StyleArm: a style-transferring robot arm.

Vernon Stanley Albayeros Duarte

Abstract

As the paradigm of human-computer interaction shifts to increasingly intelligent systems that require no direct user inputs to provide services, the way we interact with our machines is still primarily ”active interactions”, as the computer requires some sort of direct user input to provide it’s content.

In this master’s thesis, we create a physical proof of concept that implements computer vision algorithms to aid in interaction, in the form of a self-built robotic arm with a camera that is able to interface with a Raspberry Pi, a small computer. The proof of concept utilizes a server-client connection with a higher powered machine capable of relaying images treated by a fast Style Transfer neural network in near real-time, switching the context of the style transfer depending on the facial expression of the user. The goal of this robot is to provide the user with a way to interact with computer vision processes they might find useful, like Style Transfer or image stitching for story-telling and publication on social media accounts. Initially, this project meant to optimize a style transfer GAN for use on a Raspberry Pi and provide a completely autonomous project, but we were unable to produce results that could be used in real time.

Index Terms

Computer Vision, GAN, Generative Adversarial Network, Style Transfer, Deep Learning, Neural Networks, Face Detection, Raspberry Pi, Robotics

I. Introduction

HUMAN-COMPUTER interaction has remained somewhat stagnant in terms of how we interact with our computers. Our ”workflow” to interact with them has not had any major changes since modern operative systems were invented. This workflow would be to use some sort of input device, like a mouse or a touchscreen, to demand the computer to provide some sort of content. There have been alternative input methods proposed, but the overall paradigm has never changed. These alternative input methods, in our opinion, never reached full market potential because of one major issue: they require more work from the user, or an alteration of their traditional workflow.

Microsoft’s Hololens [**?**], one of the more recent proposals, is interesting because even though it requires a complete change in the user’s interaction pattern with their device, it presents the user with intuitive interaction patterns in the form of ”augmented reality” overlays on their field of view. The way this approach integrates use cases into a new ”input space inspired this master’s thesis.

In this Master’s Thesis, we aim to produce a physical prototype slash a proof-of-concept, that integrates a few computer vision use cases for simple use by an average user. This project builds and improves upon my own Bachelor’s final Thesis project [**?**].

The aim is to provide a simple gateway for any user to be able to take and save a picture after it has been treated by a Style Transfer neural network for story-telling and publication on social media accounts. An extra layer of interaction is provided by an emotion recognition system to provide relevant styles to the photographs, and the results can be viewed in an attached display. The robot is able to function as a picture frame and takes it’s own pictures for the user whenever it’s not in use (and has been allowed to).

II. State of the Art

Neural style transfer (NST) methods manipulate images or videos to adapt to the visual characteristics of another image. NSTs is most commonly implemented through deep neural networks. The first publication using neural networks to perform NST was the work of Leon Gatys et al. [**?**] on 2015. This paper used a VGG19 architecture pre-trained on the ImageNet dataset.

Just a year earlier in 2014, Ian Goodfellow’s groundbreaking Generative Adversarial Networks paper [**?**], proposed a new framework for estimating generative models using two, simpler models. Goodfellow’s team was looking for an alternative to the, at the time, state of the art undirected graphical models with latent variables such as restricted Boltzmann Machines [**?**], [**?**]. The main problem with this approach at the time, was that it was intractable for all but the most simple problems. The alternatives that did not involve defining a probability distribution explicitly, meaning that they could be trained by back-propagation, were in the form of generative stochastic networks [**?**]. This approach, however, still required Markov chains for sampling. Goodfellow’s proposed Generative Adversarial Networks (GANs from now on), does away with the need of Markov

Author: Vernon Stanley Albayeros Duarte, stanley.albayeros@gmail.com
Advisor: Fernando Vilariño, Computer Vision Centre, Universitat Autònoma de Barcelona
Thesis dissertation submitted: September 2021

chains for sampling, and because GANs do not require feedback loops during generation, they can leverage piecewise linear units, and improve the backprop performance.

This ”family” of networks has, over time, proven useful for NST. Style transfer is a field that manipulates images or videos in order to apply the ”style” or appearance of another image or video. Using GANs for style transfer was first explored by Tero Karras et. al in their paper: A Style-Based Generator Architecture for Generative Adversarial Networks [**?**]. An example of style transfer can be seen in Figure **??**.

