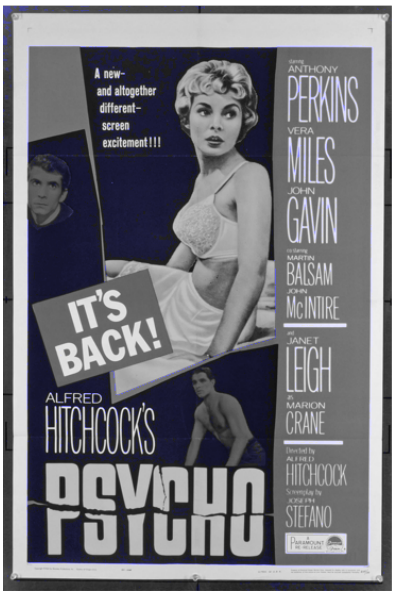
Modernize Old Movies through Colorization and Animation

Team: DeeperFour

Motivation

It is a pity to watch old classic movies are being forgotten by people due to limited shooting technology and the relatively boring black-and-white presentations. Therefore, we decided to make those films more enjoyable by filling them with vivid colors by virtue of existing colorization networks, and creating an animated cartoon version thanks to style-transfer techniques. However, existing colorization and animation networks work on independent images. In order to make the final output video smooth and well displayed, we added another image transfer network with a ConvLSTM layer to increase the consistency of the video through short-term and long-term temporal loss between consecutive frames. Training the animation network with different artist's style, our work can render any black-and-white film with realistic and colorization and cartoon style in specific art style.

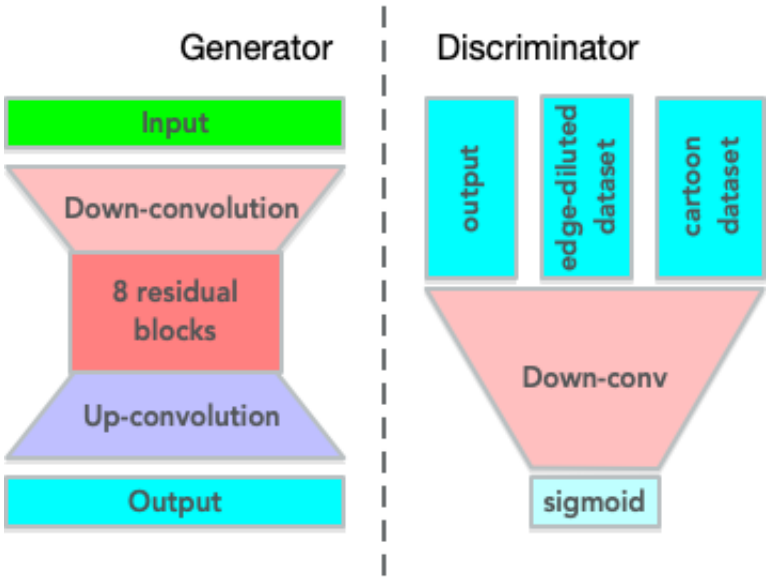


Previous Works

DeOldify is a deep learning based model, which presents an approach for colorization, espeically for photos taken with old/bad equipment. It is inspired and a combination of Self-Attention Generative Adversarial Network [1], Progressive Growing of GANs [2], Two Time-Scale Update Rule [3].

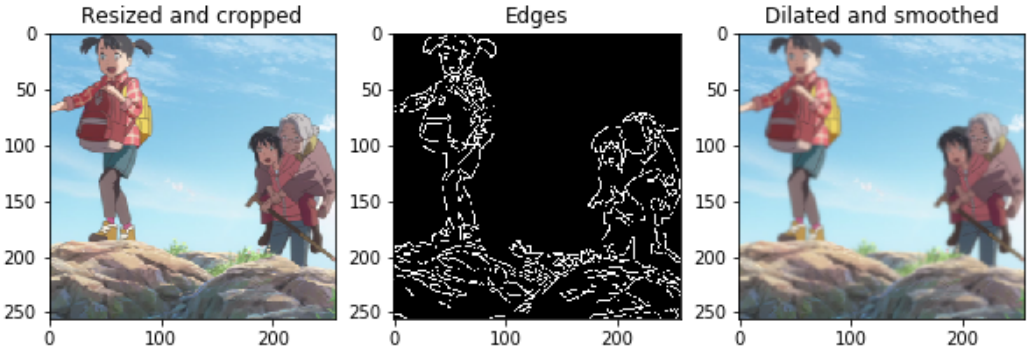
ConvLSTM, based on a deep recurrent network, proposes a solution to enforcing temporal consistency in a video after image processing algorithms applied. In this work, they minimize the temporal loss(short-term loss and long-term loss) for temporal stability and a perceptual loss for perceptual similarity.

