrices at different layers of the networks. This constraint helps to tackle the fact that regression problems generate unbounded solution spaces. Indeed, most of the KD methods so far have only tackled classification tasks. Given a layer l of the network, the feature map F_l extracted from that layer (after the activation function) has the following shape:

$$F_l \in \mathbb{R}^{B \times C \times W \times H} \tag{6}$$

where B is the batch size, C is the number of channels, W and H are the width and the height of the tensor. We first flatten the tensor along the last two components obtaining the three-dimensional feature map

$$F_l \in \mathbb{R}^{B \times C \times WH} \tag{7}$$

which now holds all the spatial information along a single axis. We define the affinity matrix A_l as the product

$$A_l = \tilde{F_l}^{\top} \cdot \tilde{F_l} \tag{8}$$

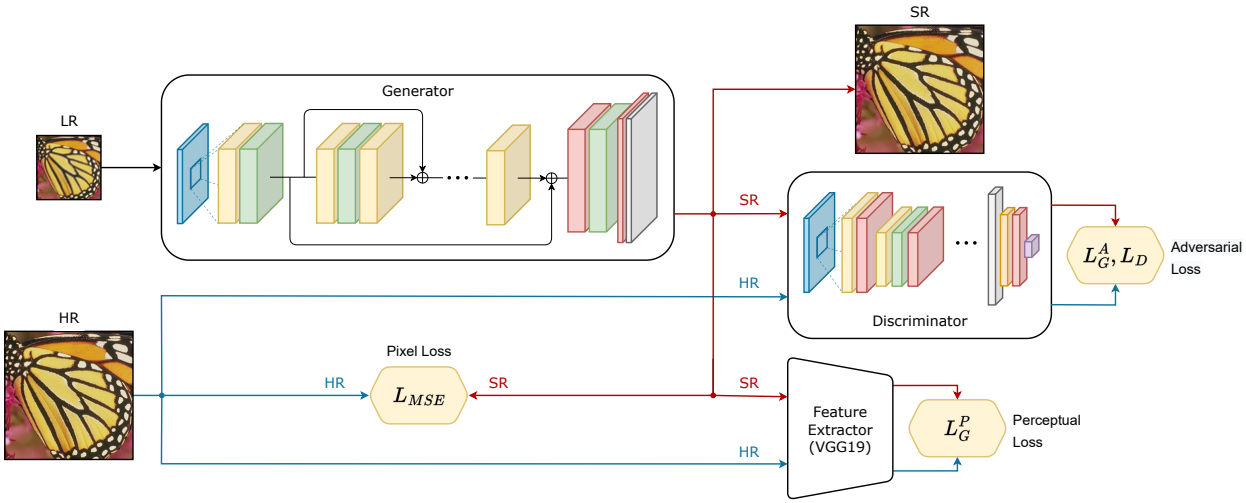

Figure 4: EdgeSRGAN Training Methodology.

where $\cdot$ is the matrix multiplication operator and the transposition $\top$ swaps the last two dimensions of the tensor. $\tilde{F}_l$ is the normalized feature map, obtained as

$$\tilde{F}_l = \frac{F_l}{||F_l||_2} \tag{9}$$

Differently from [21], the norm is calculated for the whole tensor and not only along the channel axis. Moreover, we find better convergence using the euclidean norm instead of its square. In this way, the affinity matrix has a shape

$$A_l \in \mathbb{R}^{B \times WH \times WH} \tag{10}$$

and the total distillation loss L_{Dist} becomes

$$L_{\text{Dist}} = \frac{1}{N_L}\left(\sum_{l=1}^{N_L} ||A_l^T - A_l^S||_1\right) + \lambda ||y_{\text{SR}}^T - y_{\text{SR}}^S||_1 \tag{11}$$

where N_L is the number of distilled layers. Differently from [21], we sum the loss along all the tensor dimensions and average the result obtained for different layers. These modifications experimentally lead to better training convergence. We also add another loss component, weighted by λ, which optimizes the model to generate outputs close to the teacher's ones. In our experimentation, the distillation loss is added to the overall training loss weighted by the parameter γ. The overall distillation scheme is summarized in Fig. 5.

3.4 Model Interpolation

Following the procedure proposed in [61], we adopt a flexible and effective strategy to obtain a tunable trade-off between a content-oriented and GAN-trained model. This feature can be very useful for real-time applications, as it allows the SISR network to adapt to the user's needs promptly. Indeed, some real scenarios may need better perceptual quality, for example, when the remote control of a robot has to be performed by a human pilot. On the other hand, when images are used to directly feed perception, autonomous navigation, and mapping algorithms, higher pixel fidelity might be beneficial. To achieve this goal, we linearly interpolate model weights layer-by-layer, according to the following formula:

$$\theta_G^{\text{Int}} = \alpha \theta_G^{\text{PSNR}} + (1 - \alpha)\theta_G^{\text{GAN}} \tag{12}$$

where θ_G^{Interp}, θ_G^{PSNR}, and θ_G^{GAN} are the weights of the interpolated model, the PSNR model, and the GAN fine-tuned model, respectively. $\alpha \in [0, 1]$ is the interpolation weight. We report both qualitative and quantitative interpolation results for EdgeSRGAN in Section 4.3.1. We avoid the alternative technique of directly interpolating network outputs: applying this method in real time would require running two models simultaneously. Moreover, Wang *et al.*[61] report that this approach does not guarantee an optimal trade-off between noise and blur.

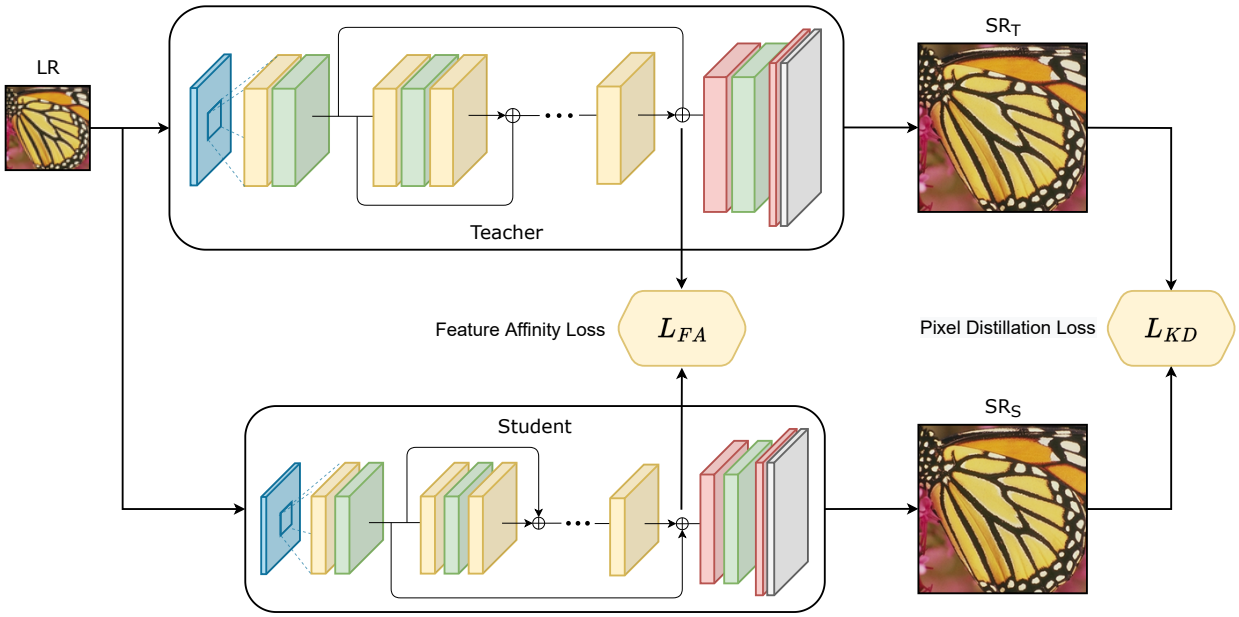

Figure 5: EdgeSRGAN Distillation Process.

