POLITECNICO DI TORINO

Master's Degree in Biomedical Engineering
(Ingegneria Biomedica LM-21, DM270)

Master's Degree Thesis

DEEP LEARNING DENOISING OF TIME-OF-FLIGHT DEPTH IMAGES AFFECTED BY RETRO-REFLECTIVE MARKER INTERFERENCE TO ENABLE CONCURRENT MARKER-BASED AND MARKERLESS MOTION CAPTURE

Supervisors

Prof. Andrea Cereatti

Prof. Bart Jansen

PhD Silvia Zaccardi

PhD Diletta Balta

Candidate

Federico TROVALUSCI

s302900

OCTOBER 2025

Collaboration and Internship Statement

The work presented in this thesis was carried out in partnership with ETRO-VUB (Department of Electronics and Informatics, Vrije Universiteit Brussel) as part of an in-person collaboration intership in Brussels.

Summary

Human Motion Capture (MoCap) systems are essential in medical diagnostics and rehabilitation, providing valuable quantitative insights into posture, joint movements, and motor functions of patients with neuro-motor diseases [1, 2, 3, 4, 5, 6]. Image-based MoCap systems can be classified in two categories: marker-based and markerless. Marker-based systems, such as the Vicon, utilize multiple Infrared (IR) cameras to accurately track the position of reflective markers placed on the patient's skin. However, their high cost, complex setup, and lack of portability limit their widespread use. On the other hand, markerless systems rely on Deep Learning (DL) models to estimate the human body position from images, offering a more affordable and portable alternative at the cost of a lower accuracy [7, 8, 9]. Among these, a popular system is the Microsoft Azure Kinect [10], which relies on a single dual-sensor RGBD camera capable to reconstruct depth images with infrared Time-of-Flight (ToF) technology.

Concurrent use of marker-based and ToF-markerless motion capture systems causes significant interference, primarily due to IR reflections from retro-reflective markers [11, 12, 13, 14, 15]. This leads to two critical challenges:

- It complicates the validation of Kinect data against the marker-based gold standard, leading researchers to opt for non-concurrent validation studies.
- It prevents the collection of clean, simultaneous datasets combining ToF images with accurate marker-based data. As a result, researchers typically rely on simulated datasets to train their DL skeletal tracking models.

The main objective of this thesis is to develop methods to automatically remove IR artifacts introduced by retro-reflective markers from ToF images captured with an Azure Kinect during concurrent acquisitions with a Vicon marker-based system.

The study relies on a dataset of 8 healthy subjects performing 4 simple motor tasks, previously collected at the VUB Rehabilitation Research Center (RERE) in Jette [16]. Two acquisitions were made available for each subject: one with reflective markers applied on the subject's skin (i.e. noisy images) and one without markers (clean images).

The work presented in this thesis can be divided into four main sections.

- The first section focuses on data preprocessing. Background information was removed from the raw Azure Kinect depth images, and a fixed size Region Of Interest (ROI) around the subject was extracted to standardize the input for the model training.

- The second section presents the development and validation of a DL framework to remove IR interference from depth images. A latent-space diffusion model (with a companion autoencoder) was trained on clean depth images from 6 participants to generate realistic synthetic images. During inference, an inpainting algorithm guides the model to modify only the regions of the input image affected by artifacts and missing values, while leaving the rest unchanged.

  The first experiment was conducted on the clean images of the 2 test participants. To simulate missing data, artificial checkerboard masks of varying sizes were applied, and the model was guided to reconstruct the masked regions. Results showed low Mean Square Error (MSE) between the reconstructed and original images of $5.45 \times 10^{-3} \pm 2.02 \times 10^{-3}$ for smaller 8×8 pixel squares and $8.25 \times 10^{-3} \pm 2.4 \times 10^{-3}$ for larger 32×32 pixel squares, demonstrating the model's ability to accurately recover missing pixels.