CartoonGAN [4] works on transforming real-world photos into cartoon style images with two novel losses: semantic content loss for style variation and the edge-promoting adversarial loss for preserving clear edges.



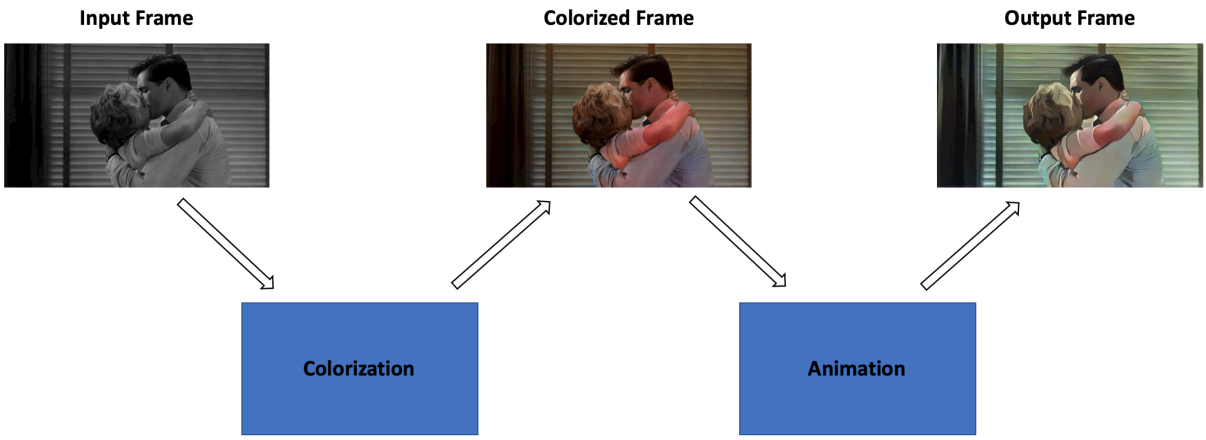
Datasets

We gathered four Makoto Shinkai's works: *The Place Promised in Our Early Days*(雲のむこう、約束の場所), *Five Centimeters per Second*(秒速5センチメートル), *The Garden of Words*(言の葉の庭), and *Your Name*(君の名は). First, we clipped the video to frames on a pace of 1 frame per 3 seconds, and then discarded black frames and frames from crew lists. Second, we resized and cropped all frames into size 256×256. And in order to make the generator to generate clearer contours, we augmented the data by dilating and smoothing the edges(regions that were found by Canny edge detector [5]).



Architectural Design

Initial Idea



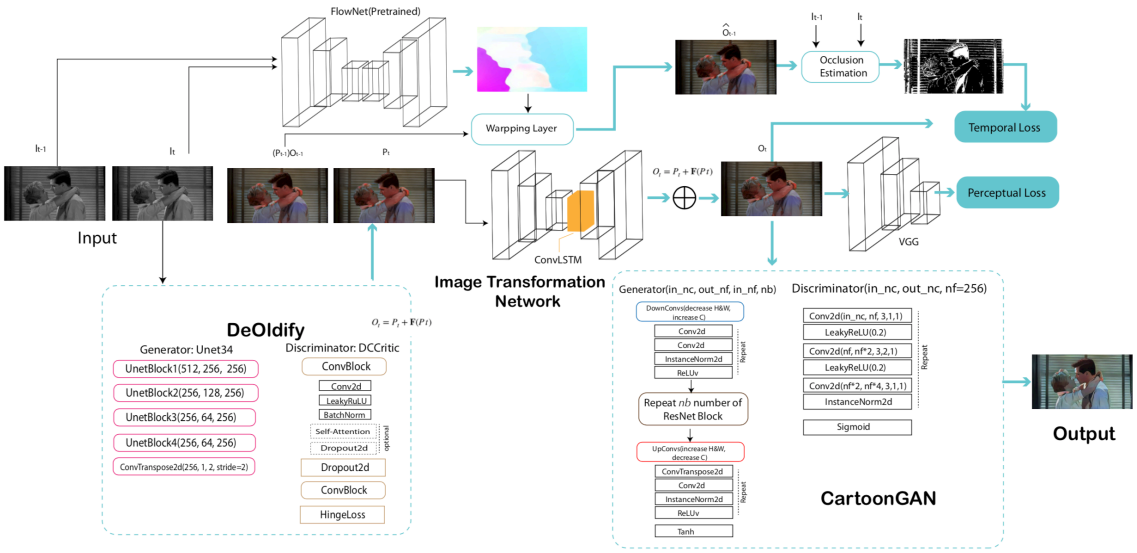
Initially, we implemented a straight-forward pipeline and let colorization and animation work as building blocks inside of it; we used original video frames as inputs, and let them pass through colorization and animation blocks and then stream together the output frames back to a video.

