Co-funded by the
Erasmus+ Programme
of the European Union

ERASMUS MUNDUS MASTER IN
IMAGE PROCESSING AND COMPUTER VISION

PÁZMÁNY PÉTER
CATHOLIC UNIVERSITY

université de BORDEAUX

UAM
Universidad Autónoma de Madrid

MSc THESIS

Real-Time Pose Estimation for Artistic Visual Effects

PRESENTED AT

UNIVERSIDAD AUTONOMA DE MADRID
ESCUELA POLITECNICA SUPERIOR

Departamento de Ingeniería Electrónica y de las Comunicaciones

Author: MORENO VILLAMARÍN, David Eduardo

Academic Supervisor: SAN MIGUEL AVEDILLO, Juan Carlos
Industrial Consultant: MASNERI, Stefano

DATE: June, 2020

Thesis Proposal Form

Student

Name: David Eduardo Moreno Villamarín	**Neptun ID:** JRMPQR
UAM student ID: e406900	**UBx student ID:** 21831703
Academic year of starting the IPCV program: 2018	

Supervisor

Name: Juan Carlos San Miguel Avedillo
Position: Associate Professor
University[1]: UAM

Co-Supervisors[2]

Name 1:	**Name 2:**
Position 1:	**Position 2:**
University 1[1]: PPCU / UAM / UBx	**University 2[1]: PPCU / UAM / UBx**

Consultant at company / external laboratory[2]

Name: Stefano Masneri
Position: Researcher in Digital Media
Institution: Vicomtech

Thesis

Title: Real-Time Pose Estimation for Artistic Visual Effects

Summary of the thesis[3]: The objective of this internship is to develop several deep learning applications (pose estimation, keypoints detection, action recognition) and to use them to drive the creation of visual effects that could be used in live settings such as concerts or festivals.

[1] underline as appropriate
[2] fill only if applicable
[3] to be defined by the supervisor

Detailed task description (cc. 10-12 lines)[4]: The tool developed should be able to run the whole pipeline (frame grabbing – inference – analysis – rendering) in real time on a PC with a modern GPU.
A mobile application, with limited functionality and no real-time requirements should also be developed.

Hereby I undertake to supervise the Master's thesis of the student.

Place and date: Madrid, 05/03/2020.....

Signature of supervisor

Hereby I apply for the approval of the topic of my Master's thesis and declare that I wish to take the final exam and defend my thesis at the following University[5]:

Pázmány Péter Catholic University

Universidad Autónoma de Madrid

Université de Bordeaux

Place and date: Donostia-San Sebastian, 04/03/2020

Signature of student

The topic of the Master's thesis has been approved by the Partner University where the final exam and thesis defense will take place.

Place and date: Madrid, 17/04/2020

IPCV local coordinator

[4] to be defined by the supervisor
[5] underline as appropriate

Declaration of Authenticity

(to be inserted as second page of the thesis)

I, the undersigned ……MORENO VILLAMARÍN, David Eduardo……., student of the Image Processing and Computer Vision program, hereby certify that this thesis has been written without any unauthorized help, solely by me, using only the referenced sources. Every part quoted in whole or in part is indicated clearly with a reference.

I declare that I have not submitted this thesis in any other higher education institution, excluding the partner universities of the IPCV Consortium.

.....

Student

Abstract

We present a set of methods that allow audiences to create artistic content in live events. The main tool of interaction consists of a pose estimation algorithm that retrieves the spatial location of people's body joints. We use the estimated poses to drive the creation of visual effects. In our first approach, we render and modify 3D visual elements according to the number of people and their poses. Our second approach creates images by means of a generative adversarial network (GAN). In this case, we use people's pose information as the input to a style-based GAN (StyleGAN2). The style-based architecture allows us to mix high-level attributes (or 'styles') of the synthesized images with other inputs. As StyleGAN2 maps its input to an intermediate representation, we propose to sample 'styles' in the intermediate representation by using a clustering method. We later mix 'styles' by applying different inputs to distinct layers of the network. In our work, we jointly use audio features and poses to guide the appearance of the generated images. In this case, the audio features guide a trajectory over a set of previously sampled 'styles'. Previous work suggests using principal component analysis (PCA) to create interpretable controls for generating images. We apply this technique so smaller groups of people are able to drive the generated image attributes in an interpretable manner. We further extend our set of frameworks with an off-the-shelf image animation method that presents users with the possibility to control a character's face with their own facial expressions. We evaluate the feasibility of our methods by studying their run-time performance. We also analyze how applying a scaling parameter to the input of the GAN relates to the diversity of the synthesized images. Finally, we present some qualitative results of our framework on different video sequences with varying numbers of people. The resulting videos can be found in our drive folder: https://drive.google.com/open?id=1MsjQFJFAzvLHN-_QVKpQsTWPUOYJlnuz.

Contents

Internship

This chapter describes the internship part of the work carried out at the Digital Media department of Vicomtech. Vicomtech is a technological centre specialised in artificial intelligence, visual computing and interaction. The objective of the internship is to develop applications for analyzing audiences in music festivals and creating visual or artistic effects. This work is part of a project that involves the development of a multi-service and multi-device platform for music festivals, where the platform's objective is to achieve a hyperconnected event. Previous work in this project includes the analysis of audiences via computer vision and wireless methods for measuring their engagement during an event [54]. The information of the previous work can then be provided in real time to the festival's management and users to enhance their experience, for instance, by creating visual content. Therefore, this work focuses on offering guests the possibility of creating and interacting with visual and artistic effects.

This chapter is organized as follows: first, we give a description of the research centre in Section 0.1. Later, Section 0.2 describes the festival platform, and how our work takes part in the project. Section 0.3 documents the system's requirements, while Section 0.4 clarifies the system's specification. Section 0.5 details the system's design.

0.1 Vicomtech

Innovations for artificial intelligence, visual computing and interaction are the main areas of Vicomtech's research. In these fields, Vicomtech collaborates with universities, businesses, and other research centres through technology transfer. Moreover, Vicomtech partners with industries in different sectors, such as digital media, e-Health, digital security, manufacturing and process industries. By transferring technology, the goal of the centre is to help businesses to be more competitive. Transferred technology includes methods for human-computer interaction, which comprise gestures, virtual and augmented reality. The scientific work performed in Vicomtech translates into top-level publications in the international field.

In the visual computing and interaction areas, Vicomtech employs technologies for capturing, transmitting, and storing different multimedia formats. In addition, their work applies analysis and interaction with visual content. With this respect, Human Computer Interaction is at the core of Vicomtech's research. The centre's application fields include gesture-based interactions, visual analytics, virtual and augmented reality.

Main research topics are additionally in the areas of artificial intelligence models applied to solve real-world problems. For instance, with regard to the analysis of industrial data, the centre uses Data Intelligence and Visual Analytics approaches for solving problems in advanced manufacturing, oil and gas industries. Other branches of work focus on language technologies, for instance in machine translation, transcription of media, and speech recognition. Moreover, Vicomtech participates in research and development applied to the medical sector. Decision-making support systems and medical image processing methods help discovering and segmenting tumours and planning surgical interventions. The centre also applies AI models to cybersecurity, autonomous driving, and biometric human activiy recognition.

This work is part of the Vinim project within the Digital media department of Vicomtech. The Digital Media department performs research in multimedia flow management, which includes the encoding, encryption and distribution of audiovisual content. For this, the department pioneers solutions that maximise Quality of Experience (QoE) and highlights new opportunities provided by 5G technologies. Moreover, the department researches and offers solutions in the fields of immersive and augmented multimedia content production. Other technologies of Digital Media are applied to interaction with hybrid broadcast internet content via multiple screens.

0.2 Context

Our work forms part of Digital Media's Vinim project for developing a platform for music festivals. Vinim is a three year's project which focuses on creating a product that enables a hyperconnected music festival through information and communication technologies. Currently, this is the third year of the project. The desired outcome is to allow guests to follow the festival from their homes or different concert halls, and offer a unique experience. For instance, spectators are able to watch video streams from each concert hall and obtain social network information. Here, a key element is that users can interact in an engaging manner with the performances and be part of the artistic creation.

Previous stages of Digital Media's project include a module for measuring audience engagement. The module applies computer vision algorithms along with wireless signal analysis techniques for detecting and tracking event guests [54]. The extracted information allows measuring the degree to which an audience is involved and interested in the event. The metrics used include gaze detection, emotion analysis, and activity recognition.

0.3 Requirements

The objective of this internship is to develop different computer vision applications and use their output information for driving or creating artistic effects. The computer vision methods should obtain information from cameras pointing at the public in live settings, such as concerts or festivals. The captured information, for instance, can include the spatial position of body joints and number of event guests. Furthermore, the tool must employ the acquired information to drive the creation of visual or artistic effects.

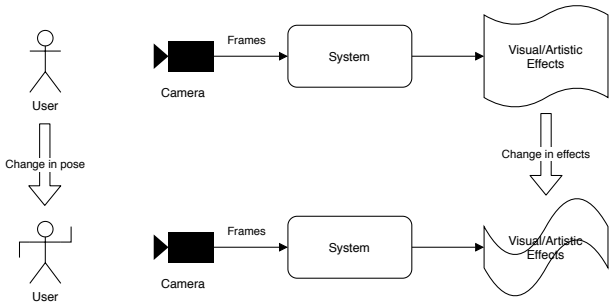


Figure 1: High-level diagram of the expected system behaviour. Any movement of a user in front of the camera should produce a change in the displayed visuals.

The visuals should ideally comprise images and abstract 3D objects that are related to the extracted information from audiences.

The developed methods should contribute to a software suite, where different tools are capable to create different content. The core of the suite, in this case, is that the artistic representation is the principal component. Moreover, the tool developed should be able to run the complete pipeline, from frame grabbing and their analysis to rendering visuals. The complete pipeline must be capable to run in real time on a computer with a modern GPU.

A use case scenario consists of a concert hall equipped with cameras pointing at the guests, while a video projector displays visuals. For instance, abstract 3D objects. A similar setting also includes a venue with cameras looking at the audience, but in this case, the tool sends visuals to a mobile application. The mobile users are other event guests outside the venue, which receive visual content representing the current state of the concert hall. Therefore, the inputs to the program consist of a video stream of the audience, while the outputs comprise the visual content. This is illustrated in Figure 1.

0.4 Specification

We can specify the expected behaviour of the system as follows: users of the system should be able to generate visual or artistic effects by participating with the event. In this case, users participate with the event by standing and moving within the field of view of an RGB camera. Any change in a user's pose translates into a change in the displayed visual effects. Furthermore, the inputs to the program consist of a video stream of the audience, while the outputs comprise the visual content.

We can define a basic use case as shown in Figure 2a. Here, given that the system is able to capture any movement of a user, the system translates the relevant changes into the visuals. Furthermore, we can define a scenario when a new user joins the interaction. In this case, the sole action of joining causes the system to update the visuals (Figure 2b). An additional scenario includes the possibility that the festival platform can receive and further stream the generated visuals to remote users, which could be spectators or the event manager (Figure 2c). We observed that there is the possibility to add other inputs

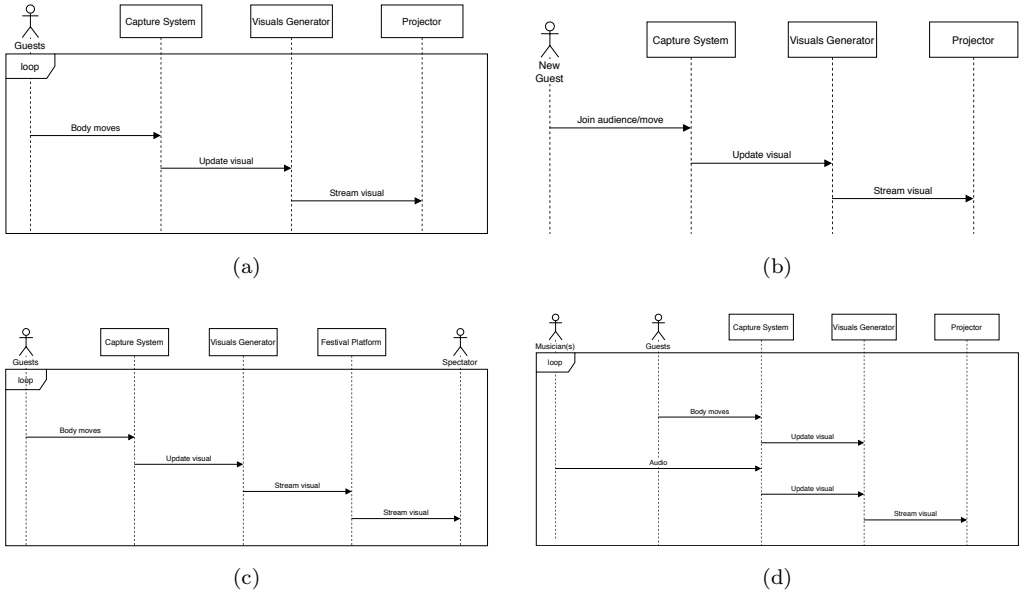


Figure 2: (a) Main use case scenario: movements by a user or group of users result in changing the visuals. (b) New guest scenario: a guest joins the interaction and the system updates the visuals. (c) A remote event spectator can receive the visuals generated by the audience. (d) Musician participating scenario: the music played by musicians also affect the visual generation.

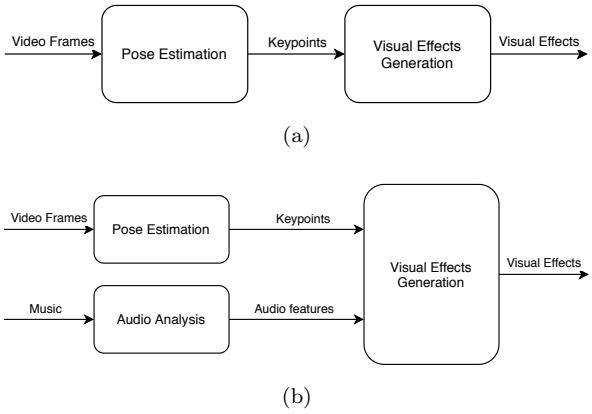


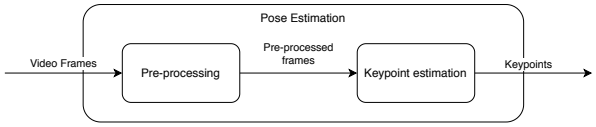
Figure 3: (a) Block diagram of the system with the pose estimation and visual effects generation modules.

to influence the generated visual effects. Therefore, we consider the case where a musician or group of musicians can also participate in the generation process with their music (Figure 2d).