  The second experiment was conducted on the noisy images of the same 2 test participants. To perform inpainting, a marker detection algorithm was implemented to identify the corrupted regions, which the model then reconstructed. Since ground truth is not available, a pixel-level accuracy metric could not be computed. Therefore, performance was evaluated using the Kernel Inception Distance (KID) which quantifies the similarity between the inpainted outputs and the clean training distribution. The statistically significant decrease in KID values (from the input noisy images to the inpainted ones) indicated that the model produced cleaner images more consistent with the clean training data.

- The third section evaluates the effect of integrating the denoising model into the standard Kinect SDK pipeline as a preprocessing step on the depth channel. Results obtained inpainting only the depth channel showed no improvement in skeletal tracking accuracy, suggesting that the Kinect body tracking model (K4ABT) is indifferent to depth channel modifications.

- The last section was conducted together with the researchers at ETRO (VUB's department of Electronics and Informatics). We found that Kinect's skeletal tracking primarily relies on IR images rather than depth images. Therefore, we developed a simple yet effective algorithm to mitigate noise in IR images. Our

method significantly improved Kinect's skeletal tracking reliability, reducing missed poses from 11.49% to 0.16%), with a significant reduction in bone length variability [17, 18].

This thesis introduced a novel approach for removing marker-induced artifacts from ToF depth images by adapting and re-training a diffusion-based generative model, originally designed for radiology imaging [19, 20]. This method enables the creation of clean, paired datasets of depth images and marker-based motion capture data, allowing future training of depth-based DL skeletal tracking model. The work also contributed new insights into the Azure Kinect's skeletal tracking pipeline, revealing that the algorithm relies more heavily on the IR stream than the depth. To address this, a simple IR inpainting algorithm was implemented, resulting in significantly improved tracking stability and reduced variability in bone length estimates.

Table of Contents

List of Figures

List of Tables

Acronyms

API

Application Programming Interface

BCE

Binary Cross-Entropy

BGRA

Blue, Green, Red, Alpha channels

C3D

3D biomechanics data file format

CMC

Coefficient of Multiple Correlation

CNN

Convolutional Neural Network

DDPM

Denoising Diffusion Probabilistic Model

DL

Deep Learning

DSP

Digital Signal Processing

ELBO

Evidence Lower Bound

ETRO

Department of Electronics and Informatics (VUB)

FoV

Field of View

FPS

Frames Per Second

HABL

Left hip abduction with left arm abduction

HABR

Right hip abduction with right arm abduction

ICC

Intra-class Correlation Coefficient

IMU

Inertial Measurement Unit

IR

Infrared

K4A

Kinect v4 Azure

K4ABT

Kinect Azure Body Tracking (SDK v4)

KFB

Squat (Knee flexion)

KID

Kernel Inception Distance

KL

Kullback-Leibler divergence

MKV

Matroska multimedia container

MoCap

Motion Capture

MPJPE

Mean Per Joint Position Error

MSE

Mean Squared Error

PAF

Part Affinity Field

PCK

Probability of Correct Keypoint

POV

Point of View

RBF

Radial Basis Function

RERE

Rehabilitation Research Center (VUB)

RGB

Red, Green, Blue channels

RMSE

Root Mean Squared Error

ROM

Range of Motion

SAB

Bilateral shoulder abduction

SDK

Software Development Kit

U-Net

U-Net architecture (encoder-decoder CNN)