While GANs have been proven to be on the bleeding edge regarding performance with NST, use cases with reduced computational capabilities might not take full advantage of them.

[h]

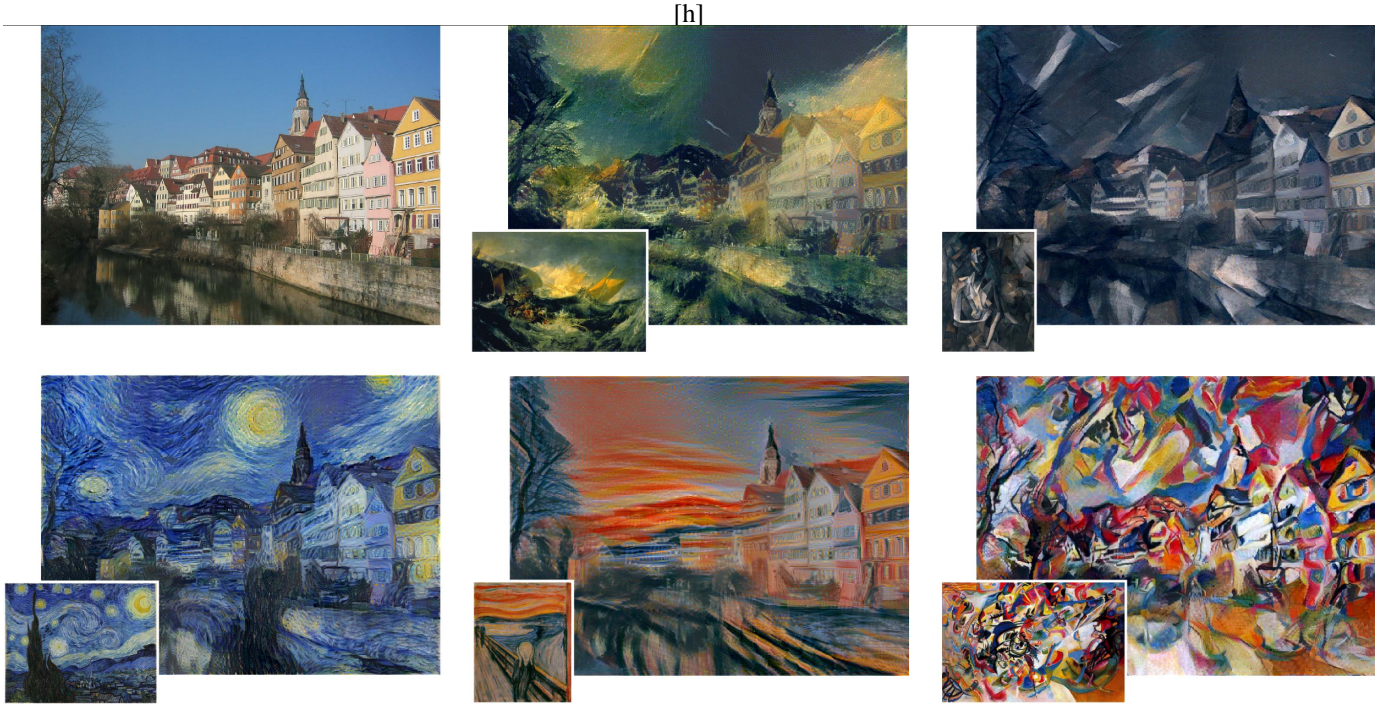


Fig. 1: Style transfer examples

III. Generative Adversarial Networks

Goodfellow’s approach was to use two models to solve the problem. The ”generator” network generates candidates in the form of randomly sampled noise from true data distribution, while the ”discriminator” network evaluates these samples. These two networks play a min-max game where the generator tries to generate ”fake” samples that it thinks the discriminator will think are real, and ”real” samples that it thinks the discriminator will think are fake, while the discriminator tries to ”beat” the generator network by correctly classifying the incoming samples. Over the years after Goodfellow’s original GAN paper, the performance of these networks has improved quite a bit. Alec Radford’s DCGANs [**?**] proposed the use of convolutional networks in place of fully connected networks, which improved the results and performance of the models. Further improvements for implementing GANs on large scale images such as Andrew Brock’s 2018 [**?**] paper on the subject, further proved that GANs could be used for larger resolution images.