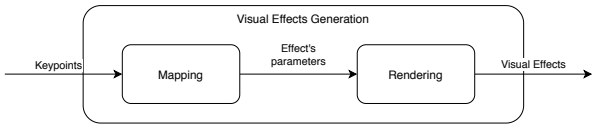
Regarding the real-time constraint, we consider that achieving a refresh rate of the visuals above 15 frames per second (FPS) is enough. However, 30 FPS is highly desirable. Therefore, the duration from capturing frames to updating the visuals should not exceed 66.7 ms and should ideally be below 33.3 ms.

0.5 Design

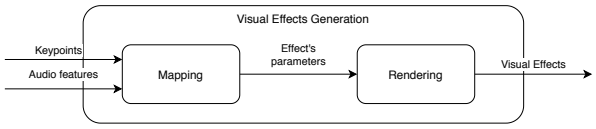
The way a user interacts with the environment is defined by their pose, or body joint position. Therefore, the system can be divided into a pose estimation module and a visuals generation module. In this case,



(a)



(b)



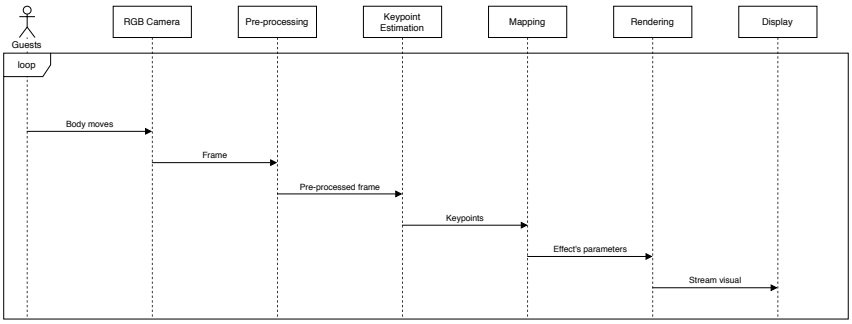
(c)

Figure 4: Pose estimation and visual effects generation modules. (a) The pose estimation module includes a functionality for pre-processing the incoming frames, so that the keypoint estimator is able to process them. (b) The visual effects generation module includes a mapping functionality that converts the keypoints to parameters that the rendering module uses to produce visual effects. (c) When using audio inputs, the mapping module of visual effects generation module also takes as input audio features.

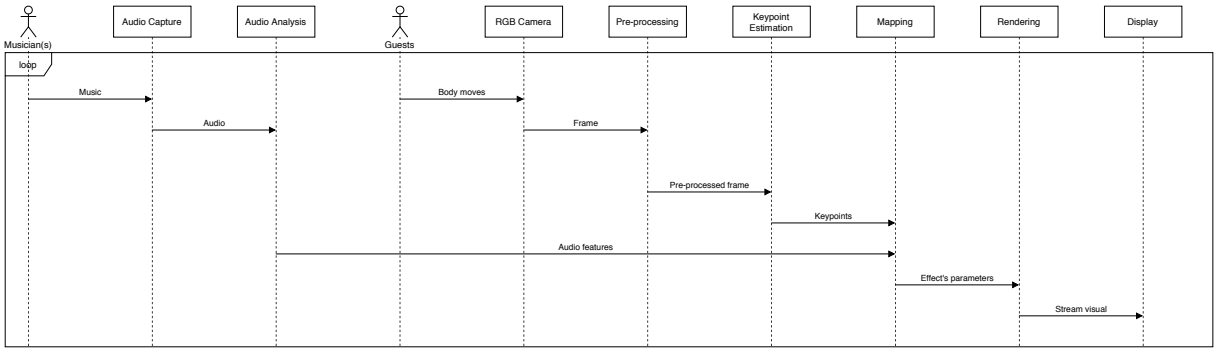
the system feeds output of the pose estimation module directly to the visuals generation module. This is illustrated in Figure 3a. The pose estimation module should produce an array containing the spatial position of all body joints of a user or group of users. Later, the visuals generation module has the task of processing all body joint positions and mapping them to an artistic representation. We also consider a system able to process audio inputs from the performers' music. Here, the visuals generation module takes as inputs both the keypoints detected in the pose estimation module and audio features extracted in an audio analysis module, as depicted in Figure 3b.

The pose estimation module contains a functionality to **pre-process** the incoming frames to convert them to an acceptable format by a **keypoint estimation** algorithm. The keypoint estimation algorithm is in charge of computing the spatial locations of users' body joints. This is depicted in Figure 4a. Moreover, the visual effects generation module should first map the incoming information to parameters that the rendering module can interpret to generate visuals. As shown in Figure 4b, the **mapping** module processes and analyses the keypoints to produce parametrized information for the **rendering** module. The visual effects generation module can additionally consider audio features as an input, as represented in Figure 4c.

Using the aforementioned design elements, we can further model the use cases as shown in Figure 5. We must know that the whole pipeline shall run under the set time constraints. Moreover, the process is sequential, and each subsequent module does not require to further communicate with preceding modules in order to execute its task.



(a)



(b)

Figure 5: (a) Main use case scenario of Figure 2a with design elements. (b) Scenario with musician participating of Figure 2 with design elements.

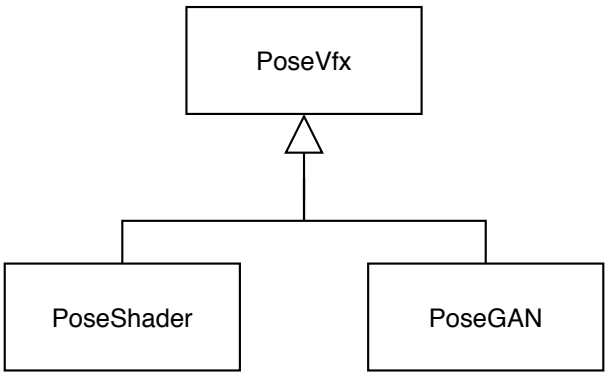


Figure 6: We use a base class for our framework called `PoseVfx`, which contains methods and properties for capturing and pre-processing frames. The `PoseShader` class contains methods for calling the keypoints detection algorithm and for rendering effects. The `PoseGAN` class similarly calls the keypoints detection algorithm and the synthesis function of StyleGAN2 for generating images.

0.6 Development

We developed our system using the Python programming language along with data science and graphics libraries, such as SciPy, NumPy, SciKit-Learn [64], Tensorflow [1], OpenCV [8], Dlib [35], and PyOpenGL. For audio processing, we use Aubio [10].

Our system employs the algorithms described in Chapter 4 for pre-processing and estimating keypoints

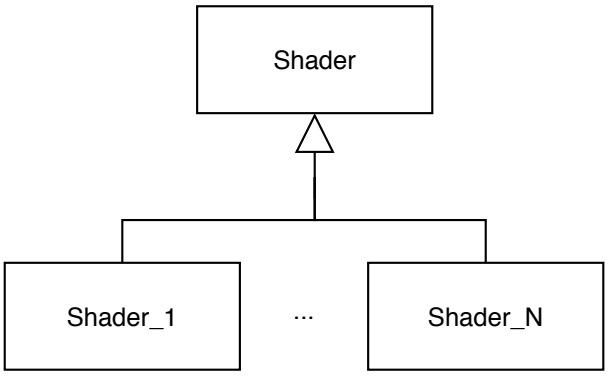


Figure 7: Our framework uses a base class for the shader wrappers called `Shader`, which contains methods and parameters for interacting with a shader. Each distinct shader uses a class wrapper that inherits the methods and parameters from `Shader`, allowing to interact with the shader's specific properties, such as objects' size and colour.

in our pose detection module, and generating effects in the rendering module. We use two different approaches for the visual effects generation module. The first approach uses shaders to render 3D objects, and the second approach uses a deep generative model [34] to produce 2D colour images, as we further clarify in Chapter 4. We define a base class `PoseVfx` containing the pre-processing methods and properties, as shown in Figure 6. Both of our approaches inherit from `PoseVfx` to capture and pre-process frames, perform detection and produce 3D or 2D visual elements.

In our shader approach, we define individual mapping functions for each shader. Each function maps keypoints to specific shader parameters, such as object size, colour palette, movement, and shape. All shaders are specified by the OpenGL Shading Language (GLSL), and we provide a class wrapper that allows our program to parametrize each shader. We use a base class that defines methods and properties for the class wrappers of each distinct shader, as we show in Figure 7.

In our generative approach, the mapping function corresponds to different steps. We first normalize the detected keypoints and multiply them by a scaling parameter. Later, the generative algorithm contains the mapping and rendering modules. Since we include the audio option, the audio analysis module is in charge of extracting the pitch and energy spikes of a signal's window. Therefore, the audio features are represented, in this case, by the pitch and the energy spikes. Moreover, our deep generative technique maps and renders 2D effects.

In the following chapters we expand on the context of this work within the creative industries, and further specify use cases followed by a thorough state-of-the-art review. We continue illustrating the methods we employ to develop our framework and present our results. We conclude our work with by summarizing our contributions and add some final remarks about future directions.

Chapter 1

Introduction

The perpetual and rapid developments in artificial intelligence (AI) have empowered the entertainment industry with a myriad of possibilities, including the way content is created and how users interact with it. According to the World Economic Forum, new developments in AI are having disruptive effects in the creative economy [20]. Moreover, complex machine learning (ML) models have become progressively efficient, allowing their operation in live settings such as concerts or festivals. These type of events can benefit from capturing real-time data of their audience and use it to enhance their experience.

The entertainment industry has primarily obtained user's feedback via surveys, questionnaires, and interviews [14, 7]. However, the transition to digital and online media has changed the panorama, offering alternative data formats related to users' interaction with content. This type of information is valuable to content providers, as they can use it to understand how consumers feel about products [20] and improve users' experience in the future.

In a similar fashion to how technology revolutionised the entertainment industry from video editing to graphics generation [3], recent developments in AI promise to take the industry to a new era [2]. Computers have been traditionally viewed as canvases, instruments or tools to aid artistic activities, as

Figure 1.1: Image style transfer. Source: [17].

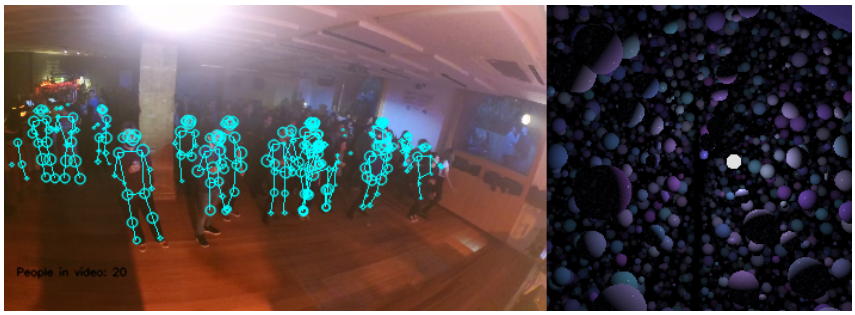
Figure 1.2: Generative Adversarial Networks are able to generate photorealistic images. Source: [34].

creators have always looked towards new ways to enrich their work [40, 65]. More recently, AI is helping content creators by performing otherwise time consuming tasks, matching content with audiences, or enhancing interactivity [20]. Furthermore, ML algorithms have become capable of generating content in different sectors such as music, fashion, and art. For instance, style transfer methods can be employed to modify pictures to look like a painting (Figure 1.1) [17, 26, 70], generative algorithms have learned to produce photo-realistic images (Figure 1.2) [18, 32, 9, 33, 34], and it is expected that AI will evolve to create more complex content [20]. Therefore, AI methods can now be simultaneously involved in interaction and content creation to create engaging experiences.

Vicomtech, a technological centre specialised in artificial intelligence, visual computing and interaction, is currently developing a multi-service and multi-device platform for music festivals. The platform's objective is to achieve a hyperconnected event, which benefits from user generated data. One of the features of the platform consists of measuring a live audience's engagement. In this line of work, audience measurements are performed using wireless and vision-based detection techniques. This information can be provided in real time to the festival's management and users to enhance their experience [54]. Moreover, the measured data also has the potential to allow the public to interact in an engaging manner with visual content in the event venue.

This work, within the scope of the aforementioned platform, proposes methods for providing audiences with the possibility to create artistic content in real time. For this matter, the tools developed jointly employ AI and visual computing methods. The backbone for user interaction consists of detecting the body joint position of event guests. This information acts as the main controller for participants in order to generate visually compelling media. A possible setting would consist of a concert hall equipped with cameras looking at the audience, and a video projector displaying visuals while musicians are playing. The displayed visuals are generated according to the audience's movement and the music, complementing the artists' performance (Figure 1.3).

This document is structured in the following fashion: Chapter 2 describes a number of scenarios where our developed frameworks could be applied. Later, according to the defined use cases, Chapter 3 reviews similar applications in the creative industry, along with methods we can employ in our framework. Chapter 4 describes our approach and the techniques applied for user interaction and content generation. Chapter 5 reports the procedure applied to evaluate our system, as well as the obtained results. Finally,



(a)

(b)

Figure 1.3: Examples of visual effects generated with peoples poses. (a) Audience's actions drive 3D objects. (b) Event guests drive the generation of 2D images with their movements.

Chapter 6 concludes this document along with possible future directions of our work.

Chapter 2

Use Cases

This chapter describes different scenarios, where our frameworks could be useful. These scenarios have been discussed and enhanced with real companies that work on the CCI (Creative and Cultural Industries), such as a concert hall. In Section 2.1 we illustrate how the user interaction with visual elements in a concert hall could be during a live performance. Later, Section 2.2 describes a case for a mobile application for big or distributed festivals where more than one event may happen at the same time (performances at different stages or concert halls). In Section 2.3, we present some example scenarios where users can animate the face of a character with their own expressions. Finally, in Section 2.4 we show other creative applications of the framework.

2.1 Live Show with Visual Elements

In this scenario, the concert hall is equipped with cameras looking at the public, while a performer plays music and visuals are projected behind the artist. The main goal is to automatically create context-aware visual effects that enhance the show, where the visual elements represent in some way the show itself. An illustration of this setting is provided in Figure 2.1.