VAE

Variational Autoencoder

VUB

Vrije Universiteit Brussel *free university of Bruxelles*

Chapter 1

Introduction

Image-based human motion capture systems enable medical professionals to gain insightful quantitative information about the posture, range of motion, and overall quality of the patient's movements by measuring the positions and angles of joints. For that reason, integrating such devices in the diagnosis and rehabilitation of patients with neuro-motor diseases [1] and stroke survivors [2] has been an active field of research over the past two decades. The widespread availability of single-camera markerless motion capture systems, such as the Microsoft Kinect [10], popularized by the Virtual Reality (VR) gaming industry, opened new possibilities in providing a more affordable and more portable alternative to expensive marker-based multi-camera systems, such as the Vicon. Those cheaper markerless systems, however, have been shown to be less reliable than their marker-based counterparts [7, 8, 9], and rely heavily on AI models (e.g. the Microsoft Kinect SDK for Windows [21]) to perform skeletal tracking. Concurrent validation studies of the Kinect device against a Vicon marker-based system are also made difficult by the noise and image artifacts caused both by the IR interference between the multiple infrared light sources, and the presence of retro-reflective markers that disrupt the depth image reconstruction and the skeletal tracking[11, 12].

For that reason the objective of this master thesis is to develop an automated marker artifacts removal pipeline based on diffusion probabilistic models that could be used to obtain a concurrent dataset of paired images (with and without markers) in the absence of a ground truth. By synthetically generating marker-free depth images from images affected by marker artifacts, researchers could run a typical AI based Kinect body-tracking pipeline on the inpainted images, as if they were obtained via a standard Kinect recording, and then use marker positions obtained by the Vicon system as a ground-truth reference for evaluating joint angles and other biomechanical parameters.

Chapter 2

Introduction to Motion Capture

Motion capture (MoCap) is the process of digitally reconstructing the 3D kinematics of human motion from recordings acquired through sensors (such as cameras, IMUs, force plates) in order to extract spatiotemporal features, estimating joints range of motion, and perform inverse-dynamics analyses.

The clinical relevance of computer-assisted motion capture has been demonstrated in the diagnosis and care of multiple conditions such as:

- Diagnosis and clinical planning for adult neurological disorders (e.g., Parkinson's disease, multiple sclerosis, epilepsy) [1].
- Care and rehabilitation of stroke survivors [2].
- Early detection of cerebral palsy in infants [3].
- Evaluation of hereditary neuro-pathologies [4].
- Assessment and treatment of dementia and frailty in elderly adults [5].
- Outcome assessment and rehabilitation in orthopedic patients [6].

In all these applications, the contribution of motion-capture technologies has been to obtain repeatable measurements that assist clinicians in detecting and identifying symptoms, clustering patients, and, in some cases, providing an inference tool for the early screening of certain conditions. However, in most practical settings, these methods are device specific and are highly dependent on sensors calibrations, preprocessing and postprocessing methods, and, despite showing promising levels of sensitivity and specificity, they currently lack a general, standardized protocol.

2.1 A comparison between motion capture methods

The two main modalities to perform MoCap in clinical applications are:

- **Inertial MoCap:** uses magneto-inertial measurement units (IMUs) attached to the subject's limbs and trunk [22] (fig. 2.1) to estimate segment orientations and joint kinematics [23]. Outputs are typically presented as time-series plots and numerical parameters; direct visual assessment is not available to the clinician unless a synchronized video recording is acquired. Inertial systems can be integrated into wearable devices that enable out-of-the-lab analysis of patients in daily life activities. Results are highly dependent on sensor placement, subject's activity and the duration of the acquisition.
- **Optical MoCap:** uses cameras in the visible/infrared spectrum, including multi-camera infrared systems and RGB-D/depth cameras [24]. Optical methods are commonly divided into:
  - *Marker-based* methods, like Qualisys [25] and Vicon [9], use stereophotogrammetry and employ multi IR cameras setup with retro-reflective markers applied on the subject's body;
  - *Markerless* methods use computer-vision to perform pose estimation. Markerless systems can employ an array of digital cameras in the visible or IR spectrum, like Theia [26], or rely on single POV or multi view RGB-D devices, like Kinect. Outputs, such as joints locations, can often be overlaid on synchronized video recordings of the patient, especially when using devices that integrate an RGB modality, enabling direct visual assessment. Even when using portable systems, patient assessment outside the laboratory remains limited by the camera's field of view.