3.5 Model Quantization

To make EdgeSRGAN achieve even lower inference latency, we apply optimization methods to the model to reduce the computational effort at the cost of a loss in performance. Several techniques have been developed to increase model efficiency in the past few years [28], from which the employed method is chosen. We reduce the number of bits used to represent network parameters and activation functions with TFLite[2]. This strategy strongly increases efficiency with some impact on performance. We quantize weights, activations, and math operations through scale and zero-point parameters following the methodology presented by Jacob *et al.*[28]:

$$r = S(q - Z) \tag{13}$$

where r is the original floating-point value, q is the quantized integer value, and S and Z are the quantization parameters (scale and zero point). A fixed-point multiplication approach is adopted to cope with the non-integer scale of S. This strategy drastically reduces memory and computational demands due to the high efficiency of integer computations on microcontrollers. For our experimentation, we deploy the quantized model on a Google Coral Edge TPU USB Accelerator[3].

4 Experiments

4.1 Experimental Setting

In this section, we define our method's implementation details and the procedure we followed to train and validate the efficiency of EdgeSRGAN optimally. As previously done by most GAN-based SISR works, we train the network on the high-quality DIV2K dataset [3] with a scaling factor of 4. The dataset contains 800 training samples and 100 validation samples. We train our model with input images of size 24x24 pixels, selecting random patches from the training set. We apply data augmentation by randomly flipping or rotating the images by multiples of $90°$. We adopt a batch size of 16.

For the standard EdgeSRGAN implementation, we choose $N = 8$, $F = 64$, $K = 3$, and $D = 1024$, obtaining a generator with around 660k parameters and a discriminator of over 23M (due to the fully-connected head). The discriminator is built with $F = 64$, $K = 3$, $D = 512$, and with a coefficient for LeakyReLU $\alpha = 0.2$. We first train EdgeSRGAN pixel-wise for 5×10^5 steps with Adam optimizer and a constant learning rate of 1×10^{-4}. Then, the model is fine-tuned in the adversarial setting described in Section 3 for 1×10^5 steps. Adam optimizer is used for the

[2] `https://www.tensorflow.org/lite/`
[3] `https://coral.ai/`

			Framerate (80×60) [fps]		**Framerate** (160×120) [fps]	
Method	**Scale**	**Params**	**CPU**	**EdgeTPU**	**CPU**	**EdgeTPU**
SwinIR [34]		11.9M	0.25 ± 0.01	-	0.06 ± 0.01	-
ESRGAN [61]		16.7M	0.40 ± 0.01	-	0.10 ± 0.01	-
Real-ESRGAN [59]		16.7M	0.44 ± 0.01	-	0.11 ± 0.01	-
SRGAN [30]	$\times 4$	1.5M	2.70 ± 0.08	-	0.95 ± 0.02	-
AGD [18]		0.42M	3.17 ± 0.12	-	0.88 ± 0.01	-
EdgeSRGAN		0.66M	10.26 ± 0.11	140.23 ± 1.50	2.66 ± 0.02	10.63 ± 0.03
EdgeSRGAN-tiny		0.09M	37.99 ± 1.42	203.16 ± 3.03	11.76 ± 0.20	20.57 ± 0.05
SwinIR [34]		12.0M	0.23 ± 0.01	-	0.06 ± 0.01	-
EdgeSRGAN	$\times 8$	0.71M	7.70 ± 0.31	14.26 ± 0.06	1.81 ± 0.04	-
EdgeSRGAN-tiny		0.11M	24.53 ± 1.28	41.55 ± 0.38	5.81 ± 0.29	-

Table 1: Framerate comparison of different methods for $\times 4$ and $\times 8$ upsampling, with two different input resolutions (80×60 and 160×120). The results are provided as mean and standard deviation of 10 independent experiments of 100 predictions each. Current content-oriented SISR state-of-art method SwinIR [34] is reported as a reference. Real-time and over-real-time framerates are in blue and red, respectively. The proposed solution is the only one compatible with EdgeTPU devices and allows reaching real-time performance in both conditions.

generator and the discriminator with a learning rate of 1×10^{-5}, further divided by 10 after 5×10^4 steps. For the loss function, we set $\xi = 1 \times 10^{-3}$ and $\eta = 0$.

To obtain an even smaller model for our distillation experiments, we build EdgeSRGAN-tiny by choosing $N = 4$, $F = 32$, and $D = 256$. We further shrink the size of the discriminator by eliminating the first compression stage ($B1$) from each block (see Fig. 3). In this configuration, we also remove the batch normalization layer from the first B2 block to be coherent with the larger version. The obtained generator and discriminator contain around 90k and 2.75M parameters. The pre-training procedure is the one described for EdgeSRGAN, while the adversarial training is performed with the additional distillation loss ($\gamma = 1 \times 10^{-2}$, $\lambda = 1 \times 10^{-1}$) of Eq. 11. EdgeSRGAN is used as a teacher model, distilling its layers 2, 5, and 8 into EdgeSRGAN-tiny's layers 1, 2, and 4. The model is trained with a learning rate of 1×10^{-4}, which is further divided by 10 after 5×10^4 steps. For the loss function, we set $\xi = 1 \times 10^{-3}$ and $\eta = 0$.

Finally, we create a third version of our model to upscale images with a factor of 8. To do so, we change the first transpose convolution layer of EdgeSRGAN and EdgeSRGAN-tiny to have a stride of 4 instead of 2 and leave the rest of the architecture unchanged. The training procedure for these models is analogous to the ones used for the x4 models, with the main difference of adding a pixel-based component to the adversarial loss by posing $\eta = 1 \times 10^2$.

The optimal training hyperparameters are found by running a random search and choosing the best-performing models on DIV2K validation. During GAN training, we use PSNR to validate the models during content-based loss optimization and LPIPS [68] (with AlexNet backbone).

We employ TensorFlow 2 and a workstation with 64 GB of RAM, an Intel i9-12900K CPU, and an Nvidia 3090 RTX GPU to perform all the training experiments.

4.2 Real-time Performance

Since the main focus of the proposed methodology is to train an optimized SISR model to be efficiently run at the edge in real time, we first report an inference speed comparison between the proposed method and other literature methodologies. All the results are shown in Tab. 1 as the mean and standard deviation of 10 independent experiments of 100 predictions each. We compare the proposed methodology with other GAN-based methods [30, 61, 59, 18] and with the current state-of-the-art in content-oriented SISR SwinIR [34]. Since the original implementations of the GAN-based solutions consider $\times 4$ upsampling only, for the $\times 8$ comparison, we only report SwinIR. We select two different input resolutions for the experimentation, (80×60) and (160×120), in order to target (320×240) and (640×480) resolutions for $\times 4$ upsampling and (640×480) and (1280×960) for $\times 8$ upsampling, respectively. This choice is justified because (640×480) is a standard resolution provided by most cameras' native video stream. We also report the number of parameters for all the models.