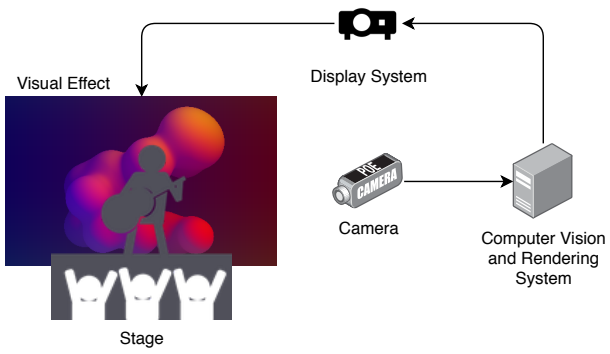


Figure 2.1: Live setting equipped with cameras recording the audience. A system processes the video sequence and automatically generates visual effects that are projected behind the performer.

In our scenario, we could take as an example a set of 3D moving objects. The objects slowly move, until a person from the public moves closer to the artist and begins dancing. As the person moves, the visuals reflect how fast the person dances. When the other guests see the interaction, they join the first dancer and dance along, while more objects in the visual illuminate and move faster. Later, the dance floor becomes more crowded and the projection lights up more objects, which also move faster. As a summary, the visual effects depend on the interaction of the guests as a whole: people's movements, engagement, and how crowded the audience is.

Another example includes projecting 2D visuals. In this case, the images represent how an artificial intelligence 'interprets' people's movements. Therefore, people from the audience dance and move to explore inside of an artificial mind. If the mind has only learned to produce images of human faces, people can try different movements to make the algorithm produce a smile. If the system only knows landscape images, the public could attempt to dance until a beautiful sunset is shown.

An additional scenario similar to the previous one could incorporate the artist's music. This presents the artist with the possibility to dictate the pace at which the systems shows images according to the music. If the music is slow, colors of the visuals change slowly, whereas a fast paced music results in rapidly changing colors and details.

2.2 Multi-Stage Music Festival App

The mobile application presents to the audience a visual representation of what takes place at each location in real time, since different locations present shows in parallel. This may help the audience to decide which event to follow, or at least have an overview of the current state of the event, without a real streaming that might have data protection issues. Figure 2.2 provides an illustration of this use case. We consider the following scenarios:

- A festival guest can look at 3D visuals in the festival app, where the visuals relate to a specific festival stage. Moreover, the objects displayed in the visuals behave according to the behaviour of the stage's audience. With this visual information, the user can decide whether to go to a certain stage or not.

- Similarly, remote spectators can also look at the visuals to decide which video stream to watch. Additionally, the displayed 3D objects can represent the audience in the video stream and reduce bandwidth usage.

- The 3D visual representation of the audience behavior can give the event managers quick insights of the audience engagement without the need to look at a video stream of the audience.

- Since the festival app shows visuals instead of a video of the audience, the bandwidth usage is reduced, and the privacy of guests is left intact.

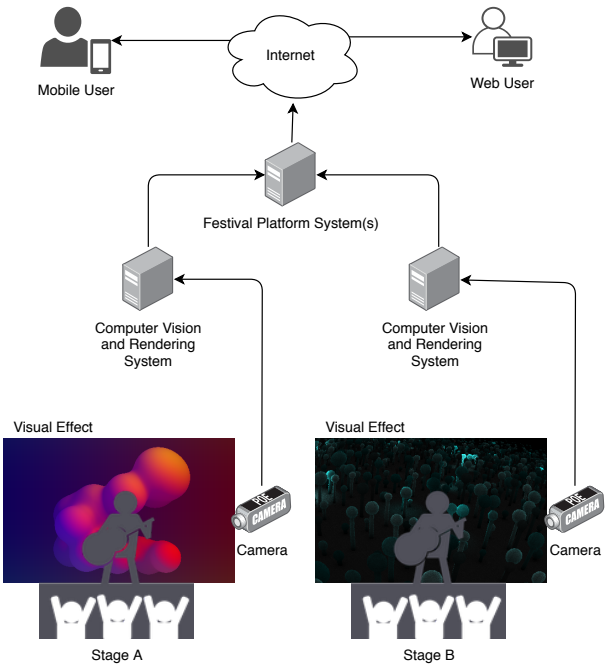


Figure 2.2: Mobile application scenario. Web or app users can look at visual elements to decide which stage to follow.

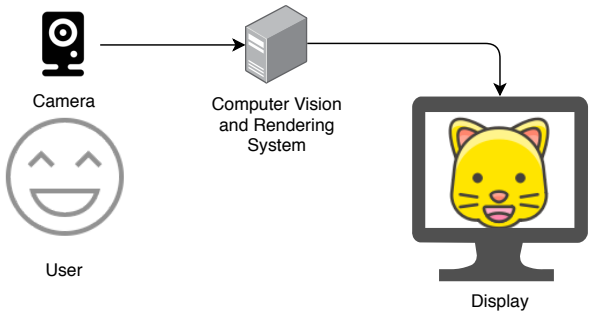


Figure 2.3: Face animation scenario. A user can animate a character with their facial expressions.

- The 2D visuals can also play similar roles to those of the 3D visuals. Moreover, since the 2D visuals also change according to the performer's music, they can also complement the video stream of a stage.

2.3 Face Animation

- **Performer Animation:** In this scenario, a projection is displayed while a musician performs. The projected visuals include a face of an animated character. In this case, as the musician moves or sings, the animated character mimics the facial expressions of the performer in real time.
- **Guest Mirror Animation:** In the concert hall, there is a monitor showing a human face. The face can belong to a celebrity, a painting, or a sculpture. In this case, a guest can approach the monitor and animate the facial expressions of the face with their own. Moreover, in special cases,

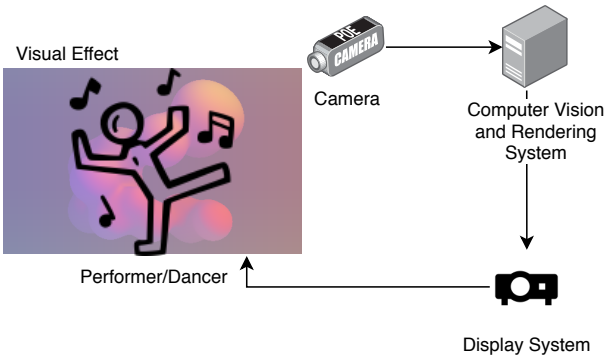


Figure 2.4: A camera system captures movements of an artist or dancer, and the proposed system renders visual elements accordingly.

random distortions and effects may appear on the face of the animation. An illustration is provided by Figure 2.3.

2.4 Other Artistic Applications

Our work presents other application scenarios for artists:

- A performer can dance along with a projection of 3D visuals, as the virtual objects completely respond to the dancer's movements. An example can be seen in Figure 2.4
- A dancer can also try different moves and body gestures while our artificial mind 'interprets' them and draws compelling images that complement the dance.
- A group of dancers can design a choreography, and in a similar manner to the previous scenario, the system can produce images related to the dancers' moves.
- In a scenario where a musician wants to display an animation that accompanies their music, the artist can configure the system to interpret solely the audio. In this case, our framework generates sequences of images that fluctuate on pair with the music.
- The performer can optionally drive the visuals by moving. In this case, the artist chooses to which degree the music controls the visuals compared to their movements.
- Another possibility is to have a dancer or a group of dancers. The performers then present a choreography with the music. At the same time, our system displays visuals driven by audio and dance moves.

Chapter 3

Related Work

As AI advances, it continues to coalesce with the entertainment industry. For instance, music festivals benefit from extracting and analyzing information from their audiences. Event organizers can enhance the offered experiences by studying user interaction with online content, such as live streams and videos on demand. Moreover, it is also possible to harvest data from the public present at event venues. By employing computer vision techniques, stakeholders can obtain information such as audience motion, interest and emotion in real time [54]. Real time audience analysis can further become a mechanism of interaction, and enhance the experience of event guests.

According to the use cases presented in the previous chapter, we review existing work regarding the analysis of audiences, and how extracted information can drive and generate content for interactive experiences. In [54], the authors develop a system as part of Vicomtech's platform for festivals, which measures how much people are involved and interested in the event. Furthermore, the system also provides information regarding peoples poses and movements during a performance. In our work, we focus on the application of pose estimation methods, since they capture information that can be used in the scenario described in Section 2.1 for interacting and creating visual content.

In addition, we give an overview of artistic applications of AI and computer vision models. On account of the democratization of technology and data, content creators have explored the roles of AI and information as expressive media. For instance, AI powered generative algorithms can now hallucinate realistic faces of people or animals that do not exist [33, 15], which has interested different artists. This work also studies the possibilities of employing generative algorithms for delivering content, albeit in live events.

Generative models, in this regard, aim at modeling a mapping from a low dimensional space to a high-dimensional distribution (such as pictures of human faces). These models have the potential to generate visually appealing images, and due to the dimensionality of the input vector, different inputs can control them. As an example, the spatial position of body joints in an image can control the appearance of generated images. Moreover, output images can potentially be displayed during an event and manipulated by the public.

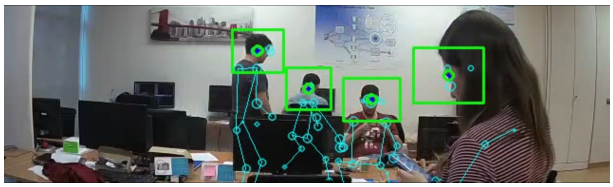


Figure 3.1: Sample frame of the pose and human detection algorithm of [54].

Estimating the pose of people has a broad range of applications, which include recognizing human activity, robotics, augmented and virtual reality. This chapter further reviews existing work related to pose estimation algorithms. Human pose estimation can be defined as the task of regressing the position of human body joints in an image. This task can be further divided into more specific objectives such as estimating the position of joints of single person or multiple people, and tracking joints in a video. In our case, we can employ pose estimation models on audiences during live events, and drive different attributes of visual effects allowing audiences to have a more engaging and interactive experience.

Moreover, keypoint estimation is part of image animation methods. More specifically, animating an image consists of generating a video sequence where an object from a source image is animated according to the motion of a driving video [58]. Such animation methods align well with the use case illustrated in Section 2.3. Therefore, we can employ image animation techniques in our framework so that event guests or performers have the possibility to animate images of their choice.

The chapter is organized as follows: Section 3.1 reviews previous work on audience analysis, Section 3.2 gives an overview of AI in art, Section 3.3 revises deep learning-based models for pose estimation, Section 3.4 gives an overview of Generative Adversarial Networks as generative models, and Section 3.5 describes image animation approaches.

3.1 Enhanced Audience Analysis

Sanz-Narrillos et al. present in [54] a system for performing audience analysis. In their work, the system employs jointly computer vision and wireless signal analysis techniques for detecting and tracking humans in live settings. Their proposed hybrid approach is robust to varying illumination and noisy environments without performing domain adaptation. The outputs of the system can be observed in Figure 3.1.

The vision-based method consists of a pose detection framework [47], which we further describe in Section 3.3. In addition, the proposed system uses detected poses to track people in the scene. Moreover, their work includes a gaze detection module to determine whether people are looking at the performers. However, the pose detector and tracker's performance decreases when tested on a live setting with low and varying illumination conditions. Therefore, the authors propose to apply a pre-processing step to the input frames of the detection algorithm. Moreover, the addition of a wireless detection scheme complements the detection system by driving the pre-processing parameters.

The wireless scheme attempts to locate mobile devices present in a scene through a tracking device. The connection handshake from both Wi-Fi and Bluetooth between the mobile and tracking device allows

to estimate the distance between them. Consequently, the wireless data provides a rough estimation of the number of people present, which complements the vision-based method. Moreover, the proposed framework later compares the outputs of both detection systems to control the parameters of the pre-processing step.

The pre-processing step aims to improve the detection accuracy of the vision-based scheme. First, the algorithm crops the input frames to a region of interest and later divides them into different zones. Subsequently, each frame slice is processed separately by a contrast stretching method [50] and gamma correction [53]. The proposed algorithm updates dynamically the parameters used for performing contrast stretching and gamma correction depending on the results of the wireless and vision-based systems.

Finally, the authors show that the hybrid framework improves the detection rate and tracking performance when used under poor and varying illumination conditions, albeit with some overhead in processing times. Nonetheless, there is still room for improving the detection accuracy of body joints.

Despite the system's limitations, an interactive tool can benefit from the information provided and drive visuals. In Chapter 4, we show how peoples poses can control different visual effects. Moreover, our proposed scheme is able to adapt to scenarios with small or large groups of people.

3.2 Artistic Applications of AI

AI is an umbrella term collecting a set of techniques capable of inferring complex structures from large amounts of data, with the aim of making predictions or taking actions on previously unknown information. As AI has evolved and its use has widely spread among different disciplines, it has become particularly visible in the media and creative industries [2].

One of the developments in AI that has grabbed the attention of creatives is DeepDream [43]. It is based on a Convolutional Neural Network and enhances patterns in images depending on the desired network activations. The resulting image is a strongly modified version of the input, while having a dream-like appearance, as presented in Figure 3.2. Although it was originally developed for understanding how convolutional networks function, its creations have resulted appealing for creative purposes.

Another artistic application of AI is Neural Style Transfer. In [17], the authors discovered that convolutional layers in a network encode different levels of abstraction from images, which can be referred as 'styles'. As an example, a network can identify style patterns in a Van Gogh painting, and apply them to a photography. The resulting image resembles a painting by the artist, as can be seen in Figure 1.1.

Moreover, generative methods based on Generative Adversarial Networks (GANs) [18] have gained significant interest in AI art. These techniques have evolved to be able to produce images with unprecedented quality, such as photo-realistic images of human faces [1], cats [2], or buildings [33, 34]. This has inspired other works on generating manga characters, paintings based on art datasets, and more. Moreover, the capability of these models to generate complex structures has also been applied to image

[1] https://thispersondoesnotexist.com
[2] https://thiscatdoesnotexist.com

Figure 3.2: Deep dream hallucinations: The original image influences what kind of objects form in the processed image, since image features bias the network towards certain activations. Here, horizon lines are filled with towers and pagodas, trees turn into buildings and leaves produce birds and insects [43]

.

