Self-supervised Perceptual Motion Deblurring using a Conditional Generative Neural Network Guided by Optical Flow

Jaihyun Koh*, **, Sungroh Yoon**, †

***Samsung Display Corporation, Display Electronics Development Team, Yongin, Korea**

****Seoul National University, Department of Electrical and Computer Engineering, Seoul, Korea**

Abstract

The sample-and-hold characteristic of flat-panel displays causes motion blur. Hardware compensation, such as frame-rate doubling, is expensive, and existing methods of software compensation are slow. We propose a display motion deblurring network (DMDnet) which compensates for motion blur using a neural network trained on pairs of images with a synthetic random displacement between them. We assess the compensated images by convolving them with a kernel produced by a perceptual blur estimation algorithm, to simulate what is seen by a viewer. This technique is approximately 87 times faster than a state-of-the-art optimization technique which produces an equivalent level of compensation; and its average PSNR is 29.70 dB, against 26.68 dB for the optimization.

Author Keywords

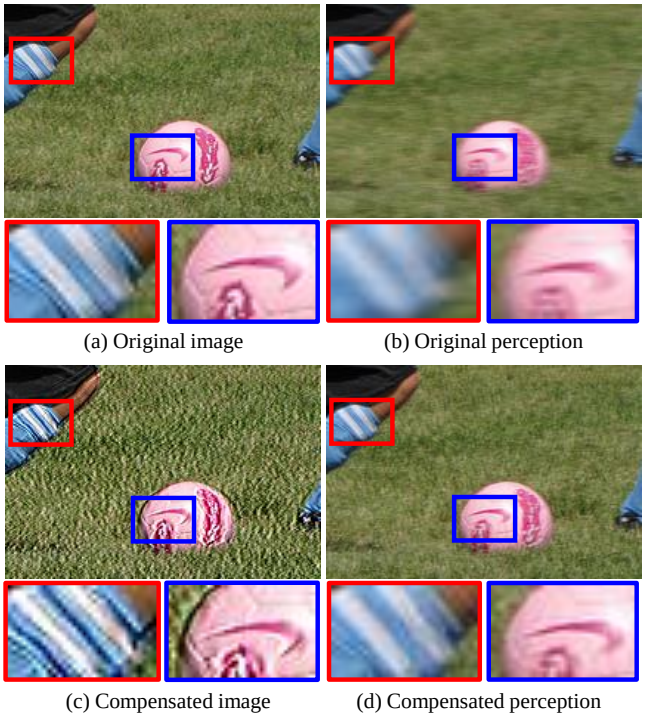
Display; motion blur; sample and hold type; perceptual blur; neural network; deep learning; generative model.

1. Introduction

The sample-and-hold emission characteristic of liquid crystal displays (LCDs) and organic light-emitting diode displays (OLEDs) produces motion blur when there is significant motion between two frames of a video [1]. We call this the *perceptual blur* because it is caused by the properties of the display interacting with the motion pursuit function of the human visual system [2].

Perceptual blur can be reduced by hardware-based approaches, which include frame-rate doubling [3] and backlight scanning [4], but these methods are not widely used because they need expensive hardware such as high-bandwidth circuits, more LEDs, and complicated LED drivers. Software, which pre-processes a sequence of images containing a motion so that it appears perceptually sharp, offers a more cost-effective approach to the problem. The approaches which have been tried include inverse filtering in the spatial frequency domain [5] and the recasting of perceptual deblurring as a regularized optimization problem [6]. These approaches are not fully applicable to arbitrary motion, and the extra processing power required for real-time operation makes them less competitive than hardware approaches.

Deep learning has proved effective in image reconstruction tasks such as super-resolution [7], in which a reliance on optimization has largely been replaced by methods based on discriminative and generative learning. Inspired by recent work on natural image deblurring [8], we present a self-supervised display motion deblurring network (DMDnet) which produces images compensated for perceptual motion blur using motion information. DMDnet consists of a pair of networks, one of which estimates the motion vector, while the other produces the deblurred image from the estimated motion information. This approach has three advantages over previous methods [5, 6]: 1) DMDnet can handle arbitrary motion with a single network, which makes it practical to implement; 2) it can utilize the parallel computation available on a GPU, facilitating real-time operations; 3) results are superior in terms of both quantitative image similarity and perceived fidelity, as shown in Figure 1 (see the caption for details).