IV. StyleGAN

Tero Karras from NVIDIA proposed in 2018 an alternative GAN architecture to implement a style-transfer network. [**?**]

While the latent space code is provided to the generator network through it’s input layer, Karras chooses to omit the input layer altogether and start from a pre-learned constant.

NVIDIA’s team provides their novel generator with a direct means to generate stochastic detail by using explicit noise inputs, as single-channel images consisting of uncorrelated Gaussian noise. These noise images are broadcasted to all feature maps using the learned per feature scaling factors and then added again to the output of their respective convolutions.

Karras et al. had amazing results using this approach, as seen in Figure **??**.

[h] 

Fig. 2: StyleGAN results

In 2019, NVIDIA's team set out to improve StyleGAN by targeting several of it's characteristic artifacts [**?**]. In this paper, the team sets out to redesign the generator normalization and regularize the generator to encourage good conditioning in the mapping from latent codes to images.

To remove normalization artifacts, NVIDIA's team identifies an AdaIN operation that normalizes mean and variance of every feature map sepearately, destroying any information found in the magnitudes of the features relative to each other. They instead remove the normalization step from the generator, causing the droplet artifacts to dissapear.

On the other hand, the generator's architecture is revisited completely. Where the original StyleGAN applies bias and noise within the style block, more predictable results are obtained by their modification, which is moving these operations outside of the style block, operating on normalized data. They also remove the application of bias, noise and normalization to the constant input without any observable drawback in performance or quality.

A. StyleGAN2

Later in 2020, Karras' team releases the "StyleGAN2" paper [**?**]. This time around, Karras and his team set out to improve their StyleGAN to work with limited amounts of data. They propose the implementation of an adaptive discriminator augmentation mechanism that stabilizes training when the dataset used is small. They demonstrate that results are not greatly affected even without modifying the loss functions or the architecture of the network.

We tried implementing a variant of the StyleGAN2 for use in our project initially, but our results proved to be too slow, and too poor compared to Karras' team's release.

V. Method

In this section, we will describe the methods used for face and emotion recognition. We also discuss the datasets and propose the methods we will use to make use of them. All of these methods are proposed with consideration that we need to achieve a minimum viable product for our robot. As this is a project oriented to produce a physical prototype, we have to consider that experimental results will have an impact when deciding the final implementation of the features and how much of a compromise between quality and performance we are willing to make.

To achieve a MVP (Minimum Viable Product) and be worthy of a MSc project, we set our goals as the following:

- Remove the need to use Google's Cloud API for all computer vision tasks, implementing local solutions.
- Implement new routines involving engaging computer vision algorithms like Style Transfer.
- The user should easily be able to obtain the style transferred files.

- The visual representation on the screen should be at least visually pleasing enough that users will want to keep it on as a background element.
- Make the robot completely autonomous: remove the need to offload computation to a 3rd party service or a local machine.
- Achieve a real-time representation of the robot's image processing.

A. Face detection

Previously, our pipeline for the robot required sending images to Google's Cloud Services (GCS) for emotion recognition. As we have removed GCS, this is no longer the case. To replace the previous face detection + emotion detection parts of the pipeline, we need to implement local methods face detection and emotion detection based off state of the art methods. Face detection allows the arm to follow a subject should it approach the limits of the camera's viewing angle. We propose using a Haar-Cascade classifier to detect faces.

B. Emotion recognition

Detecting faces in the frame also serves to improve emotion detection, as described in Gunwan et al. [**?**], a neural network trained on cropped faces performs optimally on similiarly cropped images. We propose training a network on the FERPlus [**?**] dataset.

C. FERPlus Dataset

The FERPlus dataset, published by Barsoum et. al [**?**], is an improvement over a the previous FER [**?**] dataset published for a *Kaggle.com* research competition.

FER, prepared by Pierre-Luc Carrier and Aaron Courville. Each image measuring 48 pixels wide and high the dataset consists of 28k images. The dataset annotations consist of numeric codes ranging from 0 to 6 for the emotion present in the image file: *(0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral).* The FERPlus dataset improves the previous FER dataset by re-tagging the dataset manually**??**. FERPlus adds more classes, bringing it up to 8 emotions total. Each image has been labeled by 10 independent "taggers" (humans), which means that each face image has a distribution of emotions.