MoCap kinematics systems can also be paired with force plates [27] to directly measure ground-reaction forces and moments, providing the external kinetics ground truth required for inverse-dynamics computations.

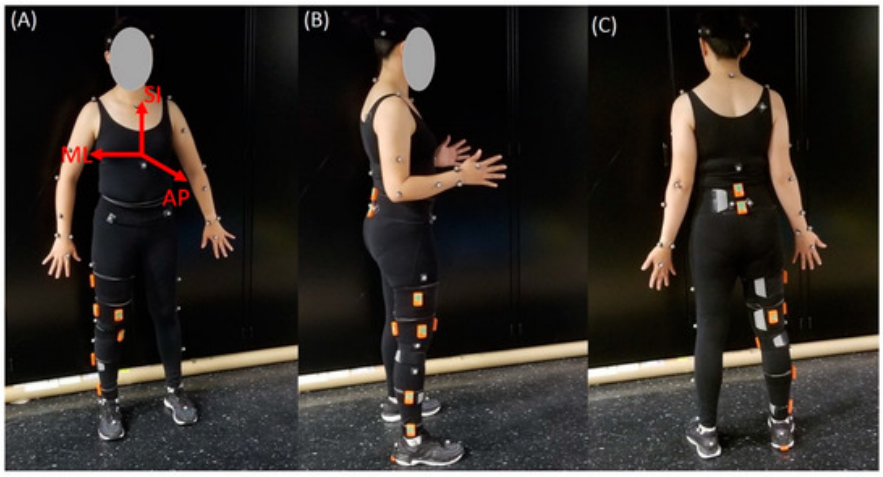

Figure 2.1: Frontal (A), lateral (B) and rear (C) view of a subject with multiple Xsens IMU devices placed on the hip and right leg during a validation study. Image obtained from Niswande, Wang et al. [22]: «Optimization of IMU Sensor Placement for the Measurement of Lower Limb Joint Kinematics». under creative commons license.

Figure 2.2: An example of a combined marker-based + force plate experimental setup. Image obtained from Conceição, Lewis et al. [27] «An Evaluation of the Accuracy and Precision of Jump Height Measurements Using Different Technologies and Analytical Methods» under creative commons license.

2.1.1 Marker based vs markerless optical motion capture systems

Marker-based systems

Marker-based systems use an array of fixed infrared cameras surrounding the capture volume. Motion capture is achieved via stereophotogrammetry by identifying Retro-reflective markers attached to the subject in different views and triangulating their 3D positions according to a reference frame of 3 non aligned reference markers.

They are considered the gold standard [28] to obtain accurate representation of the position and orientation of the body segments in both the spatial and time domain due to their accuracy (≤ 1 mm) [7] and high frame rate (≥ 250 FPS) [9].

Their main limitations include the high entry cost, which makes them inaccessible in application where the low cost is a necessity and their lack of portability, since their setup must be located in a laboratory, which make them not available in many rehabilitation and sports training applications [28].

Another technical limitation is the operator-dependent marker placement, which requires a skilled operator to locate anatomical landmarks through palpation [29]. This operator discretionality can introduce systematic inter-study disagreement bias that can exceed the system's measurement uncertainty.

Markerless devices

Some multi-camera markerless systems, like Theia, employ at least six synchronized cameras to segment subjects and reconstruct skeletal pose using computer-vision algorithms, reaching excellent levels of agreement with state of the art marker-based systems [30]. While they remove the need for marker placement, they share key limitations of marker-based setups, such as the high cost and large setup space requirements. Theia's cameras can be setup outdoor, but a careful planning is required to ensure optimal field of view for each camera. Furthermore, such systems rely on pipelines that use proprietary calibration and processing software.

Cheaper markerless devices, like Kinect, commonly employ a single RGB-D sensor to estimate the 3D scene within the frontal field of view (which is conical). Depth values can be estimated using single POV devices in two ways [31]:

- **Structured light:** the device projects a known infrared (IR) speckle pattern and an IR camera images the deformation this projected pattern goes through diffusing on scene surfaces. Pattern disparities such as changes in clustering are used with projector-camera intrinsic parameters to recover depth.