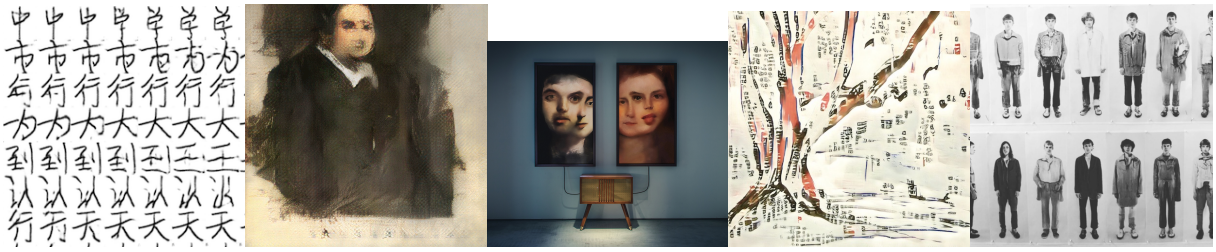
to image translation [70], which includes but is not limited to transforming doodles to images [28] or realistic landscapes [49].

The overall interest in machine learning and computer vision applications for creative domains is most evident in visual content generation and computer vision for fashion. Research in these topics, from the ICCV Workshop on Computer Vision for Fashion, Art and Design[3], includes using generative models for assisting fashion designers [56], or generating music from visual works, as explored in [44].

On the other hand, AI has begun to compete in the art market while artists explore its role as expressive media. For instance, Gene Kogan has worked on numerous AI powered projects, aided by DeepDream, GANs and other computer vision techniques for action and object recognition. In 2018, an art collective sold a portrait produced by a GAN for $432,000 at an auction, which was expected to sell for less than $10,000[4]. Other artists that have shown especial interest in GANs are Mario Klingemann[5], Helena Sarin and Robbie Barrat. Klingemann, sees GANs as "creative collaborators", while Sarin has produced art by training GANs with her paintings or drawings. More recently, Barrat collaborated with Acne studios for creating novel menswear collection, assisted by AI[6]. Some of their works can be seen in Figure 3.3

Furthermore, artists such as Refik Anadol have gained recognition due to their work in live media arts. For instance, in Latent Being[7], an algorithm tracks the movements of visitors and uses them to interact with visuals. In this installation, the visuals are generated by a GAN fed with images of Berlin and its inhabitants. More of his work that uses generative models and AI includes Machine Hallucination, WDCH Dreams, and Archive Dreaming. In these art installations, he displays images generated by sampling from historic photos of New York, the Walt Disney Concert Hall and documents of SALT research.

[3] https://sites.google.com/view/cvcreative/home
[4] https://www.bbc.com/news/technology-45980863
[5] http://quasimondo.com/
[6] https://www.vogue.com/fashion-shows/fall-2020-menswear/acne-studios
[7] https://artsandculture.google.com/partner/lightartspace



(a) (b) (c) (d) (e)

Figure 3.3: Examples of AI as expressive media. (a) A Book from the Sky (2015), Gene Kogan explores the latent space of Chinese handwriting. Source: Gene Kogan. (b) Portrait of Edmond Belamy (2018), sold at an auction for $432,000. Source: Forbes. (c) Memories of Passersby I (2018), Mario Klingemann uses GANs to create portraits based on historical paintings. Source: Art in America. (d) #bashoGAN (2018), model trained on blooming trees and a book of haiku by Helena Sarin. Source: AI Artists. (e) Robbie Barrat and Acne Studios design the Fall 2020 Menswear Collection with AI. Source: XR Goes Pop.

In our work, we also explore the generation of visuals by means of GANs, along with pose detection techniques. We propose an interactive setting, in which audiences can directly guide the image generation process. In this case, a pose estimation algorithm detects people's body joints in real time, which directly control the inputs of a generative model.

3.3 Pose Estimation

Recent developments in computer vision have allowed to progress onto more challenging and descriptive objectives than classic tasks, for instance, pose estimation [47]. This problem is defined as the spatial localization of human body joints within an image. Some of the challenges of this problem includes small and barely visible joints, occlusions, scale variance, lighting, and the importance of capturing context [63]. However, the widespread of Deep Convolutional Networks [36, 59] in the computer vision community has allowed to overcome many of these challenges.

The task of pose estimation, initially conceived for detecting body joints of a single person, has been also extended to the more challenging task of grouping into instances when multiple people are present in an image. Table 3.1 lists datasets for the aforementioned tasks. Moreover, there has been some work that attempts to apply pose estimation and tracking of body keypoints in videos. In this section, we review previous work on the first two tasks: single- and multi-person pose estimation.

3.3.1 Single-Person Pose Estimation

In [63], Toshev and Szegedy introduce one of the first Deep Learning-based pose estimation frameworks. In their work, the authors adapted the AlexNet architecture [36] for regressing a vector $\mathbf{y}$ containing joint coordinates. All joint coordinates are normalized according to a bounding box, which can enclose

Table 3.1: Pose Estimation Datasets

Dataset	# Joints	# Images	Multi-Person	Data type
LSP [30]	14	2K		sports
LSP Extended [31]	14	10K		sports
FLIC [55]	10	5K		feature movies
MPII [4]	16	25K	✓	diverse
COCO [38]	17	330K	✓	diverse

the full image or a bounding box containing a person. The network is trained by using the L_2 distance between predicted and true coordinates as loss function. Furthermore, the initial coordinate estimates can be refined by a cascade of regressors, which have the same network architecture, but only focus on the areas around a joint coordinate. This was proposed for using more detail information, as the initial input image to the network has reduced resolution (220×220). The performance of the framework consists of evaluating detection results according to two metrics: Percentage of Correct Parts (PPC) and Percent of Detected Joints (PDJ), where experiments are performed on the Frames Labeled in Cinema (FLIC) [55] and Leeds Sports (LSP) [31] datasets. Finally, [63] reports a competing or better performance when compared to traditional methods, being the first work demonstrating how Deep Neural Networks can be applied to the pose estimation problem. However, since this framework attempts to learn a direct and highly complex mapping from images to pose vectors, it lacks accuracy in the high-precision region [62].

In an attempt to incorporate priors about the structure of the human body into a part-based model, Tompson et al. [62] proposed a two-stage framework. The first stage consists of a Convolutional Neural Network (CNN) in charge of detecting body joint heat-maps, where each pixel of the heat-map represents the likelihood for key joint locations. Heat-maps are generated from labeled data as a 2D Gaussian with a small variance and mean centered at the ground-truth joint locations. The proposed architecture approximates a sliding window model with multi-resolution and overlapping receptive fields, making the detector translation invariant while capturing information across a variety of scales. This model is optimized over the Mean Squared Error (MSE) between a predicted output and a target heat-map. The second stage is a higher-level spatial model that approximates Markov Random Field (MRF) loopy belief propagation, which is in charge of filtering out false positives and anatomically incorrect predictions (an output heat-map may contain joint locations where the distance between face and shoulders is unusually high). This model constrains joint inter-connectivity and enforces global pose consistency. Both stages are trained independently before fine-tuning in an unified fashion, which improves performance according to the authors. Their approach is tested on extensions of FLIC [55] and LSP datasets [31], where performance on LSP is lower due to the simplicity of the employed spatial model and the larger range of possible poses of this dataset.

As a way of unifying bottom-up and top-down inference without recurring to graphical models, Newell et al. designed an architecture of stacked 'hourglass' modules [45]. Their design is partly based on the work by Tompson et al. [62], as the proposed network outputs heat-maps for each body joint. A hourglass

module is topologically similar to a fully convolutional network [39], as it first decreases and later increases the spatial resolution of the feature representation. However, a hourglass also allows for combining feature information across different scales, and employs skip layers [23] to preserve spatial information at each resolution. Stacking multiple hourglass modules produces different stages of predictions that are further reconsidered by later stages of the network, improving detection performance. According to the authors, the best configuration consists of 8 stacked hourglass modules and applying intermediate supervision. Training is performed by optimizing the MSE between the output of each hourglass and a ground-truth heat-map. The authors carry out experiments on the FLIC [55] and MPII [4] datasets and report a significant improvement over previous methods as measured by the Percentage of Correct Keypoints (PCK) metric.

3.3.2 Multi-Person Pose Estimation

For the more challenging task of detecting keypoints for different instances of people, several approaches that can be grouped into two categories have been proposed. *Top-down* methods consist of identifying firstly individual instances of people before performing pose estimation. On the other hand, *bottom-up* techniques detect body joints first before assigning them to instances.

With a top-down approach, Mask R-CNN [22] demonstrated competitive performance for the task of multi-human pose detection. The network, which is conceived for object instance segmentation, is an extension of the Faster R-CNN detector [52]. The pipeline followed by Mask R-CNN is similar to that of its predecessor: it employs a Region Proposal Network (RPN) followed by a feature extractor for classification and bounding box regression, with the addition of a binary mask. This is achieved by modifying the feature extractor for preserving pixel to pixel alignment and spatial locations. Although it was designed for instance segmentation, the network also presented competitive performance for multi-human pose detection, where instead of finding segmentation masks, the network is asked to provide one-hot binary masks for each body joint. The framework was tested on the COCO dataset [38], achieving a similar or better performance to other pose detection algorithms. The drawback of the method is that it performs at 5 fps, leaving room for improvement.

Papandreou et al. [48] also proposed a top-down model based on Faster R-CNN [52]. After obtaining multiple human instances, pose estimation is performed on each bounding box using a ResNet-101 in a fully convolutional fashion. The latter stage predicts heat-maps and offsets for each body joint, which are later aggregated through a form of Hough voting. More specifically, heat-maps correspond to the probability of occurrence of a keypoint within a disk of radius R, and offsets determine positional vectors from pixels to a certain keypoint. Fusing both outputs results into highly localized activation maps for each keypoint. Furthermore, a variant of non maximum suppression (NMS) is applied after pose estimation (in addition to the standard NMS for person detection). The proposed variant measures keypoint overlap between two pose predictions to filter out false positives. Evaluation of this framework is carried out on the COCO keypoint dataset [38], surpassing performance of Mask R-CNN [22] and previous works.

(a) (b)

Figure 3.4: Multi-person pose estimation examples: (a) PersonLab [47], (b) OpenPose [12].

As a bottom-up method, Papandreou et al. propose PersonLab [47]. In their work, pose estimation for multiple instances is achieved through a deep convolutional backbone based on the ResNet [23] architecture. Firstly, K heat-maps are generated for K keypoints, containing activations for multiple people, along with short-range offset vectors. The aggregation of both outputs allows a refinement of the spatial localization of keypoints, which is done in a similar manner to [48] through Hough voting. Instances are grouped by a greedy decoding algorithm. The algorithm employs an additional output of the network: mid-range offset vectors that are designed to connect pairs of keypoints. Evaluation is performed on the COCO dataset [38], outperforming previous methods. Moreover, the authors affirm the model is very fast due to its simplicity, and it can be implemented for a variety of applications. As a side note, in [46], this method can be used for real-time and web-based applications when employing a MobileNet backbone, albeit less accurate than its ResNet counterpart.

Cao et al. propose the usage of Part Affinity Fields (PAF) to circumvent the challenge of associating body joints with individuals in a bottom-up approach [13, 12]. The authors divide the task of pose estimation into two parts: (a) regressing heat-maps for joint positions and (b) PAFs. PAFs define 2D vector fields, which encode the direction that points from one body joint to another. The architecture proposed in [12] applies refinement stages to predictions of PAFs and heat-maps, allowing for a simultaneous detection and association of body joints. Finally, heat-maps and PAFs are parsed by greedy inference to find keypoints corresponding to each person. This framework is evaluated on the MPII [4] and COCO [38] datasets, and is reported to compete with state-of-the-art methods in performance, while being faster than PersonLab [47].

Since our proposed framework focuses on interaction and content generation, we primarily look for a pose estimation algorithm that is able to run in real-time. Top-down approaches tend to perform slower, since the inference time of top-down approaches heavily depends on the person detector and the number of people per frame. On the other hand, bottom-up approaches usually employ lower resolution images, which decreases detection accuracy albeit with faster inference time. Cao et al. report in [12] comparisons with other person detection methods. For instance, their work runs at over 20 FPS for images with 20 people, while PersonLab [47] performs at around 5 FPS for the same number of people. The main advantage of PersonLab, on the other hand, is its versatility. Depending on the

(a) (b) (c) (d) (e)

Figure 3.5: Human faces generated by different GAN architectures. (a) GAN [18]. (b) DCGAN [51]. (c) ProGAN [32]. (d) StyleGAN [33]. (e) StyleGAN2 [34].

CNN backbone of PersonLab, along with its parameters, it can perform faster. Moreover, the authors provide an implementation for web platforms. Therefore, in a similar fashion to previous work in audience engagement [54] our work employs versions of [47] that are able to perform with real-time constraints.

3.4 Generative Adversarial Networks

An active area of research in machine learning is unsupervised learning, which aims at finding lower level representations of data distributions. This lower level representation may help understanding patterns in data, or encoding its underlying information. Parting from such an encoding and obtaining a sample resembling the original data distribution is what generative models usually strive for.

Deep learning models have shown the capability to map high level representations of certain distributions, such as natural images, to lower level feature-based representations and excel at supervised tasks such as classification [36]. Similar mappings can be applied at an unsupervised level as performed by auto-encoders, which are trained for encoding an image to a compact representation, which can be decoded to reconstruct the image while minimizing a reconstruction loss. This compact feature representation is commonly regarded as the latent space. Models such as Generative Adversarial Networks (GANs) [18] attempt to represent a mapping $G(z)$ from a latent distribution $p_z(z)$ to a target data distribution $p_{\text{data}}(x)$. An input to a generative model can then be a sample vector $z \sim p_z(z)$ from the latent space, and the output sample from the generator distribution p_g should resemble a sample from the target distribution p_{data}.

GANs [18] have become increasingly popular as generative models. They consist of two networks competing against each other: a generator G and a discriminator D. The training objective of D is to determine whether a sample (e.g. a picture of a face) is real or fake, whereas the objective of G is to fool D. This is performed as a two-player minimax game. If G and D are modeled as neural networks, they can both be trained with backpropagation. This original method proposed by Goodfellow et al. [18] set a milestone for generative models and has led to the development models able to generate images with photorealistic quality. Some examples of generated face images by the original work are presented in Figure 3.5(a).

Furthermore, GANs can be conditioned on additional information about the desired output [42]. This information is commonly a class label, but it can also contain an encoding of its description. During training, the condition y is passed to both the generator G and the discriminator D, so that G can generate samples matching y and D learns to accept or reject samples that to not match y. The initial work by Mirza et al. [42], provide examples modeled on the MNIST dataset conditioned on their class labels as one-hot vectors and one on the MIR Flickr dataset [27] conditioned on encodings of user metadata.