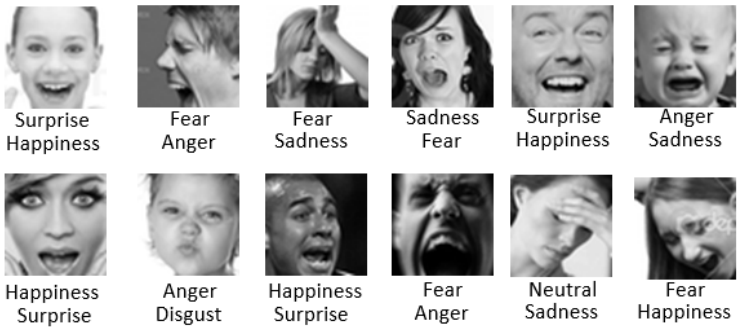


Fig. 3: Comparison of FER and FERplus labels.

While FERPlus authors used a custom VGG13 model for training, we propose using a recent VGG model (VGG19) for our implementation.

D. Style transfer method proposal

Emotion detection will be used to change the context of the style transfer network, loosely adapting the target style transfer to a user's visible emotional status.

We propose a local implementation of a neural network capable of producing style transfer images as close to real-time on the Raspberry Pi 4's hardware as possible. The models can be pre-trained in a more powerful environment, but inference should run completely locally on the device, so no external connections or dependencies are needed in run-time. We propose experimenting with two different architectures: a recent model that requires more potent hardware but has better performance (results), and an older model that does not require high-end hardware, producing worse results, but potentially giving us faster results and allowing the robot to run inference locally. Thus, we propose adapting Karras' StyleGAN2 representing the "state of the art" method, and since we are already using VGG16 and VGG19 models for our emotion recognition, adapting the method proposed by Justin Johnson et. al on their *"Perceptual Losses for Real-Time Style Transfer and Super-Resolution"* [**?**] publication representing the "old but fast" method to run on our hardware. While StyleGAN2 was trained using the FFHQ

dataset [**?**], we propose using Microsoft's COCO 2014 training dataset [**?**] to train our style transfer models, as we are not using only face images on our style transfer.

If one of these two methods is able to run locally while acheiving a good framerate, it should be strongly chosen for the final implementation, as it would complete our final two objectives for an MVP.

E. COCO 2014 dataset

Microsoft's COCO 2014 training dataset was released as part of a larger publication containing training, validation and test images. Totaling 328K images, it was the largest and most curated dataset at it's publication. The dataset contains 80 different classes and multiple captions per image.

VI. Experiments

- Fast neural Style: pytorch + vgg19 setup.
- Emotion detection: FER
- face detection haar cascades
-

A. Face and emotion detection

We used a traditional Haar-Cascade classifier to detect faces within the camera's frame. To test the accuracy and detection rate, we recorded 5 short videos, 400 frames in length each. These frames were used to test the time it took the classifier to produce a result, and the accuracy of these results for our camera's resolution and use-case.

The features we are looking for in our frames are both faces and eye features, because we not only want to keep a person within the camera's frame, but we also want to pass a cropped photo of the person to the emotion detector, which performs best when both eyes are visible.

Regarding emotion detection, we used the CNN provided by Octavio Arriaga [**?**]. On their publication, they report 66% accuracy on the FER dataset on their "sequential fully-CNN".

When training this on FERPlus, we simplified the number of classes to 5 as we found that the robot's behaviour was too erratic with more classes. Our chosen emotions are happiness, surprise, fear, sadness and anger.

We measure the performance of our face and emotion detection algorithms by how much time it takes for them to produce a result, as each passing frame affects the quality of the final product.

B. Style Transfer

We started by taking Karras' team StyleGAN2 network description and adapting it to run on the Raspberry Pi 4s hardware.

The purpose of using Karras' model was to try to minimize the requirements for inference of a trained model following their architecture. Our logic was that minimizing the footprint of the model, inference would be fast enough to complete in real time for our robot. In our early testing, reducing the input images to even below 100 x 100 px produced a model that had an inference time of around 30 seconds per frame on the Raspberry Pi hardware. We decided that the optimizations required to make this model work at a more reasonable speed were outside of our expertise, as we quickly realized that it required a deeper understanding of the Raspberry Pi's hardware to make use of platform-specific instructions.