- **Time-of-flight (ToF):** the device emits pulsed or modulated IR light and measures the round-trip time (or phase shift) of the returned signal at each

pixel. Depth is computed from the propagation delay $\Delta t/2$: since the speed of light is a fundamental constant, $d = c\,\Delta t/2$.

Their main advantages are portability, low cost, and plug-and-play operation requiring minimal setup. Their main disadvantages are the lower accuracy ($\geq$ 8 mm) [8] and lower frame rate ($15 - 30$ FPS).

They usually rely on device-dependent deep learning models to perform subject detection, body segmentation, skeletal tracking, and pose estimation.

2.2 The Azure Kinect device

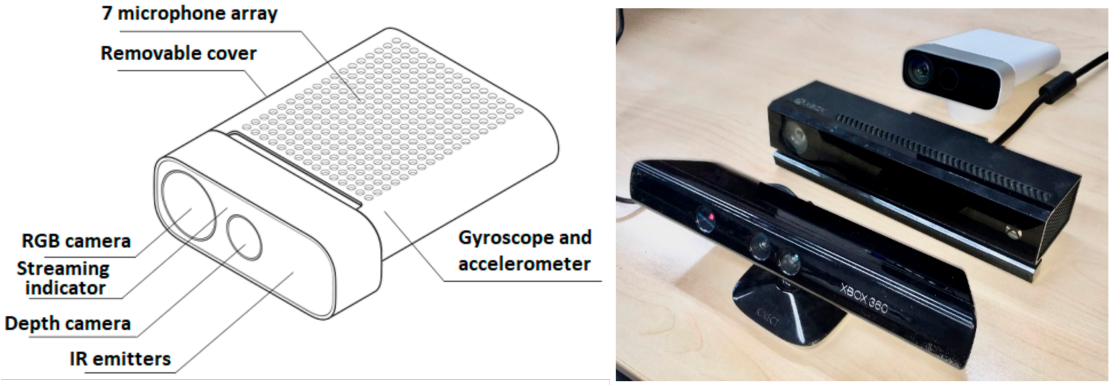

Figure 2.3: Azure Kinect device sensors schematics and its comparison against previous generations of Kinect devices, obtained from Tölgyessy, Dekan et al. [10] «Evaluation of the Azure Kinect and Its Comparison to Kinect V1 and Kinect V2» under creative commons license.

Some of the most widely available single-camera RGB-D devices come from Microsoft's Kinect product line. Launched in the 2010*s* and popularized by the video game and AR industry, they represent a promising alternative to more expensive MoCap systems to evaluate several kinematic parameters.

The Azure Kinect (K4A) is the fourth and last iteration of the series, introduced in 2019. The device integrates several sensors in a portable form factor [10, 21, 32]:

- one **RGB** 12MP camera with resolution 3840×2160 (16:9) or 4096×3072 (4:3),
- one **IR/depth-ToF** 1MP camera with resolution 640×576 (Narrow FoV) or 1024×1024 (Wide FoV)
- one **IMU** with 3 axis accelerometer and 3 axis gyroscope (208 Hz)

It outputs a Matroska file (`.mkv`) at 15 FPS (full resolution) with:

- an **RGB channel** with BGRA32 bit depth (3×8 bits for RGB, 1×8 bits for α-channel)
- a **Depth channel** with DEPTH16 bit depth, natively converted to mm
- an **IR channel** with IR16 bit depth
- an array of six **IMU tracks**

2.2.1 Kinect Azure Body Tracking SDK Version 4 (K4ABT)

As anticipated in paragraph 2.1.1, the Azure Kinect performs motion capture relying on a deep learning based pipeline. Azure Kinect Body Tracking SDK is an API made available by Microsoft that enables developers to parse Azure Kinect `.mkv` recordings to output multi-person, temporally coherent 32-joint 3D skeleton estimates.