(a) Original image (b) Original perception

(c) Compensated image (d) Compensated perception

Figure 1. Perceptual blur compensation using DMDnet. (a) is the input, and (b) is the perception of (a) for a simulated movement of 5 pixels per frame in the positive horizontal (left-to-right) direction. (c) and (d) are the compensated image and its perception, respectively.

2. Related Work

We will briefly review a model [2] of perceptual blur on displays, and a software-based method of reducing perceptual blur [6] to facilitate further reading.

A Model of Perceptual Blur: Pan *et al*. [2] developed a model of perceptual blur in sample-and-hold displays based on the motion pursuit function of the human visual system. In their model, the perceived intensity $I_{\mathrm{p}}(x, y, t)$ at pixel position (x, y) and time-step t can be expressed as follows:

$$I_{\mathrm{p}}(x, y, t) = \int_{-\infty}^{\infty} I_{\mathrm{c}}(x - u\tau, y - v\tau, t - \tau) h_{\mathrm{disp}}(\tau) d\tau, \qquad (1)$$

where $I_{\mathrm{c}}(x, y, t)$ denotes the luminous intensity at time-step t, $h_{\mathrm{disp}}(t)$ is the response as a pixel intensity on the display, and (u, v) is the motion vector. Assuming that the display has a step response during the frame-hold time T, Eq. (1) becomes

$$I_{\mathrm{p}}(x, y, t) = \int_{o}^{T} I_{\mathrm{c}}(x - u\tau, y - v\tau, t - \tau) d\tau, \qquad (2)$$

† Corresponding author.

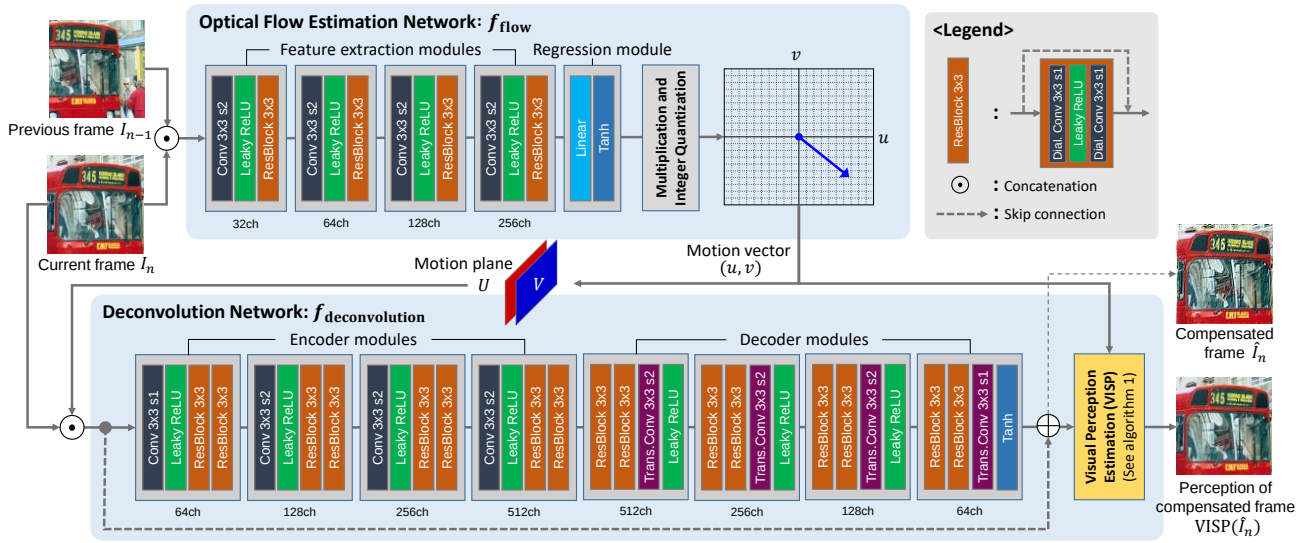


Figure 2. The architecture of DMDnet. The network in the optical flow estimation pathway estimates the motion vector between two consecutive frames. The network in the deconvolution pathway generates the compensated image.

from which the blur perception can be estimated assuming that the motion vector and the display frame hold-time are available. We used this model to predict perceptual blur in the design of DMDnet.