We moved onto implementing Justin Johnson's solution on our platform. Jhonson's original publication used Lua with Torch packages to implement a feedforward neural network to improve the speed of the original Gatys et. al paper's results. This implementation takes around 50 milliseconds per frame on a 1200 x 630 pixel image, which seems promising.

We translated Jhonson's network architecture to be used in Python with Pytorch 1.9. We noticed that, as Jhonson points out in his publication, changing instance normalization for batch normalization has a very sizeable impact on speed performance at the cost of very minimal visual performance, as seen in in an example on figure **??**.

More importantly, the average processing time for a frame using an instance normalization model was of 1.25 seconds, while batch normalization averaged 0.48 seconds per frame of processing time.

TODO: fast neural style with vgg19

C. Robot state diagram

VII. Results

- style change on the fly performance (time per frame)
- generate high resolution style transfer + save+send image performance (seconds)

A. Preprocessing: face and emotion detection

Towards real-time: detecting face-¿crop-¿emotion detector if face is persistent

Explanation about the performance evaluation procedure and results analysis.

Fig. 4: Comparison of instance vs batch normalization

B. Style transfer

-¿ switch style context if emotion is persistent.

This train of thought proved to be flawed, as we minimally improved the speed of inference of the models, while taking a massive hit on image quality for style transfer.

VIII. Conclusions

In this Computer Vision Master's thesis we have implemented machine learning and traditional computer vision methods on a physical prototype capable of functioning at real-time.