For all the considered methods, we measure the CPU timings with the model format of the original implementation (PyTorch or TensorFlow) on a MacBook Pro with an Intel i5-8257U processor. The concept of real-time performance strongly depends on the downstream task. For robotic monitoring and teleoperation, we consider 10 fps as the minimum real-time framerate, considering over-real-time everything above 30 fps, which is the standard framerate for most commercial cameras. The proposed methodology outperforms all the other methods in inference speed and achieves real-time performance on the CPU in almost all the testing conditions. It is worth noting that AGD is specifically

	Set5 [7]			**Set14** [67]			**BSD100** [40]			**Manga109** [43]			**Urban100** [26]		
Method	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓
Bicubic	28.632	0.814	0.340	26.212	0.709	0.441	26.043	0.672	0.529	25.071	0.790	0.318	23.236	0.661	0.473
SwinIR [34]	32.719	0.902	0.168	28.939	0.791	0.268	27.834	0.746	0.358	31.678	0.923	0.094	27.072	0.816	0.193
SRGAN [30]	32.013	0.893	0.191	28.534	0.781	0.294	27.534	0.735	0.396	30.292	0.906	0.111	25.959	0.782	0.244
ESRGAN [61]†	32.730	0.901	0.181	28.997	0.792	0.275	27.838	0.745	0.371	31.644	0.920	0.097	27.028	0.815	0.201
AGD [18]	31.708	0.889	0.178	28.311	0.775	0.291	27.374	0.729	0.385	29.413	0.897	0.118	25.506	0.767	0.250
EdgeSRGAN	31.729	0.889	0.191	28.303	0.774	0.301	27.359	0.728	0.405	29.611	0.897	0.120	25.469	0.764	0.266
EdgeSRGAN-tiny	30.875	0.873	0.204	27.796	0.761	0.320	26.999	0.717	0.418	28.233	0.871	0.163	24.695	0.733	0.325

Table 2: Quantitative comparison of different methods for content-oriented $\times 4$ upsampling. Current SISR state-of-art method SwinIR [34] and bicubic baseline are reported as reference.
↑: higher is better, ↓: lower is better, †: trained on DIV2K [3] + Flickr2K [57] + OST [60]

	Set5 [7]			**Set14** [67]			**BSD100** [40]			**Manga109** [43]			**Urban100** [26]		
Model	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓
Bicubic	28.632	0.814	0.340	26.212	0.709	0.441	26.043	0.672	0.529	25.071	0.790	0.318	23.236	0.661	0.473
SwinIR [34]	32.719	0.902	0.168	28.939	0.791	0.268	27.834	0.746	0.358	31.678	0.923	0.094	27.072	0.816	0.193
SRGAN [30]	29.182	0.842	0.094	26.171	0.701	0.172	25.447	0.648	0.206	27.346	0.860	0.076	24.393	0.728	0.158
ESRGAN[61]†	30.459	0.852	0.083	26.283	0.698	0.139	25.288	0.649	0.168	28.478	0.860	0.065	24.350	0.733	0.125
Real-ESRGAN [59]†	26.617	0.807	0.169	25.421	0.696	0.234	25.089	0.653	0.282	25.985	0.836	0.149	22.671	0.686	0.214
AGD [18]	30.432	0.861	0.097	27.276	0.739	0.160	26.219	0.688	0.214	28.163	0.870	0.076	24.732	0.743	0.170
EdgeSRGAN	29.487	0.837	0.095	26.814	0.715	0.176	25.543	0.644	0.210	27.679	0.855	0.081	24.268	0.716	0.170
EdgeSRGAN-tiny	28.074	0.803	0.146	26.001	0.702	0.242	25.526	0.658	0.292	25.655	0.804	0.140	23.332	0.672	0.269
EdgeSRGAN-tiny⚗	29.513	0.841	0.132	26.950	0.727	0.220	26.174	0.673	0.282	27.106	0.845	0.130	24.117	0.704	0.249

Table 3: Quantitative comparison of different methods for visual-oriented $\times 4$ upsampling. Current SISR state-of-art method SwinIR [34] and bicubic baseline are reported as reference. ↑: higher is better, ↓: lower is better. †: trained on DIV2K [3] + Flickr2K [57] + OST [60].

designed to reduce latency for GAN-based SR and has fewer parameters than EdgeSRGAN, but it still fails at achieving real-time without a GPU.

In addition, we report the framerate of the EdgeSRGAN int8-quantized models on an EdgeTPU Coral USB Accelerator. The proposed solution is the only one compatible with such devices and allows reaching over-real-time performance for (80×60) input resolution. It must be underlined how the $\times 8$ models with (160×120) input resolution cannot target the EdgeTPU device due to memory limitations.

4.3 Super-Resolution Results

To present quantitative results on image super-resolution, we refer to content-oriented SR for models trained with content-based loss only and visual-oriented SR for models trained with adversarial and perceptual losses. Content-based loss (mean absolute error or mean squared error) aims to maximize PSNR and SSIM, while adversarial and perceptual losses aim to maximize visual quality. We test EdgeSRGAN models on five benchmark datasets (Set5 [7], Set14 [67], BSD100 [40], Manga109 [43], and Urban100 [26]) measuring PSNR, SSIM, and LPIPS. We follow the standard procedure for SISR adopted in [34], where the metrics are computed on the luminance channel Y of the YCbCr converted images. Also, S pixels are cropped from each image border, where S is the model scale factor.

Tab. 2 and Tab. 3 show the comparison with other methods for content-oriented and visual-oriented $\times 4$ SR, respectively. We report results of other GAN-based methodologies [30, 61, 59, 18] as well as the current content-oriented SOTA SwinIR [34] and bicubic baseline, as reference. Unlike what is usually found in literature, we refer to the OpenCV[4] bicubic resize implementation instead of the one present in MATLAB. For visual-oriented SR, we also report the results of the distilled tiny model EdgeSRGAN-tiny⚗. The proposed method reaches competitive results in all the metrics, even with some degradation for tiny models due to the considerable weight reduction. The distillation method helps EdgeSRGAN-tiny training by transferring knowledge from the standard model and decreasing the degradation due to the reduced number of parameters. Note that ESRGAN and RealESRGAN are trained on Flickr2K [57], and OST [60] datasets in addition to DIV2K. Tab. 4 reports results of the $\times 8$ models, together with SwinIR and bicubic. Also, in this case, the proposed models reach competitive results, and knowledge distillation helps to reduce performance degradation in the tiny model. As a final qualitative evaluation, Fig. 6 compares the super-resolved images obtained by EdgeSRGAN with the considered state-of-the-art solutions. Our model shows comparable results, highlighting more texture and details than networks trained with pixel loss (L_{MSE}) while remaining true to the ground truth image.

[4] `https://docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html#resize`