Later on, Radford et al. [51] introduced Deep Convolutional GANs (DCGANs), whose architectural constraints demonstrated to be effective for stabilizing the training of GANs employing convolutional layers and improving the quality of generated images upon previous methods. Moreover, their study also explored certain properties of the latent space of GANs. For instance, interpolating vectors in the latent space resulted in smooth transitions in the generated images. Additionally, applying arithmetic operations on vectors reveals structures in the latent space. One example can be that vectors representing a "smiling woman" minus "neutral woman" plus "neutral man" results in a vector representing a "smiling man". This shows that interesting applications can be done when manipulating the latent space of GANs.

As an alternative to the training algorithm proposed in [18], [5] proposes Wasserstein GANs (WGANs). Given the competitive nature of the minimax game played by D and G while training GANs, it is very unstable and not clear at which point to stop it by looking at the loss curves. On the other hand, WGANs show to improve on the stability of training, while providing a better theoretical support and interpretability. This is done by changing the objective function to encourage convergence. This work is further improved in [19], where an additional gradient penalty term is added to the loss function, achieving better performance and stability.

GANs have also proven to perform well for image-to-image translation problems. Image-to-image translation intends to learn a mapping G between an input image X and a target image Y given a set of aligned examples for training. In `pix2pix` [28], Isola et al. employ a conditional GAN architecture for mapping pixels to pixels. In their work, the network is able to synthesize pictures from label maps, reconstruct objects from edge maps and colorizing images. This is possible since in their framework, the network learns a structural loss, that allows to generate images with high fidelity in low and high frequencies, as opposed to previous work. Moreover, some artists have employed the framework for their own creative purposes.

Although `pix2pix` achieved great performance on the image-to-image translation task, its training requires paired data. In [70], a framework is proposed where training does not required paired images. Here, cycle consistency is used. Cycle consistency enforces to learn an inverse mapping $F(G(X)) \approx X$ (and vice versa) by additionally learning a mapping F from Y to X and by applying an additional loss term. This framework performs reasonably well given that is trained on unpaired data, albeit `pix2pix` [28] performs better in most of the tasks.

Progressive GANs [32] later set another milestone by generating photorealistic images. This type of networks introduces the concept of progressive growing. Here, networks are initially asked to generate low resolution images by means of a small architecture, and slowly increase the network complexity and

the output resolution of the images by adding layers. This method achieved a much more stable and faster training.

Brock et al. present in [9] an extensive analysis of large scale training in GANs and achieve a high fidelity conditional GAN. The authors report that performance highly benefits by using deeper and larger architectures. Their work studies the instabilities related to large scale GANs, and determine techniques that can be applied to mitigate them. Moreover, the concept of truncation is introduced, which means that by sampling from a denser part of the latent space, the generated images result in higher quality albeit with reduced variability. The conditioned network presents unprecedented performance on the ImageNet dataset, being able to model 1000 classes.

By introducing spatially-adaptive normalization, Park et al. [49] synthesize photorealistic images according to a given semantic layout. This framework uses a conditional normalization layer called spatially-adaptive normalization, which preserves spatial information from the input semantic layout across the whole network. The achieved model provides control about content and style for generating images.

Further work by Karras et al. [33] built up on Progressive GANs by borrowing from style transfer methods. This novel network architecture, named StyleGAN, has been able to learn an improved separation of high level features or *styles*. The generator is capable of controlling the style of an image at each layer based on the latent code. Although the previous work on GANs has achieved unprecedented results, they still presented a number of issues. One of them being an entangled latent space representation. As an example, interpolation from two latent vectors may translate to abrupt changes between two features in the generated images. The authors work around this problem by making the network learn an embedding from the input latent space to an intermediate representation, which must follow the probability density of the training data. This intermediate space allows for a more disentangled representation. Finally, one of the most interesting characteristics of this method is the scalability of the network, which allows synthesizing high resolution images with improved photorealistic quality albeit with the presence of some artifacts.

More recently, Karras et al. were able to further improve StyleGAN and remove existing artifacts. The authors analyzed and improved over some design flaws in the previous architecture, while revisiting the effects of progressive growing [32]. In their work, they redesigned normalization steps carried out in the generator, regularized it to improve mapping from latent vectors to images, and removed progressive growing while maintaining the training scheme of shifting from low to high resolution images. Up to date, the revisited StyleGAN presents the highest photorealistic quality in unconditioned models. Some image samples are shown in Figure 3.5(e).

A number of studies [68, 29, 21] have attempted to identify interpretable directions in the latent space of GANs such as StyleGAN [33, 33] and BigGAN [9]. Yang et al. [68] use *off-the-shelf* image classifiers to learn a decision boundary in the latent space for a give attribute of synthesized images. Jahanian et al. [29] attempt to find a path in the latent space to transform a generated image to achieve camera motion or color changes. The paths in [29] are learned in a self-supervised manner by minimizing the distance

Figure 3.6: Face animation produced by the method described in [58]. Taken from the author.

between synthesized images and edited versions. Furthermore, [21] proposes an unsupervised technique to find interpretable controls for image synthesis, such as viewpoint, color or lighting. Harkonen et al. show that such edits are possible by using Principal Component Analysis on the StyleGAN intermediate representation and on BigGAN neural activations.

In this work, we show how audiences can interact and generate content by controlling the latent space of a GAN. We find that the most suitable network is StyleGAN2 [34]. StyleGAN2 is not only capable of producing high quality and realistic images in different resolutions, but it also presents other features that we can employ for our setting. For instance, the network learns a disentangled representation of image attributes, which we can control in an intepretable manner [21]. Moreover, the style mixing property of StyleGAN2 gives room for more interactivity. A possible use case consists of people controlling a subset of generative layers with their movements, while other inputs such as audio operate on the remaining layers.

3.5 Image Animation

Image animation typically uses two inputs: a source image and a driving video. In this case, the task consists of applying the motion from the driving video to an object in the source image [58]. A common example is animating the expressions of a face in a photography by using a video of a different person, as shown in Figure 3.6. Traditional approaches are tailored to specific domains, such as animating faces or silhouettes and require a strong prior of the animated object [11, 61]. However, recent methods based on deep learning techniques produce better results and generalize better to other domains [66, 6, 67, 57].

In [11], Cao et al. present an approach for estimating 2D keypoints of faces with a single video input. Their method is capable of regressing 2D facial landmarks along with the 3D facial shape from 2D video frames. The method uses the estimated 2D landmarks to adapt the camera matrix and the user identity to match the facial expressions of the current user. The proposed algorithm applies the regression and adaptation steps in an alternating manner, creating a feedback loop. In their work, the authors show how the final 3D shape can be later used to animate a 3D digital avatar in real time. The main limitation

of this method is that it is designed for animating faces, and it is non trivial to apply this approach to other domains.

Thies et al. propose a markerless approach for face animation [61]. As opposed to previous methods, their work transfers facial motion to monocular RGB videos instead of 3D avatars. The method first reconstructs the shape identity of an actor in the source video. This allows to resolve geometric ambiguities offline. Later at runtime, the algorithm tracks the expressions of both the source and driving videos by a dense analysis-by-synthesis approach based on a statistical facial prior. To transfer expressions from driving to source video in real-time, the authors employ transfer functions that apply deformation transfer. Finally, the algorithm performs image synthesis by re-rendering the face in the source video with the transferred expression coefficients. Compared to previous methods, this approach is not limited to animating digital avatars. However, it requires a training step on the target or source video that is to be animated.

More recently, Wang et al. present a deep learning technique [66] that incorporates a generative adversarial learning framework [18]. Their method learns a conditional generative adversarial model [42] given paired input and output videos, and builds up on the image-to-image translation method of Isola et al. [28]. Moreover, the authors introduce a spatio-temporal learning objective that results in a model able to learn photorealistic, temporally coherent videos. Although this framework generalizes well for the image animation task, it requires hours of videos of a person labelled with semantic information and the model must be retrained for each individual.

Similarly, Bansal et al. introduce a method [6] that uses spatio-temporal cues along with conditional generative adversarial networks [18, 42]. Their unsupervised approach enables content translation while preserving style, and can be applied to face translation. The authors emphasize the importance of temporal information for constraining the optimization problem and improving convergence. Moreover, the spatial and temporal constraints allows preserving style information. However, similar to [66], this approach requires large amounts of formation from individuals in order to perform face image animation.

In [67], Wiles et al. are able to control the pose expression of a source face given another face modality, such as pose or audio. Their work employs an embedding network and a driving network. The embedding network learns a face representation, whereas the driving network learns a mapping from the face representation to a generated frame via a driving vector. Moreover, the proposed framework is trained on a self supervised fashion. Namely, during training, the algorithm selects a random frame from a video sequence as a source image, and employs the other frames as the driving video. As the frames come from the same video, all driving and generated frames should match. Nonetheless, at test time, source and driving frames can be different.

Siarohin et al. similarly employ a reference pose to represent a source object [58]. However, their approach does not require an explicit reference pose and leads to a simpler optimization and better image quality. In their self-supervised framework, the authors employ sparse keypoint trajectories together with local affine for animating arbitrary objects. The technique additionally models occlusions to indicate the synthesis network the image regions that can be generated by warping the source image and the occluded

areas that should be inferred from the context. Finally, the method outperforms state-of-the art image animation approaches and is able to handle high-resolution datasets.

In our work, we present a module based on [58] that transfers facial expressions from an audience guest or performer to a desired source face. The module can perform on two modalities: a static and stable camera pointing at an user, and receiving video frames from a larger audience. The first approach requires minimum additions to the framework presented in [58], where users simply select a source face they wish to animate, and can change it in real time or enable artistic effects. The second modality uses a face detector and face matching systems to detect and track a guest's or performer's face. Later, the algorithm transfers facial motion of the detected face to a target image. Similarly, users can enable artistic effects and apply them to the generated animated image.

Chapter 4

Proposed Method

In this chapter we describe a number of frameworks where audiences can create artistic content and interact with it while attending live events. In our first scenario, we measure the movements of the public and use them to modify properties of 3D visuals. For instance, the number of displayed objects grows when more people are present, and decreases otherwise. This is achieved by detecting peoples poses and using them as inputs to the rendering algorithm of the 3D visuals.

Furthermore, we also explore how the public can create visuals by means of generative algorithms. In this case, an AI algorithm is used to translate people's movements into visuals. As an example, if we show the AI algorithm thousands of images of landscapes, it will learn to produce more unseen landscapes images. Moreover, if we use information from the public in an event venue, such as people's poses as the parameters of the AI, the algorithm can 'paint' landscapes while people are moving. On the other hand, if the venue is empty, the AI will only show a static image. For this scenario, we employ a pose detection algorithm whose output drives the latent space of a GAN.

Since generating content can also benefit from performers in an event, we propose the addition of audio inputs to our generative models. Here, information such as pitch and energy can act as additional controllers for generating content.

We additionally consider the case where an event guest interacts with a digital character. Here, the user's expressions control the character's face movements. Moreover, we include the possibility to add other effects to the animated face by using face landmark detection.

This chapter is organized as follows: Section 4.1 presents the technique our framework uses for real time pose estimation. Section 4.2 illustrates our method for controlling 3D visual elements via shaders. Later, Section 4.3 proposes the use of GANs for generating and interacting with 2D visuals. Finally, Section 4.4 describes the method we use for face animation along with other effects.

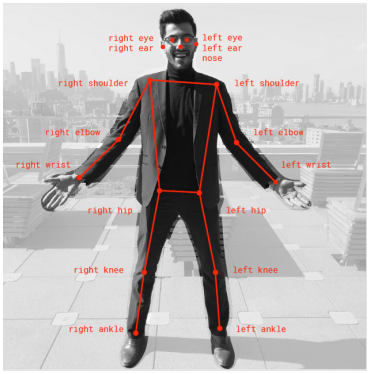
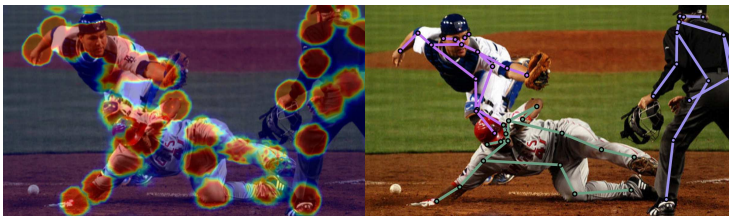


Figure 4.1: Seventeen keypoints detected by PersonLab [47]. Source: [46].



(a) (b)

Figure 4.2: (a) Heat-maps predicted by the PersonLab network [47] and (b) instance keypoints after refinement step. Images from [47].

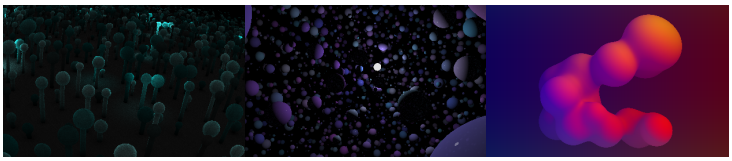
4.1 Real-time Multi-person Pose Estimation

The objective of a pose estimation algorithm, as mentioned in the previous chapter, is to localize K 2D body joints of people within an input color image of size $w \times h$, and group the joints into person instances. The algorithm detects *normalized* 2-D positions $y_{j,k} \in \mathbb{R}^2$ of the k-th keypoint of the j-th person instance, with $j = 1, ..., M$ and $k = 1, ..., K$. In this work, we employ a modified version of PersonLab [47], which retrieves $K = 17$ keypoints for each person, as seen in Figure 4.1. The network takes as input a color image and predicts heat-maps for each body joint (Figure 4.2a). The algorithm later refines the keypoint localization through short-range offset vectors, and groups them into instances through a greedy decoding process. The detected keypoints for each instance can be seen in Figure 4.2b.

In this work, the PersonLab implementation uses a MobileNet backbone [25, 47, 54]. Although the MobileNet backbone results in lower accuracy than its ResNet counterpart [46], it performs faster and adapts better to real time applications. Moreover, the output stride of the network determines how much the algorithm scales down the output relative to the input image size. With a higher output stride, the accuracy decreases. Therefore, since our work does not require a high accuracy, we find that an output stride of 16 performs well.