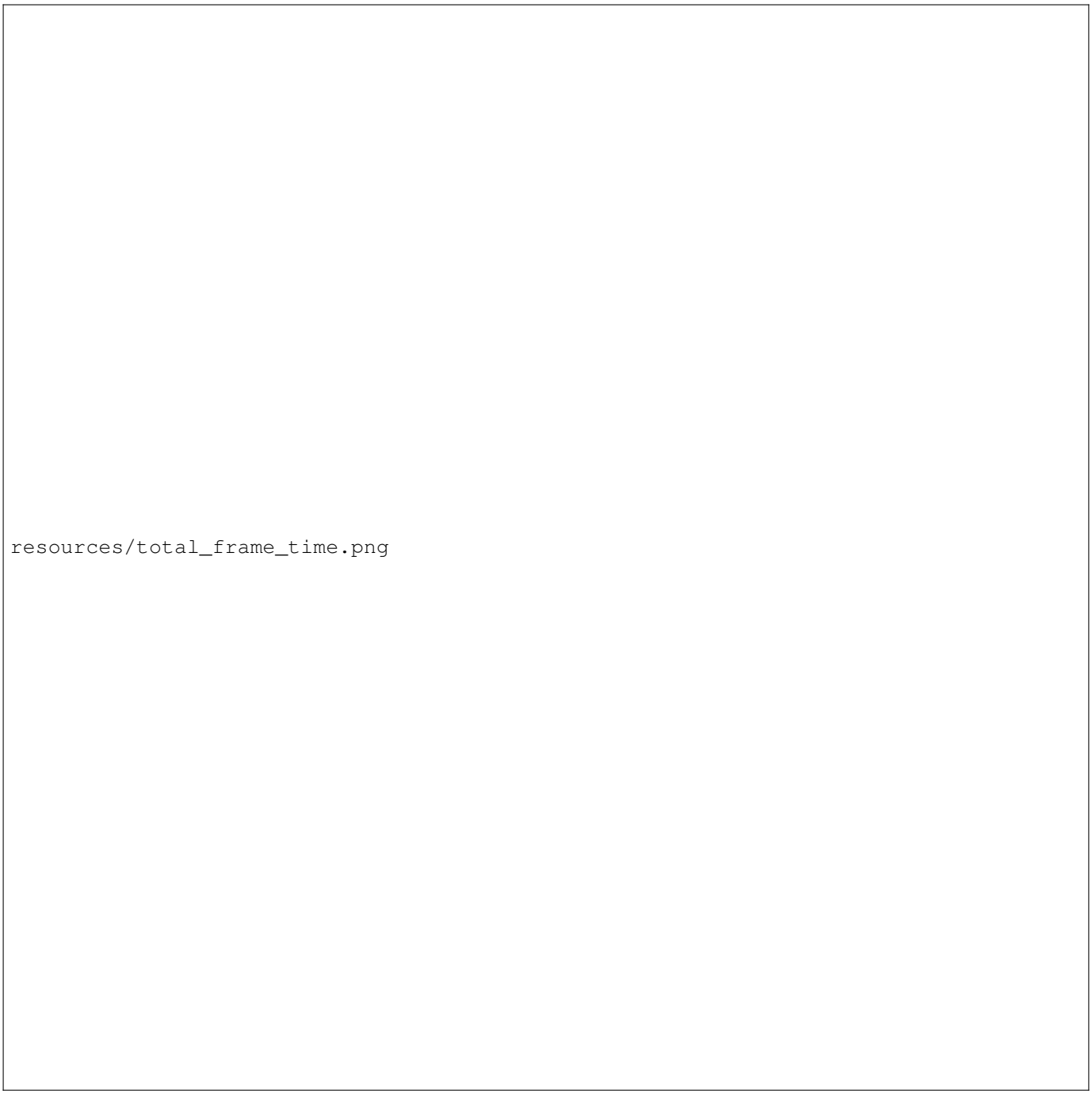


Fig. 5: Total time consumed per method each frame.