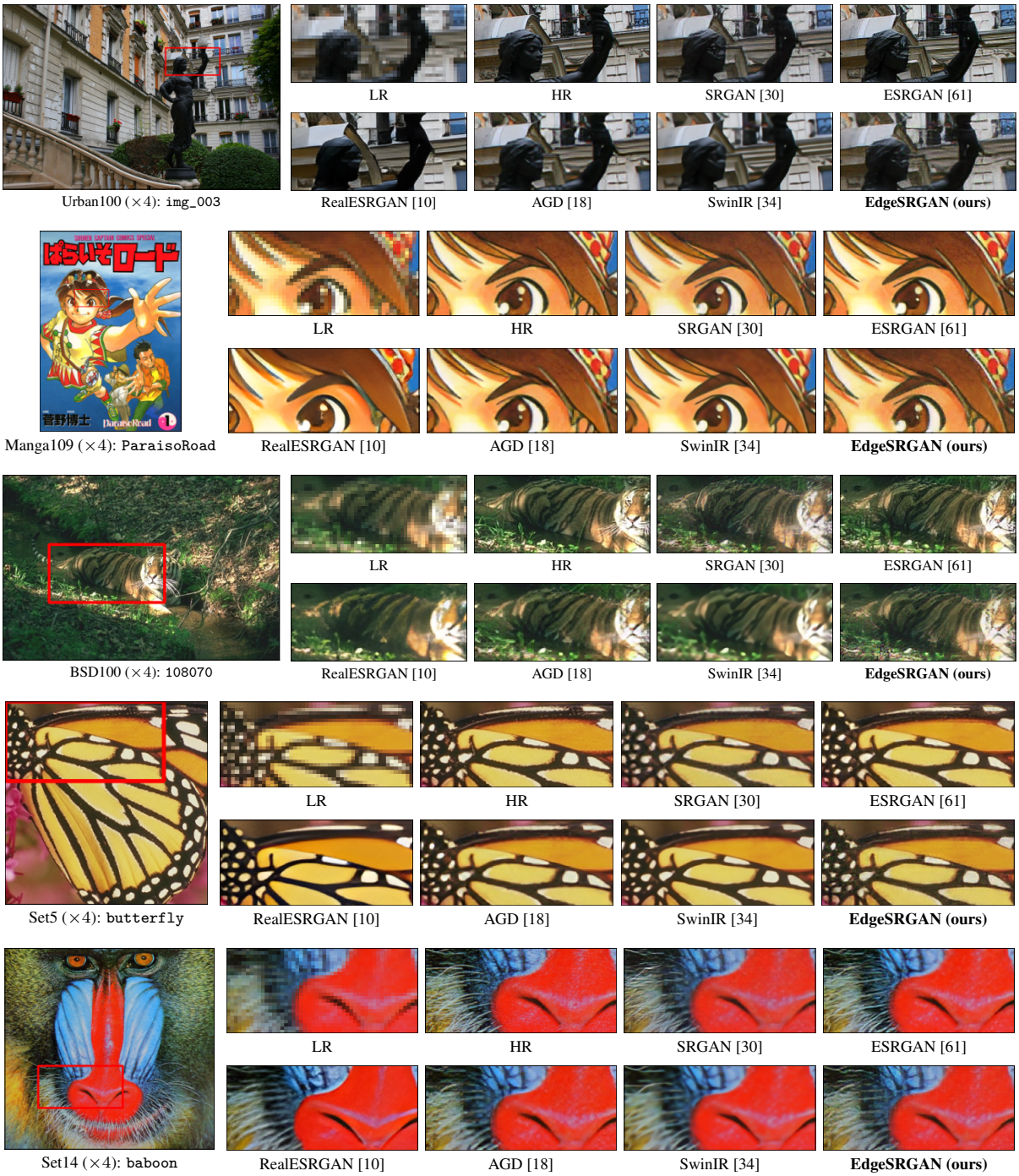

Figure 6: Visual comparison of bicubic image SR (×4) methods on random samples from the considered datasets. EdgeSRGAN achieves results that are comparable to state-of-the-art solutions with $\sim 10\%$ of the weights.

		Set5 [7]			Set14 [67]			BSD100 [40]			Manga109 [43]			Urban100 [26]		
Model		**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓
Bicubic		24.526	0.659	0.533	23.279	0.568	0.628	23.727	0.546	0.713	21.550	0.646	0.535	20.804	0.515	0.686
SwinIR [34]		27.363	0.787	0.284	25.265	0.652	0.428	24.984	0.606	0.537	25.246	0.800	0.229	23.023	0.646	0.375
EdgeSRGAN	content	26.462	0.755	0.321	24.507	0.626	0.460	24.590	0.587	0.567	23.840	0.753	0.294	22.001	0.592	0.463
EdgeSRGAN-tiny	content	26.025	0.732	0.359	24.286	0.615	0.488	24.383	0.577	0.591	23.154	0.723	0.353	21.680	0.570	0.520
EdgeSRGAN	visual	25.307	0.680	0.228	23.585	0.558	0.348	23.547	0.514	0.386	22.719	0.680	0.257	21.102	0.522	0.374
EdgeSRGAN-tiny	visual	25.523	0.693	0.280	23.976	0.589	0.399	24.163	0.557	0.475	22.874	0.695	0.317	21.477	0.546	0.459

Table 4: Quantitative performance of the proposed method for $\times 8$ upsampling. Current SISR state-of-art method SwinIR [34], and bicubic are reported as references. ↑: higher is better, ↓: lower is better.

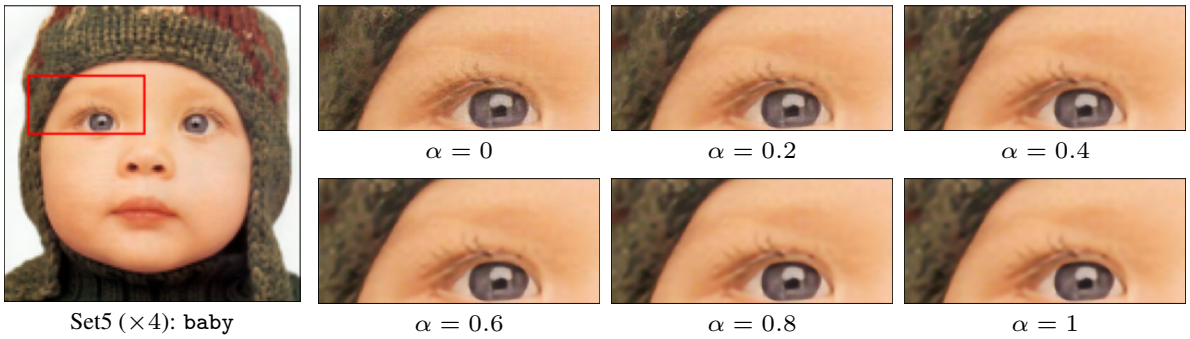

Figure 7: Visual comparison of interpolated EdgeSRGAN for different values of α. Values closer to 1 generate outputs focused on content fidelity, while small values go towards visually pleasing results.

4.3.1 Model Interpolation

We report the results of network interpolation on the benchmark datasets in Fig. 8. We consider α values between 0 and 1 with a step of 0.1, with 0 implying a full visual-oriented model and 1 a full content-oriented one. All results refer to the standard EdgeSRGAN model for $\times 4$ upsampling. This procedure effectively shows how it is possible to choose the desired trade-off between content-oriented and visual-oriented SR simply by changing the interpolation weight α. An increase in the weight value causes an improvement of the content-related metrics PSNR and SSIM and a worsening of the perceptual index LPIPS. This behavior holds for all the test datasets, validating the proposed approach. This procedure can be easily carried out in a real-time application and only requires computing the interpolated weights once. Thus, it does not affect any way the inference speed. For an additional visual evaluation, Fig. 7 reports the outputs obtained for increasing values of α on a random dataset sample.

4.3.2 Model Quantization

To target Edge TPU devices and reach over-real-time inference results, we follow the quantization scheme of Eq. 13 for both weights and activations to obtain a full-integer model. Since quantized models must have a fixed input shape, we generate a full-integer network for each input shape of the testing samples. We use the 100 images from the DIV2K validation set as a representative dataset to calibrate the quantization algorithm. We refer to the int8-quantized standard model as EdgeSRGANi8. As for the tiny model, we optimize the distilled network EdgeSRGANi8-tiny⚗. Results for the visual-oriented optimized models are shown in Tab. 5. Due to the full-integer models' reduced activation and weight, we experience a great increase in inference speed up to over-real-time at the cost of degradation in SR performance. All the proposed quantized models still outperform the bicubic baseline on the perceptual index LPIPS and therefore represent a good option for applications in which really fast inference is needed. A comparison of different models for visual-oriented $\times 4$ upsampling is shown in Fig. 1. We consider LPIPS performance on the Set5 dataset compared to framerate.