Since our work deals mostly with images under low-light conditions, we apply a pre-processing step before performing keypoint detection [54]. First, we crop the color image to a region of interest (ROI), where it is most likely to contain people from the public, and later apply gamma correction along with contrast stretching (Figure 4.3). Later, the image is fed to the PersonLab network [47] to execute the

Figure 4.3: Frame after pre-processing step. Here, we crop the frame to region of interest (ROI) where it is most likely to contain people. Our system divides the ROI into different sub-regions to which we apply contrast stretching and gamma correction. Source: [54]



(a) (b) (c)

Figure 4.4: Interactive 3D objects. Source: (a-b) Shadertoy. (c) Cineshader.

complete detection pipeline. By having the localized keypoints within a video stream, people's actions can be used to generate and interact with artistic content in other stages of our framework.

4.2 Interaction with 3D Graphics

In this stage, a system automatically updates properties of 3D graphics according to information measured from the public. Some examples of 3D objects that can be controlled are shown in Figure 4.4. In order to reflect the current state of the public, our method maps the number of present people to the number of displayed objects. Moreover, people moving within the field of view of the camera can modify other variables such as color, illumination, and movement speed.

A possible scenario, is that by moving the head or an arm, a person controls the color palette used for rendering the objects. The proposed algorithm maps the in-screen position of the keypoint to a

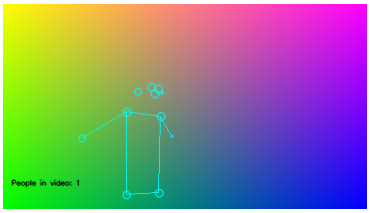


Figure 4.5: A selected body joint position of an user controls the color palette used for rendering 3D objects. For example, the user can change the color from green to orange by moving their arm over their head.

pre-defined color palette. For instance, by moving the wrist, an user selects the RGB values that the rendering algorithm employs when displaying the objects, as shown in Figure 4.5. In a similar fashion, other body joints can drive the movement speed, shapes and sizes of objects.

To better reflect the current state of the event venue, our algorithm maps the number of people to the objects' sizes and movement. This is done by computing the ratio between the current number of people and the capacity of the venue. As an example, when the number of people increases, the size of the spheres in Figure 4.4a grows, and the objects move faster. Moreover, our method renders each 3D scene by using shaders, which compute simultaneously all pixel colors according to the inputs. In our case, the inputs to the shader specify the color palettes, illumination and movement of the objects.

Typically, the detected keypoint locations are noisy in time, changing in a non smooth manner for each subsequent frame. For instance, even if a person is not moving and the camera is stationary, there are slight variations in the keypoint locations from frame to frame. This produces a jitter effect in the rendered graphics. Therefore, we apply double exponential smoothing [37] to each input parameter of the shaders to obtain smoother visual transitions. For a parameter p_t at a given time t, we compute two smoothing statistics:

$$Sp_t = \beta p_t + (1-\beta) Sp_{t-1} \tag{4.1}$$

$$Sp_t^{[2]} = \beta Sp_t + (1-\beta) Sp_{t-1}^{[2]}, \tag{4.2}$$

where $\beta \in (0, 1)$ is the smoothing parameter. We smooth p_{t+1} as:

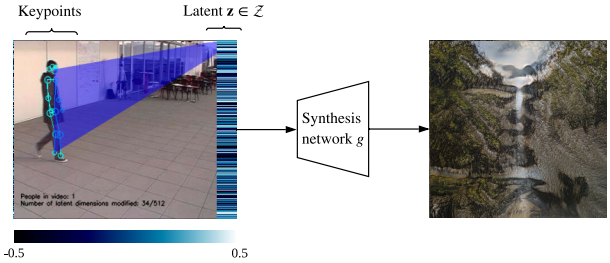
$$p_{t+1} = 2Sp_t - Sp_t^{[2]}. \tag{4.3}$$

We qualitatively hand pick a suitable β for each shader parameter that minimizes jitter effects in the visuals.

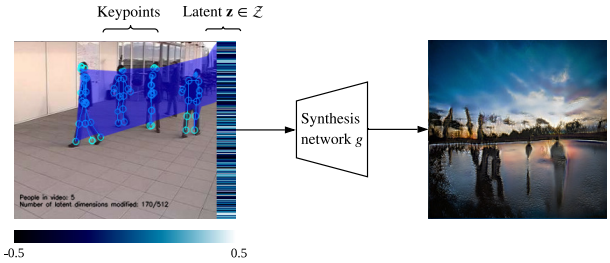
4.3 Simultaneous Interaction and Content Generation

Audiences can generate visual content in real time by standing or moving in front of a camera. Figure 4.6 illustrates the overall pipeline of this framework. First, a pose detection algorithm localizes the keypoints of a person or multiple people. Later, the keypoints' dimensions drive the latent space $\mathbf{z} \in \mathcal{Z}$ of a GAN, which produces images in real time. As more people join and move, they modify the latent space along with the generated pictures.

In our experiments, we employ a version of PersonLab [47] using a MobileNet backbone [25] for pose detection. Here, each 2-D location $y_{j,k} \in \mathbb{R}^2$ of a body joint steers two dimensions of the latent space $\mathcal{Z}$, which has $d = 512$ dimensions (in the case of StyleGAN2 [34]). Initially, we sample a random variable $\mathbf{z}$ from $\mathcal{Z}$ and the *normalized* keypoint locations later modify $\mathbf{z}$. For instance, a single person is able to



(a)



(b)

Figure 4.6: Users' keypoints modify a random variable z of 512 dimensions sampled from the latent space $\mathcal{Z}$ of a GAN. In this way, users' actions within the field of view of a camera modify the content generated by the GAN. (a) One person controls a part of the latent dimensions. (b) More people drive the latent space of the GAN, resulting in more variability of the generated content.

control up to $2K$ latent dimensions (Figure 4.6a), where K is the number of keypoints per instance. If there are M people detected in the current frame, the keypoints detected jointly control $2MK$ dimensions of the latent space $\mathbf{z}$, as shown in Figure 4.6b. However, the number of people who can simultaneously modify $\mathbf{z}$ is limited by the latent space's size d. Therefore, the number of people that can participate in the image generation process is limited by:

$$M_{\max} = \left\lceil \frac{d}{2K} \right\rceil . \tag{4.4}$$

We normalize the keypoints to $[-0.5, 0.5]$ according to the image dimensions rather than normalizing to each instance dimension. We do this so that people closer to the camera have a stronger influence in the generated images than those further away. Moreover, bounding the keypoint's scale bounds the latent $\mathbf{z}$ to a well sampled zone closer to the mode of $\mathcal{Z}$, which maintains the quality of generated images [9].

In principle, this framework can operate with any GAN architecture. In our work, we use an implementation of StyleGAN [34]. In the StyleGAN architecture, a Multiayer Perceptron (MLP) learns a mapping from the latent vector $\mathbf{z}$ to an intermediate representation $\mathbf{w} \in \mathcal{W}$, which controls the synthesized image. In a basic setting, a single vector $\mathbf{w}$ is fed through every layer of the synthesis network g. In their work [33], the authors show that by applying a different samples $\mathbf{w}$ to distinct layers enables "style mixing," which combines features of various abstraction levels. This feature opens more possibilities for steering the styles of the generated images by using other inputs.

Since keypoint locations are noisy in time, the generated sequence of images change in a non smooth manner. To mitigate this effect, we propose detecting keypoints and computing the intermediate lantent representation $\mathbf{w}$ every N frames. We can then perform linear interpolation between each subsequent $\mathbf{w}$, obtaining N interpolations. Later, we generate images from each interpolated vector $\mathbf{w}$. More specifically at frame $n = tN$, where $t \in \mathbb{Z}$, we map a set of keypoints $\mathbf{y}_t$ to the latent representation $\mathbf{w}_t$. Later we compute N interpolations $\mathbf{w}_l$ between the latent vectors as:

$$\mathbf{w}_l = \text{lerp}(\mathbf{w}_{t-1}, \mathbf{w}_t, l/N), \tag{4.5}$$

where lerp denotes linear interpolation and $l = 1, ..., N$. Next, we generate an image for each interpolation $\mathbf{w}_l$, resulting in a set of N synthesized images for the next N frames. Although this results in a shift in temporal correspondence between detections and generated images, we show in Section 5.2 that this technique not only improves the smoothness of the generated image sequence, but also boosts runtime performance.

4.3.1 Steering Principal Directions

One of the caveats of the aforementioned framework is that only modifying a low number of dimensions of $\mathcal{Z}$ usually results in low variation in the generated images. This is the case when there are few people present in our scenario, and a low number of keypoints drives the latent vector $\mathbf{z}$ (Figure 4.6a). This can be improved by finding useful directions in layer activations [21].

Since the generator architecture of StyleGAN [33, 34] maps the input samples from the latent space $\mathcal{Z}$ to an intermediate latent space $\mathcal{W}$, it is possible to find interpretable directions by computing PCA over $\mathcal{W}$ [21]. By obtaining a low-rank basis $\mathbf{V}$, we can edit an image defined by a latent vector $\mathbf{w} \in \mathcal{W}$ by varying PCA coordinates $\mathbf{x}$:

$$\mathbf{w}' = \mathbf{w} + \mathbf{V}\mathbf{x} \tag{4.6}$$

$$I' = g(\mathbf{w}'), \tag{4.7}$$

where g is the synthesis network of StyleGAN and each entry of $\mathbf{x}$ is a control parameter, for instance, the output of a pose estimation algorithm. This procedure is illustrated in Figure 4.7, where a single person controls PCA coordinates of $\mathcal{W}$ and the resulting latent vector $\mathbf{w}$ drives the generation of an image with the synthesis network g. Moreover, more people can further influence the resulting image if each user drives the coordinates of a different layer of g.

4.3.2 Mixing Styles

Since the StyleGAN architecture allows for mixing styles by using different samples $\mathbf{w}$ from the intermediate latent space $\mathcal{W}$ and applying them to distinct layers, in principle we could direct the image generation

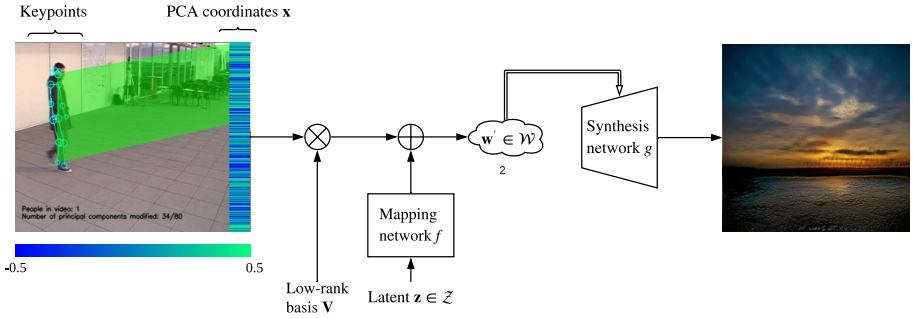
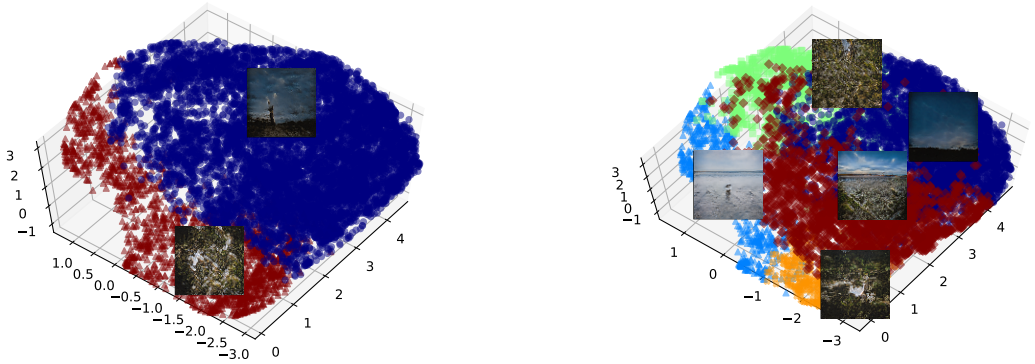


Figure 4.7: Users' keypoints steer PCA coordinates of StyleGAN's intermediate latent space $\mathcal{W}$. This results in a higher control of the output image features when there are fewer people participating. In this example, the PCA coordinate vector $\mathbf{x}$ has 80 dimensions.



(a) (b)

Figure 4.8: Clusters of $\mathcal{W}$, the displayed images depict cluster centers. The clusters are found by fitting a K-means algorithm over 10000 samples of $\mathcal{W}$ with $K = 2$ (a) and $K = 5$ (b). Path between cluster centers define changes in features of the generated images. We use UMAP [41] for reducing dimensions and visualizing $\mathcal{W}$.

process towards a desired appearance. Some methods that allow to continuously control a trained GAN over its latent space resort to some form of supervision. For instance, by using image classifiers to find latent space transformations or learning from specified transformations [21, 29, 68].

On the other hand, an unsupervised scheme has the potential to allow for more flexibility in our framework. If we can collect a set of representative styles, we could use any style from the set to direct a latent variable $\mathbf{w}$ towards a desired direction on a pre-defined number of layers of g. Therefore, any sample from the set of styles will determine the visual characteristics of the generated images.

In [33], the authors demonstrate that thanks to the mapping network f, the intermediate latent space $\mathcal{W}$ is linearly separable for different features. Recent work has shown that by fitting a hyperplane that separates two features yields directions or paths that explore the selected features [68, 29]. However, a clustering algorithm is able to find representative samples that encode distinct features and are distant from each other. For instance, defining a path between the two clusters shown in Figure 4.8a results in

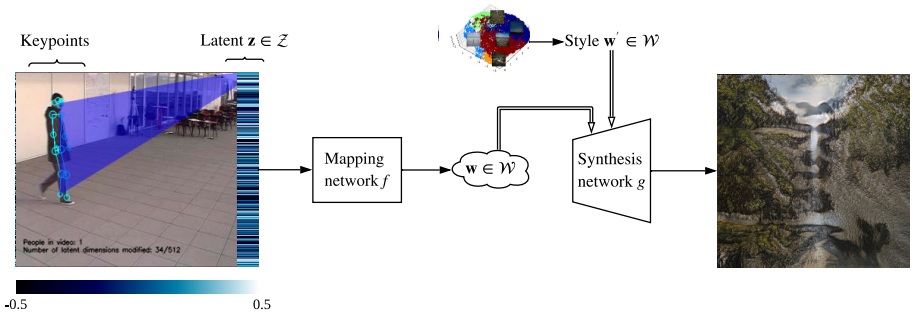


Figure 4.9: Users' keypoints control the lower layers of the synthesis network g, while a pre-defined trajectory of styles drive the higher layers of g.

a significant change of features from one image to another. We can extend the number of possible paths by finding more cluster centers, as depicted in Figure 4.8b.