One of the problems for this approach is that the output video was not stable enough because of significant color gaps and strobes. Therefore we decided to use the initial approach as baseline and moved on to find ways to stabilize the video output.

Final Solution: Pipeline with Combined Models and Consistency Loss Functions

Description To stabilize the video, we need to make the color and the tone between consecutive frames and of a tandem of frames consistent with each other. We decided to add consistency loss during the training process. The inspiration came from Wei-Sheng Lai's paper *Learning Blind Video Temporal Consistency* [6]. We used a combined loss function of perception loss, short-term and long term temporal loss. VGG perception loss was used to guarantee the quality of the colorized frame, temporal loss was designed to increase the consistency of adjacent frames. Detailed model architecture and loss functions are shown below.

Model Architecture



The above picture shows the model architecture and here are some details in every iteration:

- Sequential black-and-white frames, denoted as $I_1, \ldots, I_{t-1}, I_t$ and frames colorized by pre-trained DeOldify model, denoted as $P_1, \ldots, P_{t-1}, P_t$ are taken as inputs.
- Colorized frames: $P_1, \ldots, P_{t-1}, P_t$, are passed through Image Transformation Network, which as a ConvLSTM inside. The network will a function $O_t = P_t + F(P(t))$ to generate the colorized, stabilized output frame, O_t.
- Meanwhile, $I_1, \ldots, I_{t-1}, I_t$ are passed through a pre-trained FlowNet network, which will generate an optimal flow.
- The optimal flow will be taken, together with colorized frames: $P_1, \ldots, P_{t-1}, P_t$, as inputs to the warping layer, which will generate frames $\hat{O}_1, \ldots, \hat{O}_{t-1}, \hat{O}_t$, which will work as previous inputs in calculating LSTM loss.
- Parameter M, is computed, based on occlusion estimation, which takes black-and-white frames, $I_1, \ldots, I_{t-1}, I_t$ as inputs. This parameter controls the altitude of the consistency loss based on how different two sequential frames are(If two frames are really different, we will have a really small M, to guarantee long term and short term loss stay at a reasonable scale).
- Calculate temporal loss based on short term loss function and long term loss function.
- Calculate VGG perceptual loss by passing through a VGG network.
- Calculate overall loss, and try to minimize this loss at every iterations.

Loss Function

$$\mathcal{L}_p = \sum_{t=2}^{T} \sum_{i=1}^{N} \sum_{l} \|\phi_l(O_t^{(i)}) - \phi_l(P_t^{(i)})\|_1$$

$$\mathcal{L}_{st} = \sum_{t=2}^{T} \sum_{i=1}^{N} M_{t \Rightarrow t-1}^{(i)} \|O_t^{(i)} - \hat{O}_{t-1}^{(i)}\|_1$$

$$\mathcal{L}_{lt} = \sum_{t=2}^{T} \sum_{i=1}^{N} M_{t \Rightarrow t-1}^{(i)} \|O_t^{(i)} - \hat{O}_1^{(i)}\|_1$$

$$\mathcal{L} = \lambda_p \mathcal{L}_p + \lambda_{st} \mathcal{L}_{st} + \lambda_{lt} \mathcal{L}_{lt}$$

Notations: l denotes layers, and ϕ_l is the activation of the layer l. T is the number of frames in the batch, N is the total number of pixels.

VGG loss was used to make sure that the quality of the image corresponds well to human perception.

Short term loss minimize the pixel-by-pixel L-1 loss between two consecutive inputs, making them consistent. The input here are the previous frame that was warped by the pre-trained FlowNet, which will guide the image transformation network learn the transformation of current frame being processed.

Long term loss sums up pair-wise distance of a sequence of frames in the current batch to the first image of the current batch, the distance is the same pixel-by-pixel L-1 loss as used in the short term loss. Memory and speed constrained the length of frames coherence we consider, but we can always decrease batch size to conform with the limits.