		Set5 [7]			Set14 [67]			BSD100 [40]			Manga109 [43]			Urban100 [26]		
Model	**Scale**	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓
EdgeSRGANi8	$\times 4$	27.186	0.721	0.209	24.714	0.475	0.342	23.675	0.484	0.438	25.601	0.712	0.221	22.802	0.580	0.341
EdgeSRGANi8-tiny⚗	$\times 4$	27.330	0.710	0.257	24.807	0.562	0.390	23.837	0.485	0.481	25.299	0.696	0.286	22.580	0.538	0.454
EdgeSRGANi8	$\times 8$	24.433	0.602	0.312	22.846	0.477	0.440	22.609	0.422	0.492	22.227	0.603	0.342	20.525	0.433	0.499
EdgeSRGANi8-tiny	$\times 8$	24.956	0.642	0.333	23.487	0.532	0.461	23.591	0.494	0.544	22.445	0.632	0.386	21.125	0.489	0.548

Table 5: Quantitative performance of the full-integer quantized models for $\times 4$ and $\times 8$ visual-based SR. ↑: higher is better, ↓: lower is better.

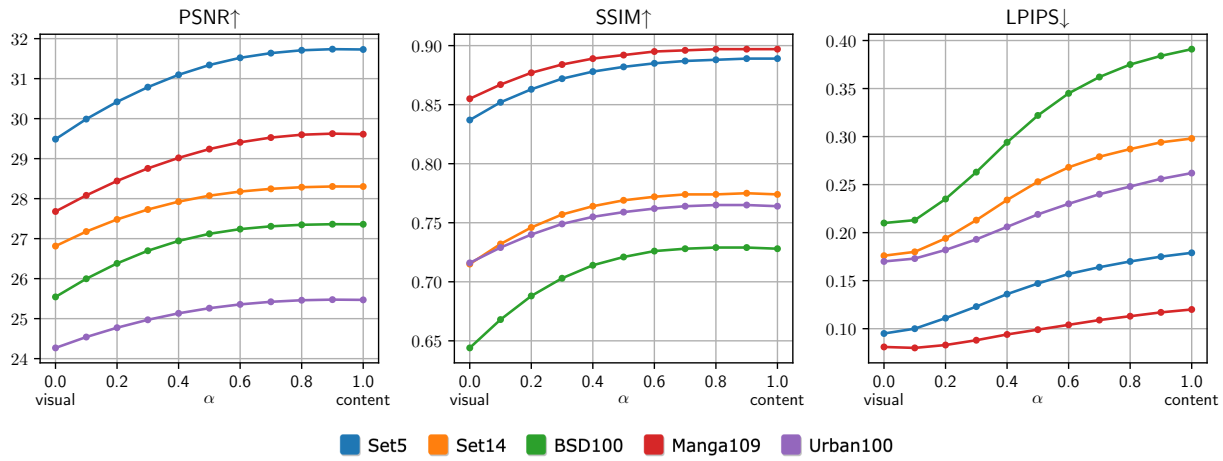

Figure 8: EdgeSRGAN network interpolation results on the benchmark datasets for $\times 4$ upsampling. Changing the network interpolation weight α, it is possible to select the desired trade-off between content-oriented and visual-oriented SR. ↑: higher is better, ↓: lower is better.

4.4 Ablation Study

To further verify the effectiveness of our model for real-time super-resolution, we conduct an ablation study to analyze the effect of our architectural design choices. In particular, we benchmark EdgeSRGAN at four progressive steps, reporting fidelity, perceptual performance, and inference speed. The steps we consider are the following:

1. Reducing the number of residual blocks N;
2. Replacing the Pixel Shuffle upsampling stage with Transpose Convolutions;
3. Removing Batch Normalization;
4. Replacing PReLU activations with ReLU.

The last step corresponds to the final version of EdgeSRGAN. For each step of the model, we use the same training procedure described in 3.2 and measure the inference speed on the CPU at (80x60) and (160x120) input resolutions. All the results are reported in Tab. 6. The experimentation confirms that each compression step gains substantial inference speed by trading minimal perceptual quality. Overall, we observe -3.7% LPIPS perceptual quality and +280% inference speed.

Model	Params	Set5			Set14			BSD100			Manga100			Urban100			Inference Speed (fps)	
		PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	80x60	160x120
SRGAN	1.5M	29,18	0,842	0,094	26,17	0,701	0,172	25,45	0,648	0,206	27,35	0,860	0,076	24,39	0,728	0,158	2.00 ± 0.03	0.48 ± 0.01
$N = 8$	956k	29,38	0,839	0,088	26,55	0,703	0,170	25,08	0,628	0,207	27,49	0,852	0,085	24,21	0,718	0,168	2.47 ± 0.01	0.62 ± 0.01
TransposeConv	663k	28,98	0,829	0,113	26,46	0,706	0,204	25,25	0,641	0,243	26,72	0,833	0,116	23,66	0,689	0,214	9.16 ± 0.31	2.52 ± 0.03
No BatchNorm	661k	29,40	0,838	0,105	26,65	0,709	0,194	25,09	0,630	0,236	27,54	0,851	0,091	24,01	0,707	0,191	9.91 ± 0.16	2.56 ± 0.06
ReLU	661k	29,49	0,837	0,095	26,81	0,715	0,176	25,54	0,644	0,210	27,68	0,855	0,081	24,27	0,716	0,170	10.26 ± 0.11	2.66 ± 0.02

Table 6: Results of the ablation study conducted on EdgeSRGAN for four different steps. The last step corresponds to the final model. Overall, we observe -3.7% LPIPS perceptual quality and +280% inference speed. ↑: higher is better, ↓: lower is better.

4.5 Application: Image Transmission for Mobile Robotics

Our real-time SISR can provide competitive advantages in a wide variety of practical engineering applications. In this section, we target a specific use case of mobile robotics, proposing our EdgeSRGAN system as an efficient deep learning-based solution for real-time image transmission. Indeed, robot remote control in unknown terrains needs a reliable transmission of visual data at a satisfying framerate, preserving robustness even in bandwidth-degraded conditions. This requirement is particularly relevant for high-speed platforms and UAVs. Dangerous or delicate tasks such as tunnel exploration, inspection, or open space missions all require an available visual stream for human supervision, regardless of the autonomy level of the platform. In the last few years, the robotics community has focused on developing globally shared solutions for robot software and architectures and handling data communications between multiple platforms and devices. ROS2 [39] is the standard operative system for robotic platforms. It is a middleware based on a Data Distribution System (DDS) protocol where application nodes communicate with each

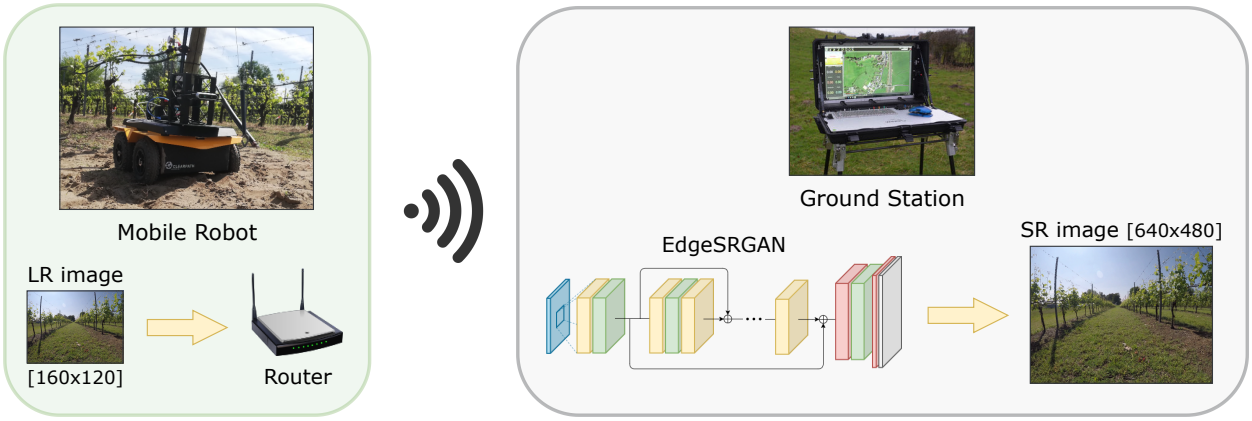

Figure 9: Efficient image transmission system with EdgeSRGAN for mobile robotic applications in outdoor environments.

other through a topic with a publisher/subscriber mechanism. However, despite the most recent attempts to improve the reliability and efficiency of message and data packet communications between different nodes and platforms, heavier data transmission, such as image streaming, is not yet optimized and reliable.