Optimization-based Compensation for Perceptual Blur: Based on the model above [2], a perceptual deblurring algorithm based on optimization was introduced [6]. It estimates the perceptual blur kernel corresponding to a given motion vector and then effectively inverts that kernel by means of an optimization that produces a compensated image. This inversion is highly ill-posed, and therefore both spatial and temporal regularization are required. This method is a significant achievement but it is computationally expensive, especially obtaining and combining blur kernels for arbitrary motion and time-consuming because many iterations are required to refine the output image.

3. Proposed Method

The architecture of DMDnet, shown in Figure 2, consists of an optical flow estimation pathway and a deconvolution pathway. In the following explanation, we will be concerned with a pair of consecutive video frames. The first or previous frame is $I_{n-1} \in \mathbb{R}^{C\times H\times W}$, and the second or current frame is $I_n \in \mathbb{R}^{C\times H\times W}$, where C is the number of color channels, and H and W respectively denote the height and width of the image.

Optical Flow Estimation Network: The optical flow estimation network takes the two images I_{n-1} and I_n as an input and predicts the motion vector (u, v) between them, which can be expressed as follows:

$$(u, v) = f_{\text{flow}}([I_{n-1}, I_n]), \tag{3}$$

where the function f_{flow} represents the effect of the optical flow estimation network, and the square brackets denote a concatenation operation along the axis of a color channel. The optical flow extraction network consists of four feature extraction modules and a regression module. Each feature extraction module has a single 3x3 convolution layer with stride 2, a leaky ReLU with a negative slope of 0.2, and a residual block which contains two dilated 3x3 convolution layers with a dilation factor of 2. Dilated convolution broadens the receptive field of the network, which allows it to cope with more extensive motion. In the regression module, a fully connected (FC) network squeezes the input features and two motion-related scalars are then estimated by a hyperbolic tangent (tanh) activation layer. These two scalars are in the range -1 to $+1$, and we multiply them by 10 to obtain values of u and v which correspond to horizontal and vertical motion of ± 10 pixels. The human eye can follow motion at about 35 degree per second (DPS) [9] which equates to approximately 10 pixels per frame in our experimental setup, in which a 27-inch FHD 60Hz frame-rate display is viewed at a distance of 30 cm [10].

Deconvolution Network: The deconvolution network generates a perceptually compensated image $\hat{I}_n$ from the current image I_n conditioned on the motion vector (u, v). This conditional generative model [11] makes it possible to estimate the compensated images for arbitrary motion. The network creates two motion planes, in the form of two-dimensional matrices U and $V \in \mathbb{R}^{1\times H\times W}$, all the elements of which are respectively filled with the scalars u and v. These matrices are then concatenated with the current image I_n, and deconvolved to find $\hat{I}_n$, as follows:

$$\hat{I}_n = f_{\text{deconvolution}}([I_n, U, V]), \tag{4}$$

where the function $f_{\text{deconvolution}}$ expresses the operation of the deconvolution network. The deconvolution network consists of an encoder and a decoder, each of which has four unit modules. Each module of the encoder has a single 3x3 convolution layer with a leaky ReLU with a negative slope of 0.2, and two residual blocks. Each unit module in the decoder has a single 3x3 transposed convolution layer for feature reconstruction.

We train DMDnet in a self-supervised manner employing a perceptual blur prediction algorithm instead of using a large scale labeled dataset. To evaluate the extent to which the compensated image $\hat{I}_n$ is perceptually corrected, we introduce a visual perception estimation block (VISP) into the deconvolution network. VISP is based on a perceptual blur model [2] and a blur estimation algorithm [6], and can deal with motion in any direction. VISP also has the characteristics, such as differentiability, which are required in training a neural network.