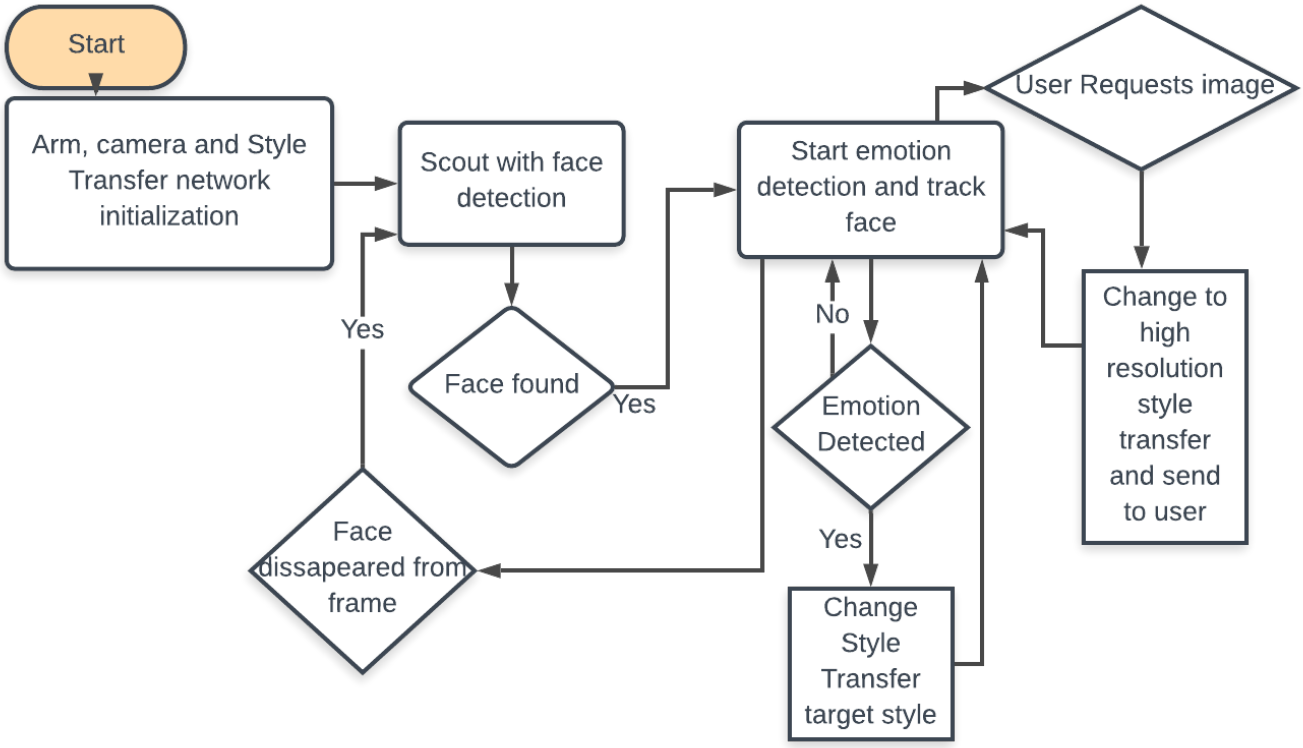
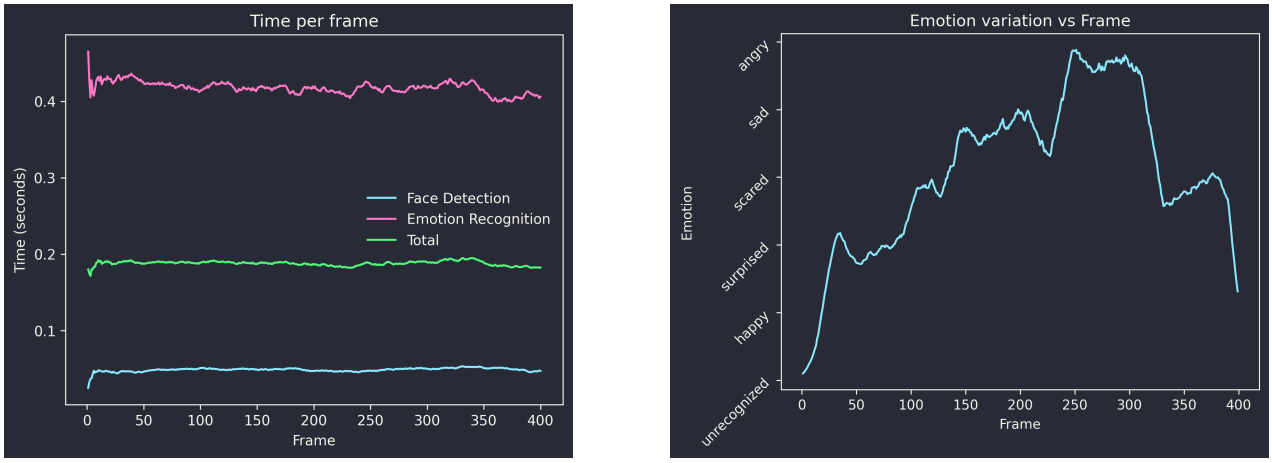