The typical practical setting used for robot teleoperation and exploration in unknown environments is composed of a ground station and a rover connected to the same wireless network. As shown in Fig. 9, we adopted this ground station configuration to test the transmission of images through a ROS2 topic, as should be done in any robotic application to stream what the robot sees or to receive visual data and feed perception and control algorithms for autonomous navigation and mapping. For this experiment, we use both an Intel RealSense D435i camera[5] and a Logitech C920 webcam[6] mounted on a Clearpath Jackal robot[7], together with a Microhard BulletPlus[8] router for image transmission. The available image resolutions with RealSense cameras, the standard RGBD sensors for visual perception in robotics, are (320×240) and (640×480), whereas the framerate typically varies between 15 and 30 fps.

Despite the absence of strong bandwidth limitations, transmission delays, or partial loss of packets, the maximum resolution and framerate allowed by ROS2 communication are extremely low: we find that at 30 fps, the maximum transmissible resolution for RGB is (120×120) with a bandwidth of 20 Mb/s while reducing the framerate to 5 fps the limit is (320×240). This strict trade-off between framerate and resolution hinders the high-speed motion of a robotic platform in a mission, increasing the risk of collision due to reduced scene supervision. Even selecting *best effort* in the Quality of Service (QoS) settings, which manage the reception of packages through topics, the detected performances are always scarce.

Adopting our real-time Super-Resolution system ensures the timely arrival of RGB and depth images via ROS2. Thanks to the fast-inference performance of EdgeSRGAN, we can stream low-resolution images (80×60) at a high framerate (30 fps) and receive a high-resolution output: (320×240) with a x4 image upsampling and (640×480) with a x8 upsampling, showing a clear improvement on standard performance. Our system allows the ground station to access the streaming data through a simple ROS topic. Hence, it provides multiple competitive advantages in robotic teleoperation and autonomous navigation: high-resolution images can be directly exploited by the human operator for remote control. Moreover, they can be used to feed computationally hungry algorithms like sensorimotor agents, visual-odometry, or visual-SLAM, which we may prefer to run on the ground station to save the constrained power resources of the robot and significantly boost the autonomy level of the mission. In Fig. 10, we report a qualitative comparison to highlight the effectiveness of EdgeSRGAN for real-world robotic scenarios. In particular, we consider apple monitoring, navigation in vineyards, drone surveillance for autonomous rovers, and tunnel inspection.

We also test video transmission performance in a more general framework to reproduce all the potential bandwidth conditions. We use the well-known video streaming library GStreamer[9] to transmit video samples changing the available bandwidth. We progressively reduce the bandwidth from 10 Mbps to 10 kbps using the Wondershaper library[10] and measure the framerate at the receiver side. We use 10 seconds of the standard video sample *smtpe* natively provided

[5] https://www.intelrealsense.com/depth-camera-d435i/
[6] https://www.logitech.com/it-it/products/webcams/c920-pro-hd-webcam.960-001055.html
[7] https://clearpathrobotics.com/jackal-small-unmanned-ground-vehicle/
[8] https://www.microhardcorp.com/BulletPlus-NA2.php
[9] https://gstreamer.freedesktop.org/
[10] https://github.com/magnific0/wondershaper

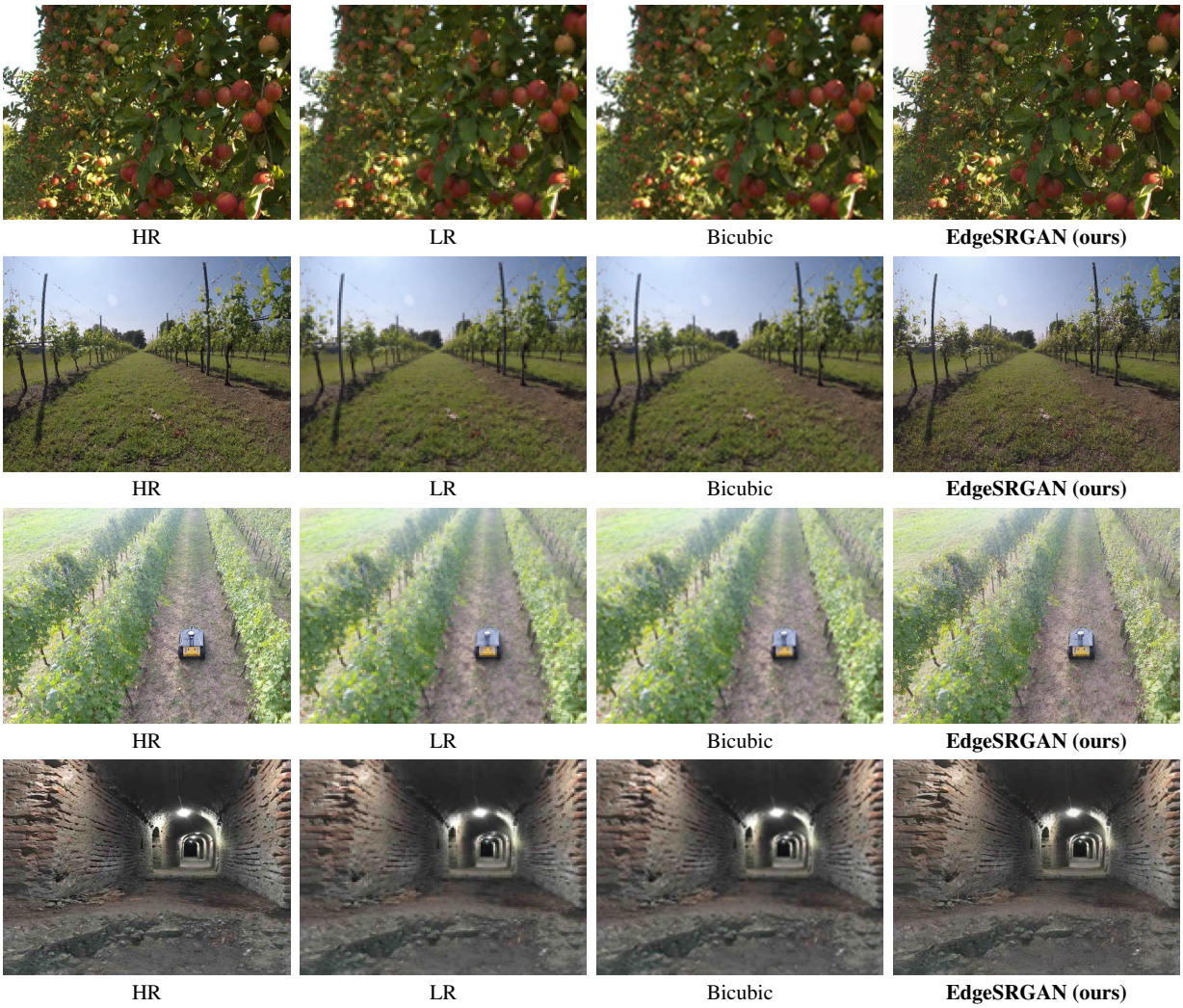

Figure 10: Qualitative demonstration of applying EdgeSRGAN ($\times 4$) on real scenarios (zoom for more detail). From top to bottom: apple monitoring, navigation in vineyards, drone surveillance for autonomous rovers, and tunnel inspection.