Overall loss is a weighted sum of the above losses, the weight hyper-parameters from the paper[6] was used here, because training is too time consuming to tune them.
$\lambda_p = 10, \lambda_{st} = 100, \lambda_{lt} = 100.$

Training of the stabilization

For stabilization, we selected 60 videos with 4,209 frames in total named DAVIS-gray[7], which contains a lot of moving objects. And also additional 80 videos with 21,526 frames named VIDEVO-gray videos[8] as our training set. All frames were processed to be 480x480.

Limited by time, we only managed to finish training for 30 epoch, 1000 batch per epoch, which took around 4 days. Considering the memory limits we had, the long-term loss was handled on a 10 image period.

Evaluation

It is difficult to quantify the quality of synthesized images, especially cartoons and videos. Here, we used the warping metric[6] to measure the temporal stability on the output videos. The metric is based on the flow warping error between two frames,

$$E_{warp(V_t,V_{t+1})} = \frac{1}{\sum_{i=1}^{N} M_t^{(i)}} \sum_{i=1}^{N} M_t^{(i)} ||V_t^{(i)} - \hat{V}_{t+1}^{(i)}||_2^2$$

where $\hat{V}_{t+1}$ is the warped frame V_{t+1} and $M_t \in 0, 1$ is a non-occlusion mask indicating non-occluded regions, which is based the occlusion detection method in[9]. The average warping error over the entire sequence is calculated as:

$$E_{warp(V)} = \frac{1}{T-1} \sum_{t=1}^{T-1} E_{warp(V_t,V_{t+1})}$$

Quantitative evaluation on temporal warping error

Videos	Our model	Baseline(original processed video)
psycho_kettle	0.000293	0.000688
psycho_kiss	0.000271	0.000782
BeachByDrone	0.000188	0.000375
Cat	0.000409	0.000996
CloudsTimelapse	0.000181	0.000293
TajMahal	0.000138	0.000260
WestminsterBridge	0.000224	0.000450
SanFrancisco	0.000561	0.000739
MotherAndSon	0.000295	0.001414
WomanWorking	0.000099	0.000337
Average	**0.000266**	**0.000633**