Fig. 6: Robot flow diagram.



(a) Time needed to process a frame.

(b) Emotion variation by frame of a sample.

Fig. 7: Graphs for face and emotion detection performance.

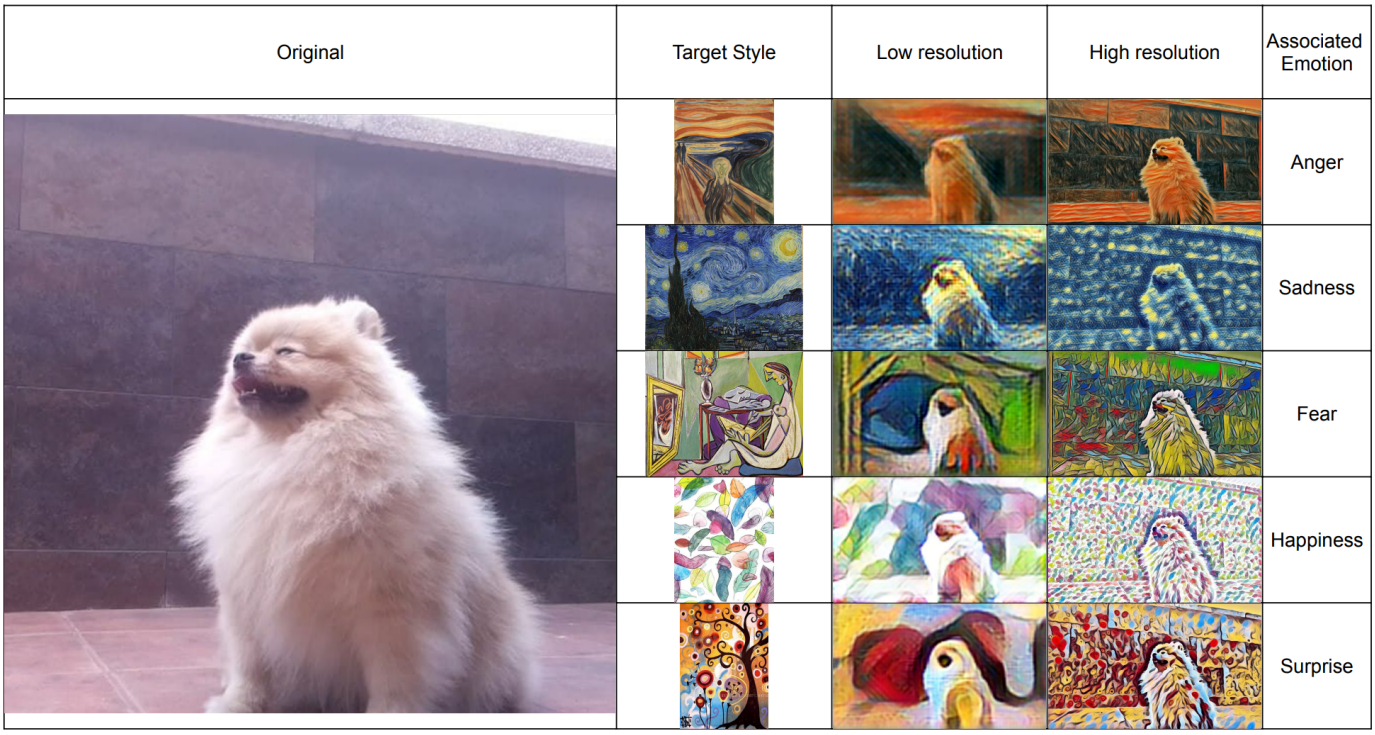


Fig. 8: Available style transfers for emotion context switching sample. From top to bottom: *The scream by Edvard Munch, The Starry Night by Vincent Van Gogh, La Musa (Mujer Leyendo) by Pablo Picasso, Feathers b Kathryn Corlett, June tree BY Natasha wescoat.*

APPENDIX A
APPENDIX TITLE

Appendix one text goes here.

ACKNOWLEDGMENT

The authors would like to thank...

REFERENCES

[1] Microsoft, 2021. [Online]. Available: https://www.microsoft.com/en-us/hololens/
[2] S. Albayeros Duarte, "Face-following robot arm with emotion detection," 2019. [Online]. Available: https://ddd.uab.cat/pub/tfg/2019/tfg-151968/Final.pdf
[3] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," 2015. [Online]. Available: http://arxiv.org/abs/1508.06576
[4] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.
[5] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, p. 1527–1554, Jul. 2006. [Online]. Available: https://doi.org/10.1162/neco.2006.18.7.1527
[6] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. MIT Press, 1986.
[7] Y. Bengio, E. Thibodeau-Laufer, and J. Yosinski, "Deep generative stochastic networks trainable by backprop," *CoRR*, vol. abs/1306.1091, 2013. [Online]. Available: http://arxiv.org/abs/1306.1091
[8] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," *CoRR*, vol. abs/1812.04948, 2018. [Online]. Available: http://arxiv.org/abs/1812.04948
[9] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015. [Online]. Available: https://arxiv.org/abs/1511.06434
[10] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," *CoRR*, vol. abs/1809.11096, 2018. [Online]. Available: http://arxiv.org/abs/1809.11096
[11] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," *CoRR*, vol. abs/1912.04958, 2019. [Online]. Available: http://arxiv.org/abs/1912.04958
[12] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila, "Training generative adversarial networks with limited data," *CoRR*, vol. abs/2006.06676, 2020. [Online]. Available: https://arxiv.org/abs/2006.06676
[13] T. Gunawan, A. Ashraf, B. Riza, E. Haryanto, R. Rosnelly, M. Kartiwi, and Z. Janin, "Development of video-based emotion recognition using deep learning with google colab," *TELKOMNIKA Indonesian Journal of Electrical Engineering*, vol. 18, pp. 2463–2471, 10 2020.
[14] E. Barsoum, C. Zhang, C. Canton Ferrer, and Z. Zhang, "Training deep networks for facial expression recognition with crowd-sourced label distribution," in *ACM International Conference on Multimodal Interaction (ICMI)*, 2016.
[15] 2013. [Online]. Available: https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data
[16] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*, 2016.
[17] NVlabs, "Nvlabs/ffhq-dataset: Flickr-faces-hq dataset (ffhq)," Dec 2019. [Online]. Available: https://github.com/NVlabs/ffhq-dataset
[18] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312
[19] O. Arriaga, M. Valdenegro-Toro, and P. Plöger, "Real-time convolutional neural networks for emotion and gender classification," *CoRR*, vol. abs/1710.07557, 2017. [Online]. Available: http://arxiv.org/abs/1710.07557