by GStreamer *videotestsrc* video source at 30 fps, and we encode it for transmission using MJPEG and H264 video compression standards. The encoding is performed offline to ensure that all the available resources are reserved for transmission only. Indeed, most cameras provide hardware-encoded video sources without requiring software compression. To be consistent with the other experiments, we keep using (640×480) and (320×240) as high resolutions and (160×120) and (80×60) as low resolutions. Each experiment is performed 10 times to check the consistency in results. Fig. 11 presents the average framerate achieved with different bandwidths. Streaming video directly without any middleware, such as ROS2, ensures a higher transmission performance. However, as expected, streaming high-resolution images is impossible in the case of low bandwidth and the framerate quickly drops to very low values, resulting unsuitable for real-time applications. On the other hand, lower resolutions can be streamed with minimal frame drop, even with lower available bandwidths. H264 compression shows the same behavior as MJPEG but shifts to lower bandwidths. Indeed, H264 is more sophisticated and efficient, as it uses temporal frame correlation in addition to spatial compression. In a practical application with a certain bandwidth constraint, a proper combination of a low-resolution video source and an SR model can be selected to meet the desired framerate requirements on the available platform (CPU or Edge TPU). This mechanism can also be dynamically and automatically activated and deactivated depending on the current connectivity to avoid framerate drops and ensure a smooth image transmission.

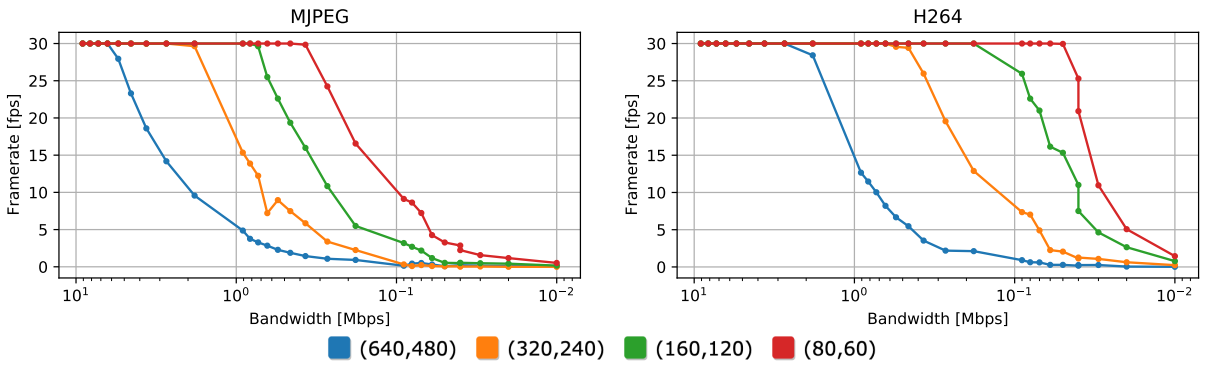

Figure 11: Framerate results vs. bandwidth for video transmission at different input resolutions with MJPEG and H264 compression. Bandwidth is in log scale.

5 Conclusions and Future Works

In this paper, we proposed a novel Edge AI model for SISR exploiting the Generative Adversarial approach. Inspired by popular state-of-the-art solutions, we design EdgeSRGAN, which obtains comparable results, being an order of magnitude smaller in terms of the number of parameters. Our model is 3 times faster than SRGAN, 30 times faster than ESRGAN, and 50 times faster than SwinIR while retaining similar or even better LPIPS performance. To gain additional inference speed, we applied knowledge distillation to EdgeSRGAN and obtained an even smaller network (EdgeSRGAN-tiny) which gains an additional 4x speed with limited performance loss. Moreover, model quantization is used to optimize the model for execution on an Edge TPU. At the same time, network interpolation was implemented to allow potential users to balance the model output between pixel-wise fidelity and perceptual quality. Extensive experimentation on several datasets confirms the effectiveness of our model regarding both performance and latency. Finally, we considered the application of our solution for robot teleoperation, highlighting the validity and robustness of EdgeSRGAN in many practical scenarios in which the transmission bandwidth is limited. Future work may investigate the effect of additional optimization techniques, such as pruning [32] and neural architecture search [48]. Moreover, developing optimized Edge AI versions of more recent architectures like transformers [34] might bring advantages in tackling real-time SISR.

Acknowledgements

This work has been developed with the contribution of the Politecnico di Torino Interdepartmental Center for Service Robotics PIC4SeR[11] and SmartData@Polito[12].

References

[1] D. Aghi, S. Cerrato, V. Mazzia, and M. Chiaberge. Deep semantic segmentation at the edge for autonomous navigation in vineyard rows. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3421–3428. IEEE, 2021.

[2] A. Aguinaldo, P.-Y. Chiang, A. Gain, A. Patil, K. Pearson, and S. Feizi. Compressing gans using knowledge distillation. *arXiv preprint*, 2019.

[3] E. Agustsson and R. Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 126–135, 2017.

[4] J. E. Almanza-Medina, B. Henson, and Y. V. Zakharov. Deep learning architectures for navigation using forward looking sonar images. *IEEE Access*, 9:33880–33896, 2021.

[5] S. Angarano, V. Mazzia, F. Salvetti, G. Fantin, and M. Chiaberge. Robust ultra-wideband range error mitigation with deep learning at the edge. *Engineering Applications of Artificial Intelligence*, 102:104278, 2021.

[6] H. Bae, K. Jang, and Y.-K. An. Deep super resolution crack network (srcnet) for improving computer vision–based automated crack detectability in in situ bridges. *Structural Health Monitoring*, 20(4):1428–1442, 2021.

[11] `https://pic4ser.polito.it`
[12] `https://smartdata.polito.it`

[7] M. Bevilacqua, A. Roumy, C. Guillemot, and M.-L. A. Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In *British Machine Vision Conference (BMVC)*, 2012.

[8] A. Brodie and N. Vasdev. The future of robotic surgery. *The Annals of The Royal College of Surgeons of England*, 100(Supplement 7):4–13, 2018.

[9] J. Cao, Y. Li, K. Zhang, and L. Van Gool. Video super-resolution transformer. *arXiv preprint arXiv:2106.06847*, 2021.

[10] H. Chen, X. He, L. Qing, Y. Wu, C. Ren, R. E. Sheriff, and C. Zhu. Real-world single image super-resolution: A brief review. *Information Fusion*, 79:124–145, 2022.

[11] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, S. Ma, C. Xu, C. Xu, and W. Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12299–12310, 2021.

[12] J. Chen and X. Ran. Deep learning with edge computing: A review. *Proceedings of the IEEE*, 107(8):1655–1674, 2019.

[13] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, and L. Zhang. Second-order attention network for single image super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11065–11074, 2019.

[14] R. de Queiroz Mendes, E. G. Ribeiro, N. dos Santos Rosa, and V. Grassi Jr. On deep learning techniques to boost monocular depth estimation for autonomous navigation. *Robotics and Autonomous Systems*, 136:103701, 2021.

[15] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015.

[16] D. S. Drew. Multi-agent systems for search and rescue applications. *Current Robotics Reports*, 2(2):189–200, 2021.

[17] T. Elmokadem and A. V. Savkin. A method for autonomous collision-free navigation of a quadrotor uav in unknown tunnel-like environments. *Robotica*, 40(4):835–861, 2022.

[18] Y. Fu, W. Chen, H. Wang, H. Li, Y. Lin, and Z. Wang. Autogan-distiller: searching to compress generative adversarial networks. In *Proceedings of the 37th International Conference on Machine Learning*, pages 3292–3303, 2020.

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[20] J. Gou, B. Yu, S. J. Maybank, and D. Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

[21] Z. He, T. Dai, J. Lu, Y. Jiang, and S.-T. Xia. Fakd: Feature-affinity based knowledge distillation for efficient image super-resolution. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 518–522. IEEE, 2020.