Algorithm 1. Visual Perception Estimation (VISP)

Input: Motion vector $(\boldsymbol{u}, \boldsymbol{v})$ and image $\hat{\boldsymbol{I}}_{\boldsymbol{n}}$

Step 1. Blur kernel generation

$abs_u \leftarrow |u| + 1, abs_v \leftarrow |v| + 1, K \leftarrow abs_u \times abs_v$
$Su \leftarrow \{0, 1 \times abs_v, 2 \times abs_v, \cdots, K\}$
$Sv \leftarrow \{0, 1 \times abs_u, 2 \times abs_u, \cdots, K\}$
$S \leftarrow Su \cup Sv,\ ku \leftarrow 0,\ kv \leftarrow 0,\ i \leftarrow 0,\ kernel \leftarrow \mathbb{0} \in \mathbb{R}^{21\times 21}$
while $i <$ **length**(S) **do**
  $weight \leftarrow S(i+1) - S(i)$
  if $S(i) \in Su$ **and** $S(i) \in Sv$
    $ku \leftarrow ku + 1,\ kv \leftarrow kv + 1$
  else if $S(i) \in Su$ **and** $S(i) \notin Sv$
    $ku \leftarrow ku + 1$
  else if $S(i) \notin Su$ **and** $S(i) \in Sv$
    $kv \leftarrow kv + 1$
  $kernel[10 + (kv * \textbf{sign}(v)), 10 + (ku * \textbf{sign}(u))] \leftarrow weight$
  $i \leftarrow i + 1$
end while

Step 2. Convolution

$B \leftarrow$ **Convolution_2d**$(\hat{I}_n, kernel/K)$

Output: Simulated perceptual blur image $\boldsymbol{B}$

Algorithm 1 summarizes VISP, which generates a perceptual blur kernel from a motion vector, then estimates the perceived image by convolving the input image with the kernel. The VISP is implemented, and accelerated by CUDA, therefore it is fast and efficient to train the network.

Loss Function: To train the optical flow estimation network, we used the mean squared error (MSE) loss $\mathcal{L}_{\text{motion}}$, which expresses the difference between the estimated motion vector $\hat{\boldsymbol{m}} = (\hat{u}, \hat{v})$ and the actual motion vector $\boldsymbol{m} = (u, v)$:

$$\mathcal{L}_{\text{motion}} = \frac{1}{2}||\hat{\boldsymbol{m}} - \boldsymbol{m}||_2^2 \,. \tag{5}$$

The deconvolution network uses two loss functions. The first is an MSE loss $\mathcal{L}_{\text{content}}$, which expresses the difference between the perception of the compensated image, as estimated by VISP, and the original sharp image I_n:

$$\mathcal{L}_{\text{content}} = \frac{1}{CHW}||f_{\text{VISP}}(\hat{I}_n) - I_n||_F^2, \tag{6}$$

where the function f_{VISP} expresses the operation of VISP block, C, H and W are the dimensions of the image, and the subscript F denotes the Frobenius norm. However, an MSE loss alone often produces visual artifacts, in particular ringing, in reconstructed images. We therefore augment $\mathcal{L}_{\text{content}}$ with a perceptual loss $\mathcal{L}_{\text{perceptual}}$ defined as the L2 distance between two feature maps constructed by a trained VGG network [12]. The perceptual loss can be expressed as follows:

$$\mathcal{L}_{\text{perceptual}} = \frac{1}{C_j H_j W_j}||\Phi_j(\hat{I}_n) - \Phi_j(I_n)||_F^2, \tag{7}$$

where $\Phi_j(\cdot)$ denotes feature maps output by the j^{th} convolution layer of a VGG19 network [13] trained on the ImageNet dataset [14], and C_j, H_j and W_j are the dimensions of the feature maps. We use the 14^{th} layer, as is the usual case. The perceptual loss expresses the similarity of edge features between the reconstructed image and the original image. This suppresses the ringing caused by noise at a high spatial frequency. The total loss function $\mathcal{L}_{\text{total}}$ for the deconvolution network is defined as follows:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{content}} + \lambda\mathcal{L}_{\text{perceptual}}, \tag{8}$$

and we set the hyperparameter λ to 0.0001 when we train our DMDnetwork.

Table 1. Average scores on the COCO test set.

Metric	Original	Compensated
PSNR (dB)	20.94	**27.12**
SSIM	0.5912	**0.8389**

Table 2. Comparison of DMDnet with an optimization method [6]. The results are averaged over 24 test images.