A possible scenario would be to pre-define a trajectory of styles to mix with the inputs of the pose estimation network, as depicted in Figure 4.9. In the following section, we consider using audio features, to drive a style trajectory and perform style mixing.

4.3.3 Audio Styles

In a concert setting, the generated visuals can use both the information from the audience and the performers. More specifically, applying audio analysis techniques to the music being played by the performers yields features that we can apply to further enhance the generated content. For instance, measuring energy changes in the audio signal can determine the pace at which some image features change.

Energy changes are measured by computing the spectrogram over a window of samples of the audio signal. Here, we use the Mel Scale, according to [60], to have a representation similar to human perception of frequency bands. For each window, we compute the mean energy E_m across all Mel bands and compare it to the mean energy of the previous window E_{m-1}. With this information, sudden increases in energy $E_m - E_{m-1} > 0$ are measured to adjust the speed at which some image features are modified. For instance, larger energy changes result in traversing faster along a pre-defined 'style' trajectory in the latent space $\mathcal{W}$. In this case, we sample a trajectory over $\mathcal{W}$ as described in the previous section. Moreover, our pipeline assigns the style $\mathbf{w}_E$ from the trajectory to higher layers, while the detected poses control the lower layers of the generator g, as shown in Figure 4.9.

Another option is to determine the pitch for driving the style over a subset of layers. We do this by detecting the pitch F_0 of the current window of the signal and use it to query a style $\mathbf{w}_{F_0}$ from a pre-sampled set of styles. We employ the method described in [16] and implemented in [10] for estimating the pitch. Later, we mix the queried style $\mathbf{w}_{F_0}$ with latent variables controlled by other sources, for instance, poses and audio energy changes. An example of a scheme employing audio and pose estimation inputs is presented in Figure 4.10.

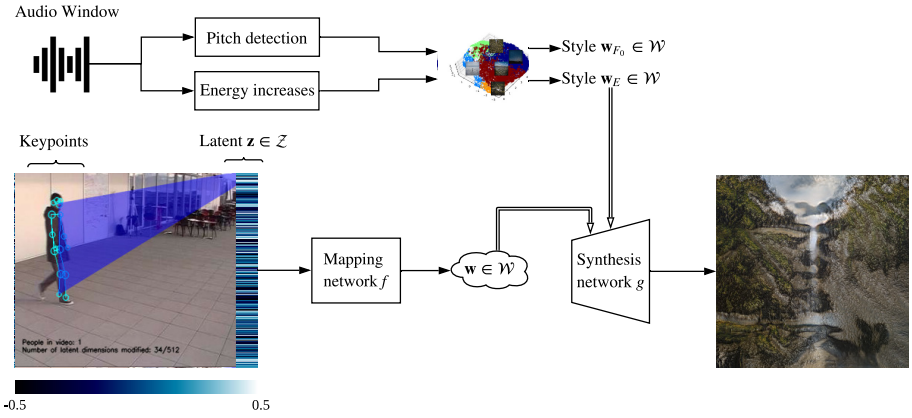


Figure 4.10: Audio and pose inputs drive the latent space of StyleGAN. Here, the pose keypoints control the lower layers, whereas audio energy changes and pitch drive the mid and higher layers.

When using the audio option, synthesizing N images every N frames is not desirable, since would like that changes in each generated frame relate in real time with the audio input. We therefore perform the mapping step to obtain N intermediate vectors $\mathbf{w}$, which we later use to mix with styles influenced by the audio input. Since our synthesis network must generate images each frame according to the mixed styles, the overall run-time increases. Nonetheless, our system is able to perform above 15 frames per second, as we illustrate in Section 5.2.

4.4 Face Animation

The task of image animation consists of producing a video sequence by combining the appearance extracted from a *source image* with motion patterns inferred from a *driving video* [57, 58] (Figure 3.6). The modeled developed in [58] is trained in a self-supervised manner. This means, that the data used for training does not require explicit labels. In this approach, the model is trained so that it learns to synthesize training videos by combining a single frame and a learned latent representation of the motion in the video. The learned motion depends on the data domain. For instance, the motion representation can encode facial expressions, body movements, or other object movements. At evaluation time, the algorithm is able to animate a source image given the extracted motion of a driving video.

The method described in [58] incorporates a motion estimation module, and an image generation module. The motion estimation part has the task of regressing a dense motion field from a color frame $\mathbf{D} \in \mathbb{R}^{3\times H\times W}$ of the driving video $\mathcal{D}$ to the source frame $\mathbf{S} \in \mathbb{R}^{2\times H\times W}$. The network uses the dense motion field $\hat{\mathcal{T}}_{\mathbf{S}\leftarrow\mathbf{D}}$ to align feature maps computed from $\mathbf{S}$ with the object pose in $\mathbf{D}$. Furthermore, the motion estimator additionally outputs an occlusion mask $\hat{\mathcal{O}}_{\mathbf{S}\leftarrow\mathbf{D}}$, which indicates the synthesis network which image parts of the frame $\mathbf{D}$ should be reconstructed according to the dense motion field, and which image parts should be inferred from the context.

We incorporate the image animation method proposed by Siarohin et al. [58] into our framework so that event guests can animate a face with their facial expressions, as depicted in the top two rows of

Figure 4.11: A guest (top row) animates the image of a bust (second row) with the method proposed in [58]. A facial landmark recognition algorithm [35] detects keypoints around the eyes and nose of the animation (third row), which can be used so that an user can optionally apply effects for distorting the eyes in the face animation (bottom row).

Figure 4.11. Moreover, we also propose applying other effects around the eyes of the face animation. This is done by detecting the eyes' position with Dlib's 5-point face landmark detector [35]. The detected keypoints allow defining a region around the eyes, on which we apply different transformations, as shown in the two bottom rows of Figure 4.11.

Our algorithm performs the distortions around the eyes randomly. The region to which we apply the transformations has a size of $H_e \times W$, where W is the width of the animation frame and $H_e = \alpha_{\text{eyes}} H$ is a fraction of the height H of the animation frame and $\alpha_{\text{eyes}} \sim U(a, b)$ with $a, b \in (0, 1)$. The region is centered around the mean eye vertical position, and we randomly apply spatial shifts along color channels (Figure 4.11 bottom right) or horizontal shifts across several smaller regions (Figure 4.11 bottom left).

We additionally propose a module that detects a face from the audience and tracks it to animate a face in a source image. In this case, we use Dlib's [35] face detector, and use a greedy search for tracking the face in subsequent frames. We attempt to match face bounding boxes by maximizing intersection over union. We apply double exponential smoothing to the bounding box to account for detection and tracking noise. If a face is lost, we continue tracking the next available face. Later, we crop the face bounding box and use it to animate the source face.

Chapter 5

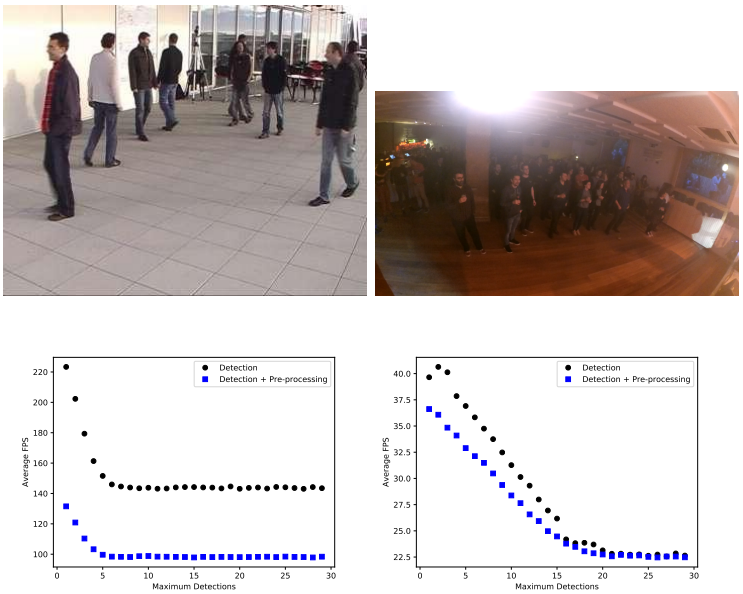
Evaluation

We evaluate the performance of our framework by testing the required run-time for its different stages. In Section 5.1, we study how selecting the maximum number of expected person instances affects the speed of the pose estimation module. Section 5.2 analyzes the processing speed of our framework for synthesizing images with GANs. More specifically, we evaluate how choosing different intervals N for performing pose detection and synthesizing images affects the processing time of the whole pipeline. Later, in Section 5.3 we test how diverse are the synthesized images according to different scales applied to the latent vector $\mathbf{z}$. Finally, we present some qualitative results in Section 5.4.

For all of our tests, we use a NVIDIA 2070 RTX GPU. We set the pose detection module with an output stride of 16. Moreover, we modify the original StyleGAN2 architecture [34] to produce images with a resolution of 256×256 with a synthesis network of 7 layers. We train the modified architecture of StyleGAN2 on the Monet2photo dataset [70], which contains primarily landscape images. Nonetheless, our qualitative results present a mix of sequences generated using the original architecture of 18 layers trained on the FFHQ dataset [33] along with the modified 7-layers version trained on the Monet2photo [70] dataset.

5.1 Run-time Analysis of Pose Estimation

We performed a number of experiments for measuring the run-time of the keypoint detector as a function of the maximum expected number of people. In our experiments, we measure the average frames per second (FPS) of processing the audience frames according to the maximum expected number of people to be detected by the pose estimation algorithm. Figure 5.1 shows the results on two videos. As can be seen, the lighter implementation of PersonLab [47], PoseNet [46], performs in real time for different settings with varying numbers of people. Although the pre-processing step adds some overhead, the detection still performs in real-time and the difference decreases with a higher number of people. Moreover, the pose detection algorithm performs faster when we set the expected number of people to lower numbers, since the grouping stage of the algorithm employs a greedy approach [47]. On the other hand, setting



(a) (b)

Figure 5.1: First row: frame from test videos. Second row: corresponding results. (a) Scene with up to 8 people walking and a resolution of 360×288. (b) Video sequence with a group larger than 20 people and a resolution of 1920×1080. The pre-processing step presents an average difference of 49.9 FPS for (a) and 1.79 FPS for (b).

the expected number of detections to a number above to the actual maximum people does not affect the final performance of the algorithm.

With this in mind, we can define the best parameter for a possible scenario if we have prior knowledge on the average people that are attending an event and we want to put constraints on the performance and how many people can be detected. For instance, if we employ a GAN to produce visuals, as described in Section 4.3, we know that the maximum number of people that can participate in the image generation process is limited by Equation 4.4. Furthermore, we observed that the overhead for displaying the shaders is minimal when using a dedicated hardware (GPU).

5.2 Run-time Analysis of Interaction and Content Generation

This section shows results after measuring the run-time of our framework for generating 2D images according to people's poses. We measure an average estimate of FPS for different detection intervals N as explained in Section 4.3. In this test, we determine the duration from reading and processing frames to generating images, excluding the display step. Later, we divide the duration by the number of frames to calculate the average frames per second (FPS). We perform tests for $N = 1, .., 30$ detection intervals with three video sequences set in different scenarios and varying numbers of people. As seen in Figure 5.2, higher detection intervals N improve run-time performance. However, the overall speed depends on the video sequence. For instance, the video labeled as "terrace" has the lowest resolution (360×288), whereas the other two have a resolution of 1920×1080. Nonetheless, we include a resize step for the latter videos,

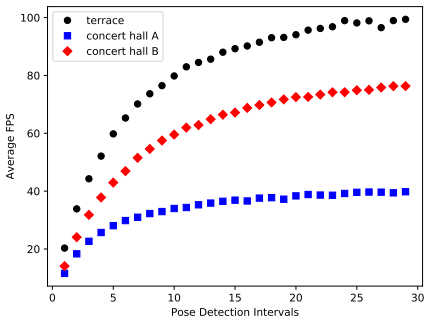
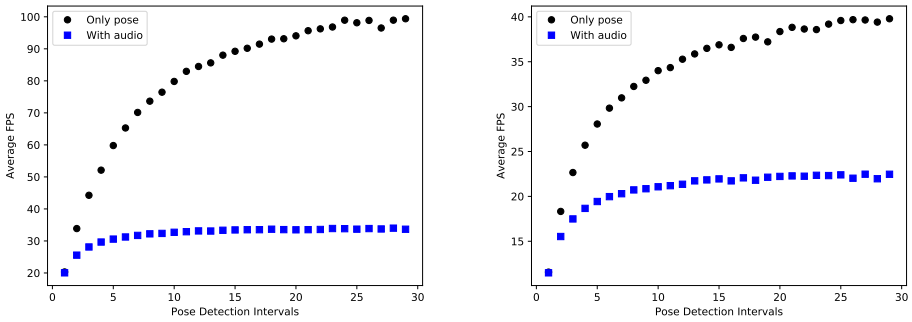


Figure 5.2: Run-time analysis for different detection intervals N. A larger N results in higher FPS performance. However, the overall processing speed depends on the type of video and its resolution.



(a) (b)

Figure 5.3: Run-time results for different detection intervals N. (a) Video with 360×288 resolution and with up to 8 people. (b) Video with 1920×1080 resolution and a group larger than 20 people. After around $N = 5$, selecting a larger detection interval N does not improve FPS, albeit the system still performs above 15 FPS.

reducing the resolution by up to 50% before performing detection.

On the other hand, we note that setting a higher N produces some lag, since it results in a delayed temporal correspondence between detections and generated images. Therefore, users should select the detection interval N according to the desired performance.

We additionally carry out tests to measure how including the audio option affects the execution time of our pipeline. As seen in Figure 5.3, including the audio processing module decreases the average FPS considerably. This is because the generator network must synthesize images every frame instead of every N frames. As explained in subsection 4.3.3, we do this to allow audio features to influence the generated images in real time. Nonetheless, depending on the selected detection interval N, the system is able to perform at above 15 FPS.

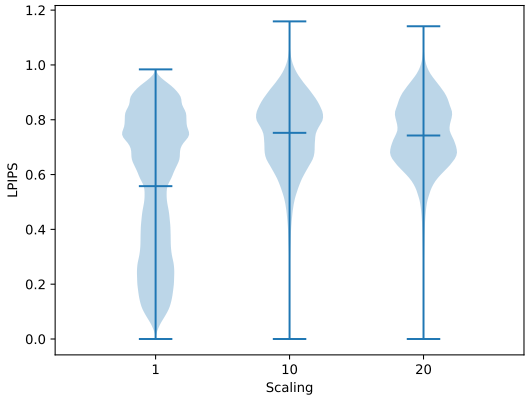


Figure 5.4: LPIPS distribution for video sequences generated by applying different scaling factors to the latent vector **z**.