[22] H. Hedayati, M. Walker, and D. Szafir. Improving collocated robot teleoperation with augmented reality. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pages 78–86, 2018.

[23] B. Heo, J. Kim, S. Yun, H. Park, N. Kwak, and J. Y. Choi. A comprehensive overhaul of feature distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1921–1930, 2019.

[24] B. Heo, M. Lee, S. Yun, and J. Y. Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3779–3787, 2019.

[25] G. Hinton, O. Vinyals, J. Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[26] J.-B. Huang, A. Singh, and N. Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5197–5206, 2015.

[27] M. Islam, P. Luo, and J. Sattar. Simultaneous enhancement and super-resolution of underwater imagery for improved visual perception. In M. Toussaint, A. Bicchi, and T. Hermans, editors, *Robotics*, Robotics: Science and Systems. MIT Press Journals, 2020.

[28] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2704–2713, 2018.

[29] J. Kim, J. K. Lee, and K. M. Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654, 2016.

[30] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. 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.

[31] D. Li, J. Xu, H. He, and M. Wu. An underwater integrated navigation algorithm to deal with dvl malfunctions based on deep learning. *IEEE Access*, 9:82010–82020, 2021.

[32] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf. Pruning filters for efficient convnets. In *International Conference on Learning Representations*, 2017.

[33] Y. Li, K. Zhang, R. Timofte, L. Van Gool, F. Kong, M. Li, S. Liu, Z. Du, D. Liu, C. Zhou, et al. Ntire 2022 challenge on efficient super-resolution: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1062–1102, 2022.

[34] J. Liang, J. Cao, G. Sun, K. Zhang, L. Van Gool, and R. Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1833–1844, 2021.

[35] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[36] D. Liu, H. Kong, X. Luo, W. Liu, and R. Subramaniam. Bringing ai to edge: From deep learning's perspective. *Neurocomputing*, 2021.

[37] J. Liu, J. Tang, and G. Wu. Residual feature distillation network for lightweight image super-resolution. In *European Conference on Computer Vision*, pages 41–55. Springer, 2020.

[38] I. Lluvia, E. Lazkano, and A. Ansuategi. Active mapping and robot exploration: A survey. *Sensors*, 21(7):2445, 2021.

[39] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall. Robot operating system 2: Design, architecture, and uses in the wild. *Science Robotics*, 7(66):eabm6074, 2022.

[40] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 416–423. IEEE, 2001.

[41] D. E. Martinez, W. Meinhold, J. Oshinski, A.-P. Hu, and J. Ueda. Super resolution for improved positioning of an mri-guided spinal cellular injection robot. *Journal of Medical Robotics Research*, 6(01n02):2140002, 2021.

[42] M. Martini, S. Cerrato, F. Salvetti, S. Angarano, and M. Chiaberge. Position-agnostic autonomous navigation in vineyards with deep reinforcement learning. *arXiv preprint*, 2022.

[43] Y. Matsui, K. Ito, Y. Aramaki, A. Fujimoto, T. Ogawa, T. Yamasaki, and K. Aizawa. Sketch-based manga retrieval using manga109 dataset. *Multimedia Tools and Applications*, 76(20):21811–21838, 2017.

[44] P. N. Michelini, Y. Lu, and X. Jiang. edge-sr: Super-resolution for the masses. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1078–1087, 2022.

[45] B. Niu, W. Wen, W. Ren, X. Zhang, L. Yang, S. Wang, K. Zhang, X. Cao, and H. Shen. Single image super-resolution via a holistic attention network. In *European conference on computer vision*, pages 191–207. Springer, 2020.

[46] A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016.

[47] S. Ooyama, H. Lu, T. Kamiya, and S. Serikawa. Underwater image super-resolution using srcnn. In *International Symposium on Artificial Intelligence and Robotics 2021*, volume 11884, pages 177–182. SPIE, 2021.

[48] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean. Efficient neural architecture search via parameters sharing. In *International conference on machine learning*, pages 4095–4104. PMLR, 2018.

[49] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint*, 2014.

[50] T. Rouček, M. Pecka, P. Čížek, T. Petříček, J. Bayer, V. Šalanskỳ, D. Heřt, M. Petrlík, T. Báča, V. Spurnỳ, et al. Darpa subterranean challenge: Multi-robotic exploration of underground environments. In *International Conference on Modelling and Simulation for Autonomous Systems*, pages 274–290. Springer, 2019.

[51] P. Roy and C. Chowdhury. A survey of machine learning techniques for indoor localization and navigation systems. *Journal of Intelligent & Robotic Systems*, 101(3):1–34, 2021.

[52] M. S. Sajjadi, B. Scholkopf, and M. Hirsch. Enhancenet: Single image super-resolution through automated texture synthesis. In *Proceedings of the IEEE international conference on computer vision*, pages 4491–4500, 2017.

[53] W. Shi, J. Caballero, F. Huszar, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. 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 (CVPR)*, June 2016.

[54] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[55] P. Stotko, S. Krumpen, M. Schwarz, C. Lenz, S. Behnke, R. Klein, and M. Weinmann. A vr system for immersive teleoperation and live exploration with a mobile robot. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3630–3637, 2019.

[56] D. Tardioli, L. Riazuelo, D. Sicignano, C. Rizzo, F. Lera, J. L. Villarroel, and L. Montano. Ground robotics in tunnels: Keys and lessons learned after 10 years of research and experiments. *Journal of Field Robotics*, 36(6):1074–1101, 2019.

[57] R. Timofte, E. Agustsson, L. Van Gool, M.-H. Yang, and L. Zhang. Ntire 2017 challenge on single image super-resolution: Methods and results. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 114–125, 2017.

[58] R. Wang, D. Zhang, Q. Li, X.-Y. Zhou, and B. Lo. Real-time surgical environment enhancement for robot-assisted minimally invasive surgery based on super-resolution. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3434–3440. IEEE, 2021.

[59] X. Wang, L. Xie, C. Dong, and Y. Shan. Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1905–1914, 2021.

[60] X. Wang, K. Yu, C. Dong, and C. C. Loy. Recovering realistic texture in image super-resolution by deep spatial feature transform. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 606–615, 2018.

[61] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. C. Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *The European Conference on Computer Vision Workshops (ECCVW)*, September 2018.

[62] X. Xiao, B. Liu, G. Warnell, and P. Stone. Motion planning and control for mobile robot navigation using machine learning: a survey. *Autonomous Robots*, pages 1–29, 2022.

[63] F. Yin. Inspection robot for submarine pipeline based on machine vision. In *Journal of Physics: Conference Series*, volume 1952, page 022034. IOP Publishing, 2021.

[64] C. Yuan, B. Xiong, X. Li, X. Sang, and Q. Kong. A novel intelligent inspection robot with deep stereo vision for three-dimensional concrete damage detection and quantification. *Structural Health Monitoring*, 21(3):788–802, 2022.

[65] S. Zagoruyko and N. Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint*, 2016.

[66] M. K. Zein, M. Al Aawar, D. Asmar, and I. H. Elhajj. Deep learning and mixed reality to autocomplete teleoperation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4523–4529. IEEE, 2021.

[67] R. Zeyde, M. Elad, and M. Protter. On single image scale-up using sparse-representations. In *International conference on curves and surfaces*, pages 711–730. Springer, 2010.

[68] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[69] Y. Zhang, H. Chen, X. Chen, Y. Deng, C. Xu, and Y. Wang. Data-free knowledge distillation for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7852–7861, 2021.

[70] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. 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.

[71] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2472–2481, 2018.

[72] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss functions for image restoration with neural networks. *IEEE Transactions on computational imaging*, 3(1):47–57, 2016.

[73] F. Zhu, Y. Zhu, V. Lee, X. Liang, and X. Chang. Deep learning for embodied vision navigation: A survey. *arXiv preprint*, 2021.