Motion (pixels per frame)	Original		Optimization method [6]			DMDnet		
	PSNR	SSIM	PSNR	SSIM	Time	PSNR	SSIM	Time
3 PPF	24.13	0.7717	29.31	0.8944	5.04	**33.00**	**0.9900**	**0.072**
5 PPF	22.39	0.6667	27.40	0.8221	5.65	**30.14**	**0.9512**	**0.073**
7 PPF	21.38	0.6034	25.50	0.7512	6.49	**28.43**	**0.9080**	**0.067**
9 PPF	20.71	0.5620	24.49	0.7137	7.35	**27.24**	**0.8734**	**0.071**

4. Experiments

We train DMDnet on the COCO dataset [15], which contains 118k training images and 40k test images.

Data Preparation: We generate pairs of images, each pair consisting of a previous and a current frame, together with the corresponding motion vector. We start by cropping an image from the dataset to a square with opposite corners at (x, y) and $(x + 256, y + 256)$. This becomes the previous frame I_{n-1}. The current frame I_n is another square image with its corners at $(x - u, y - v)$ and $(x - u + 256, y - v + 256)$. The values of u and v, which are the horizontal and vertical displacements between the previous and the current image that form the motion vector, are chosen in the range $[-10, 10]$. The offsets x and y are random numbers between 10 and $(\varepsilon - 10)$, where ε is the extent of the source image in the corresponding direction. Images which have either edge shorter than 276 pixels are discarded.

Training: We train our two networks on the COCO dataset. We use the ADAM optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and set the initial learning rate to 10^{-4}, making it decay linearly to zero after 4,800k iterations. During training, random transformations are applied to the image pairs, including horizontal and vertical flipping. We also add zero-mean Gaussian random noise from $\mathcal{N}(0, 0.01)$ to each image in the pair, with the aim of increasing the robustness of the trained network. All our experiments are performed on an NVIDIA TITAN X GPU.

Results: We evaluated the optical flow estimation network on the 40k COCO test dataset using the MSE metric. The average error rate is 0.035, which means that the predicted motion has on average errors of 0.35 of a pixel. An error of less than 0.5 pixel is not significant because of rounding operations in the network.

We evaluated DMDnet, by assessing the difference between each current frame and its perceived image, as estimated by VISP from the deconvolved image, in terms of peak signal-to-noise ratio (PSNR) and structural similarity (SSIM). Table 1 gives the average of these metrics for the current images and the images output by VISP. We also conducted experiments to compare DMDnet with an optimization method [6]. We applied each method to 24 test images. The results, including inference timings, are shown in Table 2. DMDnet outperforms the optimization method in terms of both visual quality and inference time. The average PSNR and SSIM are respectively improved by 3.02 dB and 0.1353, and the inference time is approximately 87 times faster than optimization