Table 5.1: Frame Diversity and Quality Measures

Scale	Average LPIPS	FID
$\alpha_{\text{scale}} = 1$	0.558	121.391
$\alpha_{\text{scale}} = 10$	**0.752**	**68.629**
$\alpha_{\text{scale}} = 20$	0.742	69.197

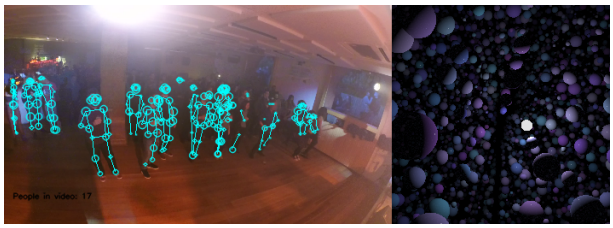
5.3 Generated Frame's Diversity Study

Are the synthesized images diverse enough? In this experiment we use the LPIPS metric [69] to measure image diversity within a video sequence and compare it with the Fréchet Inception Distance (FID score) [24] for the generated frames. The LPIPS metric measures the perceptual similarity of two images. Moreover, the FID score measures the quality of the generated image samples by comparing their distribution to the distribution of the real image samples. Therefore, we measure how similar are different possible combinations of images pairs of a video sequence. We perform this experiment with different **z** latent scales ($\alpha_{\text{scale}}z$, $\alpha_{\text{scale}} \in \{1, 10, 20\}$). In this case, the latent vector **z** represents the normalized keypoints obtained from the pose estimation algorithm. The resulting distribution can be observed in Figure 5.4, where according to the LPIPS metric, a higher scale α_{scale} results in frames with a higher diversity than using only the normalized keypoints **z**. Moreover, we report the average LPIPS metric across all possible generated frame pairs in Table 5.1 along with the estimated FID score. In this case, we observe that scaling also improves the quality of generated images, but setting α_{scale} too high results in decreased quality.

5.4 Qualitative Results

We present some results of our developed framework for the use case considered in Section 2.1, where audiences drive 3D rendered visual objects. Figure 5.5a illustrates how a single person controls visual

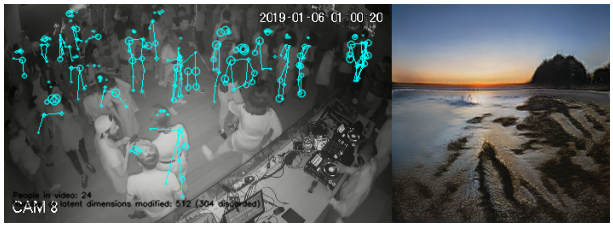
(a)



(b)

Figure 5.5: (a) The animation moves according to a dancer's movements. (b) As more people come into the concert hall and move, the projector displays more objects and the animation moves faster.

(a)



(b)

Figure 5.6: (a) Event guests can move to produce a smile (b) or a sunset.

elements by moving, and Figure 5.5b shows a group of event guests driving visual effects. In the first case (Figure 5.5a), the user controls the colour with the head position, the size of the objects depends on the right wrist position, and the objects' size changes according to the person's overall movement. Moreover, the audience's size is the main factor for controlling the movement speed of the elements in second case (Figure 5.5b). We also consider that this scheme aligns well with the use case about the music festival app (Section 2.2), since the platform could send the shader information to users instead of a video stream. We refer the reader to our drive folder which contains videos with more examples[1].

Furthermore, Figure 5.6 shows results of the scenario presented in Section 2.1 where event guests can generate 2D visuals. For instance, at some point during the event, a dancing group is able to make the GAN produce a smiling face (Figure 5.6a) when trained on the FFHQ dataset [33] or a sunset (Figure 5.6b)

[1]Folder with shader examples: https://drive.google.com/open?id=1VnYMVvtcSX8W3SrY4LTbLSsUWVOcZxkT

Figure 5.7: A guest animates the image of a virtual character with his own expressions. The top row presents frames of the driving video and the bottom row displays frames of the generated animation.

Figure 5.8: Example of a single performer generating 2D artistic content. Here, the GAN was trained on landscape images (monet2photo dataset [70]).

if the network was trained on the Monet2photo dataset [70]. We also provide video examples showing how an increasing number of people generates different face[2] and landscape[3] images. Moreover, an example of the scheme that employs audio and pose estimation inputs, as mentioned in subsection 4.3.3, can be found here[4].

Section 2.3 covers the scenario of image animation, which we include in our framework using the method described in Section 4.4. Figure 5.7 shows an additional example of image animation. Moreover, some video examples can be found in our drive folder[5].

We additionally present some results for the case of a single performer or a small group of performers that we mentioned in Section 2.4. For instance, a dancer is able to control 3D shapes with her movements, as seen in Figure 2a. Moreover, we evaluate the case of 2D image generation by applying the technique described in subsection 4.3.1 for steering principal directions of the latent representation. Figure 5.8 shows examples of a single person generating images with a GAN. More videos exemplifying the case of a single performer can be observed in our drive folder[6].

[2]Video with multiple people driving a GAN trained on faces: https://drive.google.com/open?id=1ZwVbi8Eb9XwZrzOpt-DoitBvwsaIdqVx.

[3]Video with a group of people generating landscape images: https://drive.google.com/open?id=1Qb1XavB8JGbbMVJZCAsE6i4JawqYHMlh.

[4]Video illustrating style mixing with audio and poses: https://drive.google.com/open?id=1YnvuaQHGFjZTXBcnHWGurKCBIg8FqqUY

[5]Videos illustrating face animation: https://drive.google.com/open?id=1SKctTZ1shIeE-hiCUmZVpFxOhY3oMm2F

[6]Videos with one person generating 2D content: https://drive.google.com/open?id=1rqiLKC_dT5r5e4RnjzH_yDeYX_Qa4Zf2

Chapter 6

Conclusion

In this work, we presented different approaches for generating visual effects in real time according to audience behavior. First, we showed how people's poses can drive 3D objects in a live setting. We also proposed a framework for applying a deep generative model such as StyleGAN2 [33, 34] as an artistic effects generator, and how the poses of the audience can influence the synthesized images. As an additional feature for live settings, we proposed using a face animation technique while adding other visual effects with stochastic behaviour.

In our generative framework, we applied a simple and efficient style sampling technique based on k-means clustering. By sampling from the intermediate latent space $\mathcal{W}$ of StyleGAN2, we were able to define a set of styles for mixing with the latent variables produced by keypoint detections. We also showed how settings with fewer people can employ PCA to drive the latent space of the generator in interpretable directions. Moreover, we introduced how simple audio features from music, such as pitch and energy spikes, can also control the generated visuals by defining style trajectories.

Later, we examined the feasibility of our framework for real-time applications by measuring the execution time under different assumptions. We concluded that depending on the parameters, the complete pipeline from frame grabbing to generating images is able to perform above 15 FPS. Furthermore, we evaluated how the synthesized frames in our generative framework vary depending on the input parameters. We showed that by applying a scaling factor to the latent representation of the mapping network results in more image diversity. We also presented some qualitative results, covering the use cases described in Chapter 2.

We additionally proposed using light-weight methods to account for possible jitter in the pose detection stage. When rendering 3D visual elements, we apply double exponential smoothing to the objects' parameters, resulting smoother visuals. Moreover, we perform interpolation in the intermediate latent representation of StyleGAN2 [34], which produces smoother transitions between synthesized images and decreases the execution time. Another possibility that we could further explore in the future is to use a pose tracking algorithm and evaluate whether it is also capable of reducing jitter effects in the generated visuals.

Future directions of our generative approach could include using a large scale conditional GAN such as BigGAN [9], and sample styles by applying the clustering step on the activation space in the generator network. For instance, similar to how Härkönen et al. [21] apply PCA on BigGAN's activation space. Furthermore, using a conditional GAN and adding the pose information to condition the generator could yield interesting results.

Other possible future work could show our generated visuals in a 3D model of a concert hall. Moreover, by employing 3D pose estimation methods, it would be possible to show digital avatars of event guests within the virtual venue.

Bibliography

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Giuseppe Amato, Malte Behrmann, Frédéric Bimbot, Baptiste Caramiaux, Fabrizio Falchi, Ander Garcia, Joost Geurts, Jaume Gibert, Guillaume Gravier, Hadmut Holken, et al. Ai in the media and creative industries. *arXiv preprint arXiv:1905.04175*, 2019.

[3] Scott E Anderson and Lindy L Wilson. Entertainment industry, computers in the. In *Encyclopedia of Computer Science*, pages 651–655. John Wiley and Sons Ltd., 2003.

[4] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, pages 3686–3693, 2014.

[5] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[6] Aayush Bansal, Shugao Ma, Deva Ramanan, and Yaser Sheikh. Recycle-gan: Unsupervised video retargeting. In *Proceedings of the European conference on computer vision (ECCV)*, pages 119–135, 2018.

[7] Regina Bernhaupt. *Game user experience evaluation*. Springer, 2015.

[8] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[9] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[10] Paul Brossier, Tintamar, Eduard Müller, Nils Philippsen, Tres Seaver, Hannes Fritz, cyclopsian, Sam Alexander, Jon Williams, James Cowgill, and Ancor Cruz. aubio/aubio: 0.4.9, February 2019.

[11] Chen Cao, Qiming Hou, and Kun Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Transactions on graphics (TOG)*, 33(4):1–10, 2014.

[12] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[13] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7291–7299, 2017.

[14] Martin Čertickỳ, Michal Čertickỳ, Peter Sinčák, Gergely Magyar, Ján Vaščák, and Filippo Cavallo. Psychophysiological indicators for modeling user experience in interactive digital entertainment. *Sensors*, 19(5):989, 2019.

[15] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[16] Alain De Cheveigné and Hideki Kawahara. Yin, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.

[17] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[18] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[19] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[20] Stefan Hall, Claudio Cocorocchia, and Ryo Takahashi. Creative disruption: The impact of emerging technologies on the creative economy. Technical report, World Economic Forum, February 2018.

[21] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *arXiv preprint arXiv:2004.02546*, 2020.

[22] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[24] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pages 6626–6637, 2017.

[25] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017.

[26] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017.

[27] Mark J Huiskes and Michael S Lew. The mir flickr retrieval evaluation. In *Proceedings of the 1st ACM international conference on Multimedia information retrieval*, pages 39–43, 2008.

[28] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[29] Ali Jahanian, Lucy Chai, and Phillip Isola. On the "steerability" of generative adversarial networks. In *International Conference on Learning Representations*, 2020.

[30] Sam Johnson and Mark Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *Proceedings of the British Machine Vision Conference*, 2010. doi:10.5244/C.24.12.

[31] Sam Johnson and Mark Everingham. Learning effective human pose estimation from inaccurate annotation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[32] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[33] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[34] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. *arXiv preprint arXiv:1912.04958*, 2019.

[35] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[36] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[37] Joseph J LaViola. Double exponential smoothing: an alternative to kalman filter-based predictive tracking. In *Proceedings of the workshop on Virtual environments 2003*, pages 199–206, 2003.

[38] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[39] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[40] Ramón López de Mántaras. Artificial intelligence and the arts: Toward computational creativity. In *The Next Step: Exponential Life*. Fundación BBVA, 2016.

[41] L. McInnes, J. Healy, and J. Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv e-prints*, February 2018.

[42] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[43] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Deepdream-a code example for visualizing neural networks. *Google Research*, 2(5), 2015.

[44] Maximilian Muller-Eberstein and Nanne van Noord. Translating visual art into music. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[45] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016.

[46] D Oved, I Alvarado, and A Gallo. Real-time human pose estimation in the browser with tensorflow.js. *Retrieved from TensorFlow Blog: https://blog. tensorflow. org/2018/05/real-time-humanpose-estimation-in. html*, 2018.

[47] George Papandreou, Tyler Zhu, Liang-Chieh Chen, Spyros Gidaris, Jonathan Tompson, and Kevin Murphy. Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 269–286, 2018.

[48] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4903–4911, 2017.

[49] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[50] Stephen M Pizer. Contrast-limited adaptive histogram equalization: Speed and effectiveness stephen m. pizer, r. eugene johnston, james p. ericksen, bonnie c. yankaskas, keith e. muller medical image display research group. In *Proceedings of the First Conference on Visualization in Biomedical Computing, Atlanta, Georgia, May 22-25, 1990*, page 337. IEEE Computer Society Press, 1990.

[51] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[52] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[53] Rudolf Richter, Tobias Kellenberger, and Hermann Kaufmann. Comparison of topographic correction methods. *Remote Sensing*, 1(3):184–196, 2009.

[54] Miguel Sanz-Narrillos, Stefano Masneri, and Mikel Zorrilla. Combining video and wireless signals for enhanced audience analysis. In *International Conference on Agents and Artificial Intelligence*, 2020.

[55] Ben Sapp and Ben Taskar. Modec: Multimodal decomposable models for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3681, 2013.

[56] Othman Sbai, Mohamed Elhoseiny, Antoine Bordes, Yann LeCun, and Camille Couprie. Design: Design inspiration from generative networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[57] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. Animating arbitrary objects via deep motion transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2377–2386, 2019.

[58] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. In *Advances in Neural Information Processing Systems*, pages 7135–7145, 2019.

[59] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference for Learning Representations*, 2015.

[60] Malcolm Slaney. Auditory toolbox, 1998.

[61] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2387–2395, 2016.

[62] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in neural information processing systems*, pages 1799–1807, 2014.

[63] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014.

[64] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1. 0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[65] M Mitchell Waldrop. *The dream machine: JCR Licklider and the revolution that made computing personal*. Viking Penguin, 2001.

[66] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. *arXiv preprint arXiv:1808.06601*, 2018.

[67] Olivia Wiles, A Sophia Koepke, and Andrew Zisserman. X2face: A network for controlling face generation using images, audio, and pose codes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 670–686, 2018.

[68] Ceyuan Yang, Yujun Shen, and Bolei Zhou. Semantic hierarchy emerges in deep generative representations for scene synthesis. *arXiv preprint arXiv:1911.09267*, 2019.

[69] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver 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.

[70] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.