Results

Psycho videos

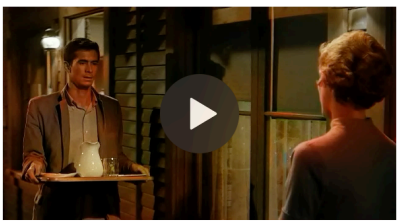
Psycho Section I

Baseline Video

Our Stable Video

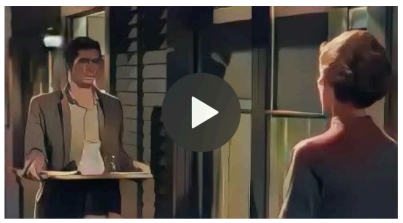
Cartoon Style

Psycho Section II



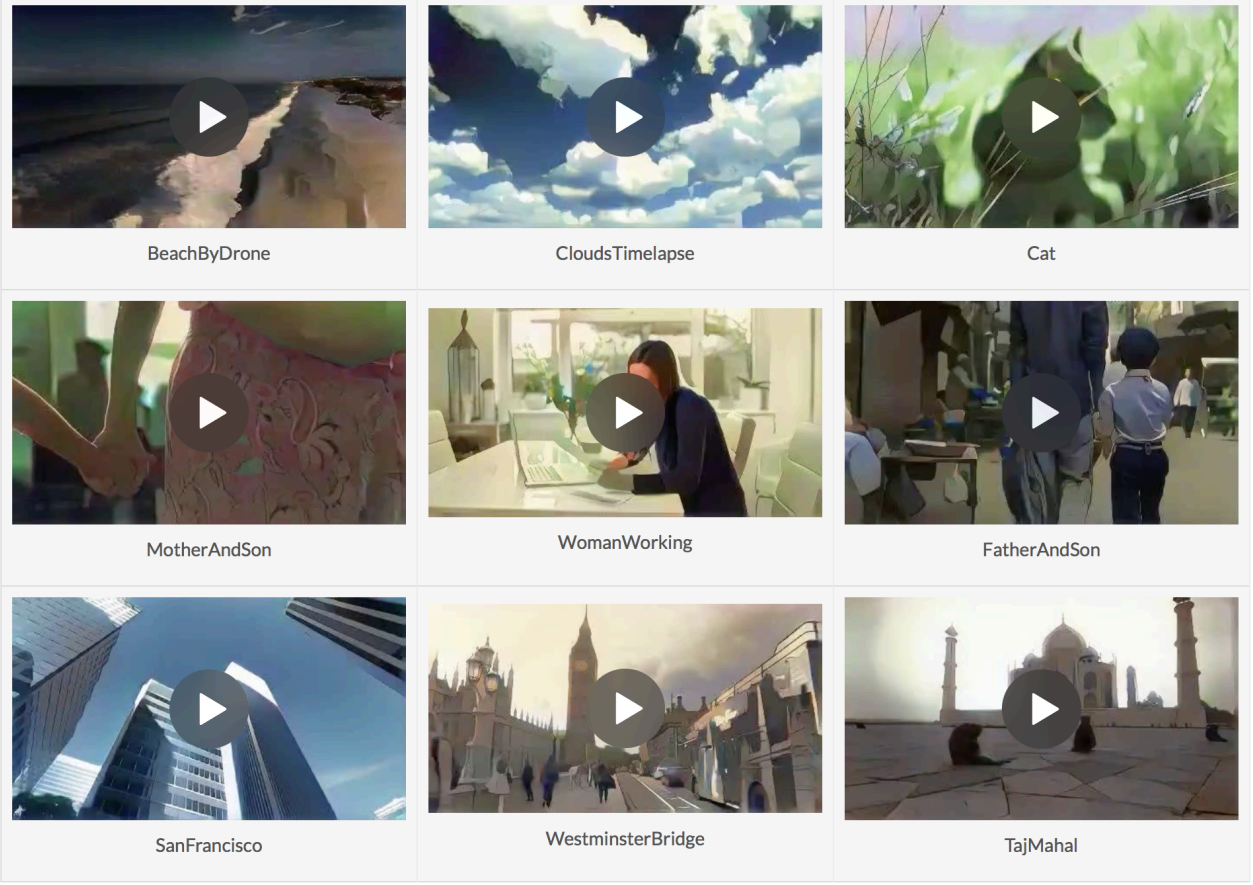
Baseline Video

Our Stable Video



Cartoon Style

More results



Experiments and ideas we tried

Choosing between Colorization Models

We also tried another colorization model, for example an implementation of GAN model based on [10]. And we eventually chose DeOldify as our model because it renders more vivid colors, thus will be better for animation in next phase.