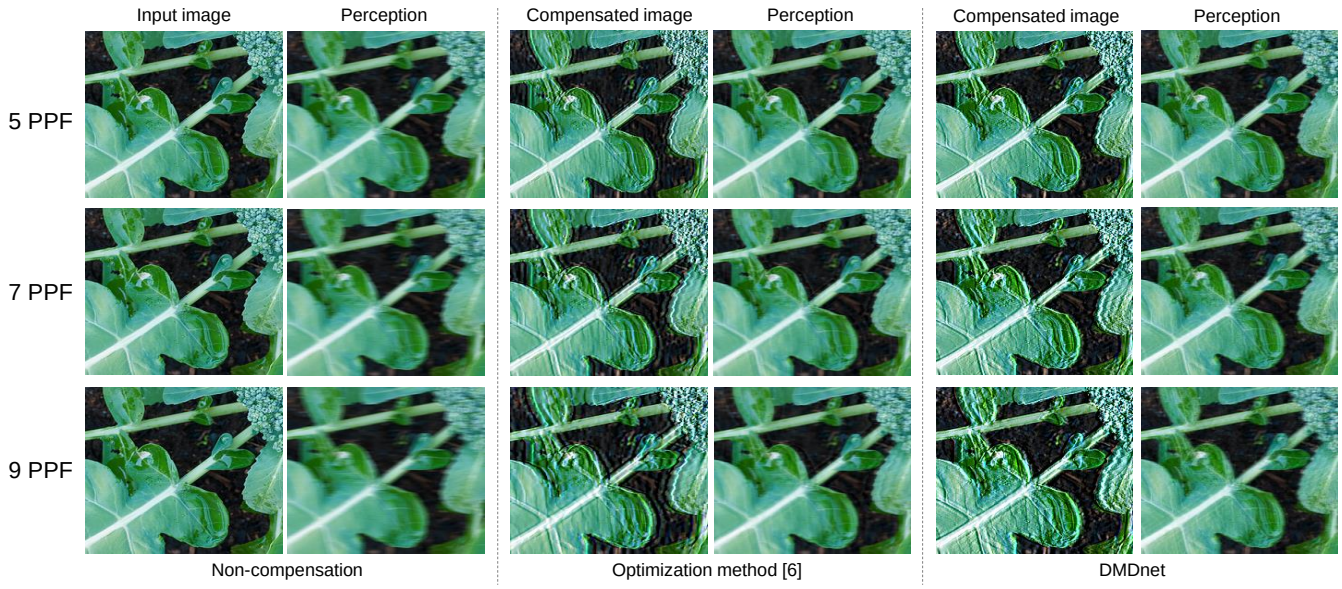


Figure 3. Examples of motion blur corrected by DMDnet and by an optimization method [6].

method. Note that the times presented for the optimization method exclude generation of the blur kernel which provides its estimate of the human perception. This would increase times significantly, widening the performance gap even larger. DMDnet does not require the equivalent perception estimation step (VISP) during inference because the network can implicitly learn the blur kernel from the large scale training dataset during training.

A qualitative comparison between DMDnet and the optimization is presented in Figure 3. The optimization method [6] appears to have limited ability to cope with large motion. This can be attributed to the need for more iterations as motion become larger. By contrast, the performance of DMDnet is not much affected by the extent of the motion because its response to small and large motion is learned in the same way.

5. Conclusion and Future Work

DMDnet is trained on linear blurs which affect the whole image. However, real videos often depict objects moving against a stationary background. An acceptable approach to this situation is to divide the image into small patches and then to perform perceptual deblurring separately on each patch. Temporal and spatial smoothing is then required to avoid the appearance of artifacts at the boundaries between patches.

Another approach to more complicated blurs would be the compilation of an image dataset containing more complicated motion, with estimates of the corresponding perceptual blurs. Such a dataset could be synthesized quite straightforwardly by combining a background and a foreground object, each with its own motion vector. If we were able to predict the perceptual blur for two different motion of this sort, we could apply DMDnet to more complicated images.

Both of the above methods will likely be required to produce an approach to motion blur compensation based on deep learning suitable for commercialization.

LCD and OLED displays have been inferior to CRTs in terms of motion blur. We have proposed an economical of ameliorating motion blur through image processing. We have shown that deep learning techniques can outperform optimization methods in this task. Our approach is orthogonal to any other hardware-based methods, and we would expect that it could be combined with them.

6. References

1. Kurita T. Moving picture quality improvement for hold-type AM-LCDs. SID Symposium. 2001.
2. Pan H, Feng XF, Daly S. LCD motion blur modeling and analysis. IEEE ICIP. 2005.
3. Itoh G, Mishima N. Novel frame interpolation method for high image quality LCDs. Journal of Information Display, 2004.
4. Fisekovic N. Improved motion picture quality of AMLCDs using scanning backlite. Asia Display, 2001.
5. Klompenhouwer M *et al.* LCD motion blur reduction with motion compensated inverse filtering. SID Symposium. 2004.
6. Chan S, Nguyen T. LCD motion blur: modeling, analysis, and algorithm. IEEE TIP. 2011.
7. Kim J *et al.* Accurate image super-resolution using very deep convolutional networks. IEEE CVPR. 2016.
8. Nah S *et al.* Deep multi-scale convolutional neural network for dynamic scene deblurring. IEEE CVPR. 2017.
9. Westerink. J, Teunissen. K. Perceived sharpness in complex moving images. Displays. 1995.
10. Winkler S. Digital video quality: vision models and metrics. John Wiley & Sons. 2005.
11. Mirza M, Osindero S. Conditional generative adversarial nets. arXiv. 2014.
12. Johnson J, Alahi A, Fei-Fei L. Perceptual losses for real-time style transfer and super-resolution. ECCV. 2016.
13. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv. 2014.
14. Deng J *et al.* Imagenet: A large-scale hierarchical image database. IEEE CVPR. 2009.
15. Lin T *et al.* Microsoft COCO: Common objects in context. ECCV. 2014.