Results of Using Johnson's Network

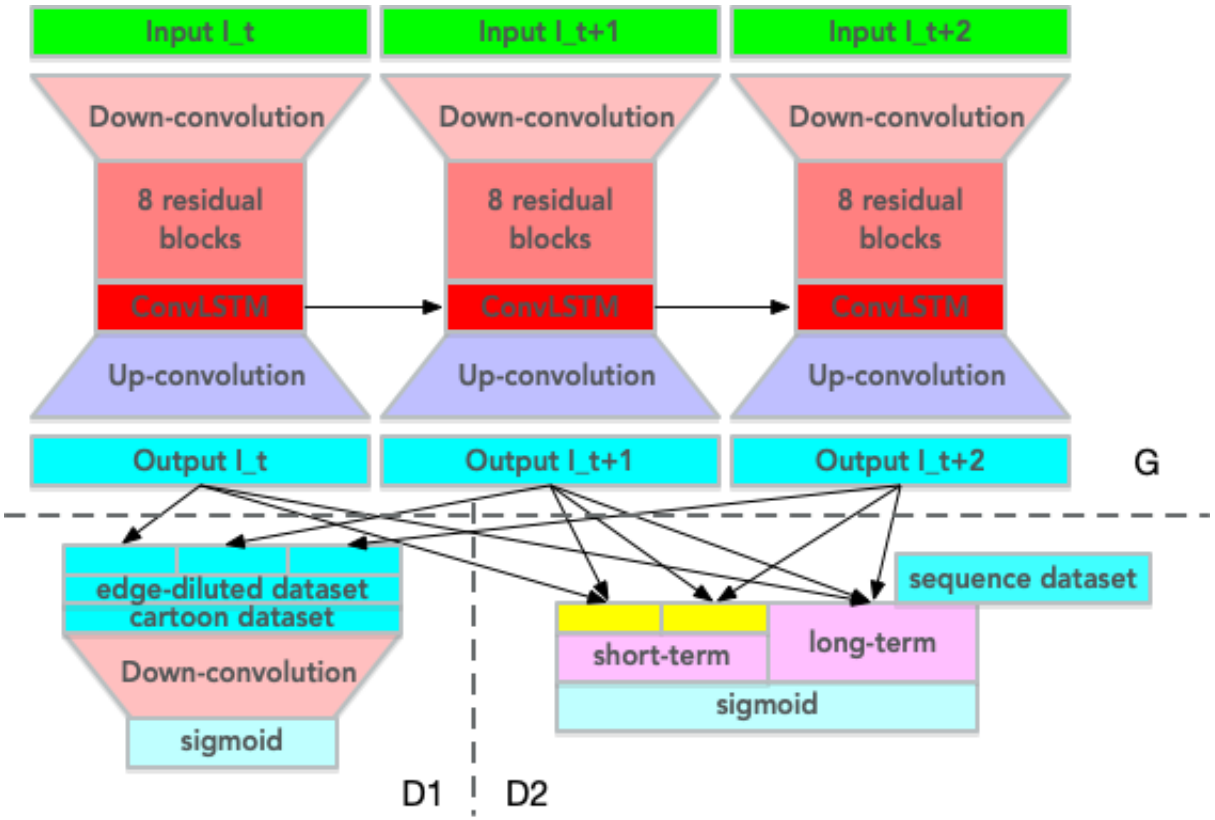
Results of Using DeOldify

Tune DeOldify

Previously, we tried to do some transfer learning on DeOldify with video frames to improve its capability to work on video colorization. But the training failed because of “cuda is out of memory” on Google Cloud Platform. Therefore, we stayed with the pre-trained model.

GAN for Stabilization

As discussed with Professor Lim, we thought about adding a stabilization structure to CartoonGAN and a stabilization discriminator. We enriched our original dataset with its consecutive frames (5 frames, 10 frames, and 20 frames), we calculated pixal-based squared distance between every two consecutive frames to make sure there are no scene-cut inside our dataset(Afterwards, we realized that we shouldn’t do that, we manually made our dataset more biased). We utilized a dual-discriminator structure [11], one is the same as CartoonGAN, to determine whether the frame-by-frame input is a real cartoon and the otherone is used to determine whether the sequence input is a consecutive cartoon using short-term and long-term loss [6] [12]. However, this GAN is incredible harder to train, we downsize our batch size from 6 to 3 to 1, and reduce consecutive frames from 5 to 3, then we managed to get rid of Cuda’s out of memory error. And training speed is very slow(couldn't finish even one epoch in a day), and still had many hyper-parameter to tune. It seemed unworthy to finish it to check if it works and tune everything, therefore we dropped this idea.



References

1. Han Zhang et al. “Self-Attention Generative Adversarial Networks”. In: arXiv e-prints, arXiv:1805.08318 (May 2018), arXiv:1805.08318. arXiv: 1805.08318 [stat.ML].↩

2. Karras, Tero, Timo Aila, Samuli Laine, and Jaakko Lehtinen. "Progressive growing of gans for improved quality, stability, and variation." *arXiv preprint arXiv:1710.10196* (2017).↩

3. Heusel, Martin, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. "Gans trained by a two time-scale update rule converge to a nash equilibrium." *arXiv preprint arXiv:1706.08500* 12, no. 1 (2017).↩

4. Yang Chen, Yu-Kun Lai, and Yong-Jin Liu. “CartoonGAN: Generative Adversarial Networks for Photo Cartoonization”. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (2018), pp. 9465– 9474.↩

5. Canny, John. "A computational approach to edge detection." In *Readings in computer vision*, pp. 184-203. Morgan Kaufmann, 1987.↩

6. Wei-Sheng Lai et al. “Learning Blind Video Temporal Consistency”. In: CoRR abs/1808.00449 (2018). arXiv: 1808.00449. url: http://arxiv.org/abs/1808. 00449.↩↩↩↩

7. https://davischallenge.org/index.html↩

8. https://www.videvo.net↩

9. Xingjian, S., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., Woo, W.C.: Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In: NIPS (2015)↩

10. Johnson, Justin, Alexandre Alahi, and Li Fei-Fei. "Perceptual losses for real-time style transfer and super-resolution." In *European conference on computer vision*, pp. 694-711. Springer, Cham, 2016.↩

11. Nguyen, Tu, Trung Le, Hung Vu, and Dinh Phung. "Dual discriminator generative adversarial nets." In *Advances in Neural Information Processing Systems*, pp. 2670-2680. 2017.↩

12. Ruder, Manuel, Alexey Dosovitskiy, and Thomas Brox. "Artistic style transfer for videos and spherical images." *International Journal of Computer Vision* 126, no. 11 (2018): 1199-1219.↩