According to the available documentation [32], for every frame, a CNN is run on the IR and depth channels to predict a set of 18 per-joint $2D$ confidence heatmaps $H_k(x, y)$ called Keypoint heatmaps. For each $k \in K$ the local peaks of the k_{th} heatmap give the $2D$ keypoints that represent the best estimates of joint k's position. Having multiple candidates positions for each joint improves robustness and help keep the tracking consistent in presence of occlusions.

A tracker object then attempt to construct limbs by generating 17 2-channels vector fields, one per limb type. For each of the pixels that lie on that limb, the vectors in the corresponding field point along the estimated limb direction. These vector fields are called Part Affinity Fields. Integrating the PAFs along the line segment that connects each pair candidate joints the tracker evaluates association scores that enable the construction of each subject's candidate skeleton from the bottom up.

The same CNN predicts a set of 15 Body-part segmentation masks that estimate per-pixels class probabilities for coarse anatomical parts. These semantic masks provide dense spatial context to regularize the skeletal tracking.

Each of the skeletons grouped from the bottom up is assigned an ID, and a Body Index map is generating mapping each pixel to the subject it belongs to. All the other pixels are marked as background.

A 32-joints $3D$ skeleton is generated for each subject by back-projecting the $2D$ keypoints in the $3D$ camera space coordinates using the depth information. This is achieved by fitting an articulated skeleton model parametrized so that:

- joint angles that are bounded by anatomical joint limits
- rotations and translations of joint centers are constrained
- body proportions are maintained via a person-specific scale factor estimated from a statistical prior.

The skeleton model is regularized to maximize temporal (constrained speed) and pose (pose prior) coherence.

The final output consists of a set of 32-joints skeletons, with $3D$ joints positions expressed in millimeters in the camera reference frame, and joints orientations expressed in quaternions. Each of the joint is also paired with a confidence level.

Chapter 3

Literature Review

3.1 Noise sources in Kinect recordings

Kinect recordings can be affected by noise that degrade the quality of the depth reconstruction and disrupt the skeletal tracking performances, especially when a markerless MoCap system and a markerless RGB-D device are used at the same time to obtain a concurrent acquisition. This happens because both platforms emit and sense in the IR spectrum. While the Kinect's emission is too weak to meaningfully affect the optical system, the strong IR illumination used by the Vicon system can saturate the Kinect's time-of-flight sensor and disturb its phase/return measurements. The effect is further exacerbated by retroreflective markers, which can reflect incident IR at very high intensity, producing bright blobs in the Kinect's IR images and artifacts in the depth images.

Mallick et.al. [11] tried to introduce a nomenclature to characterize the noise in Kinect depth and IR images, providing two initial definitions:

- **Spatial noise**, defined as all kinds of noise observable within a single depth frame.
- **Temporal noise**, defined as observed depth instability across various frames that can be measured even when the scene has no motion.

Mallick measured the presence of both these types of noise in different experimental conditions, and provided a series of parameters that influence noise intensity of each class.

Naamebadi confirmed Mallicks results performing experiments on a static flat surface [12] and on a mannequin [13], and provided two additional definitions:

- **Active noise sources**, defined as devices that actively injects IR light into the scene.

- **Passive noise sources**, defined as objects (like the retroreflective markers) that reflect IR light in the direction of the Kinect device at an intensity greater than expected.

The results obtained by Naamebadi shows that passive noise sources contributed more than active sources to variations in estimated bone lengths produced by skeletal tracking.

The presence of retroreflective markers appears in IR images as points with much higher intensity values than their surroundings and are readily segmented via simple thresholding, as noted by Chatzitofis et al. [14](fig. 3.1-b). In depth images, by contrast, they manifest as blob-like artifacts, as shown by Hesse et al. [15], which are considerably harder to remove(fig. 3.1-d).

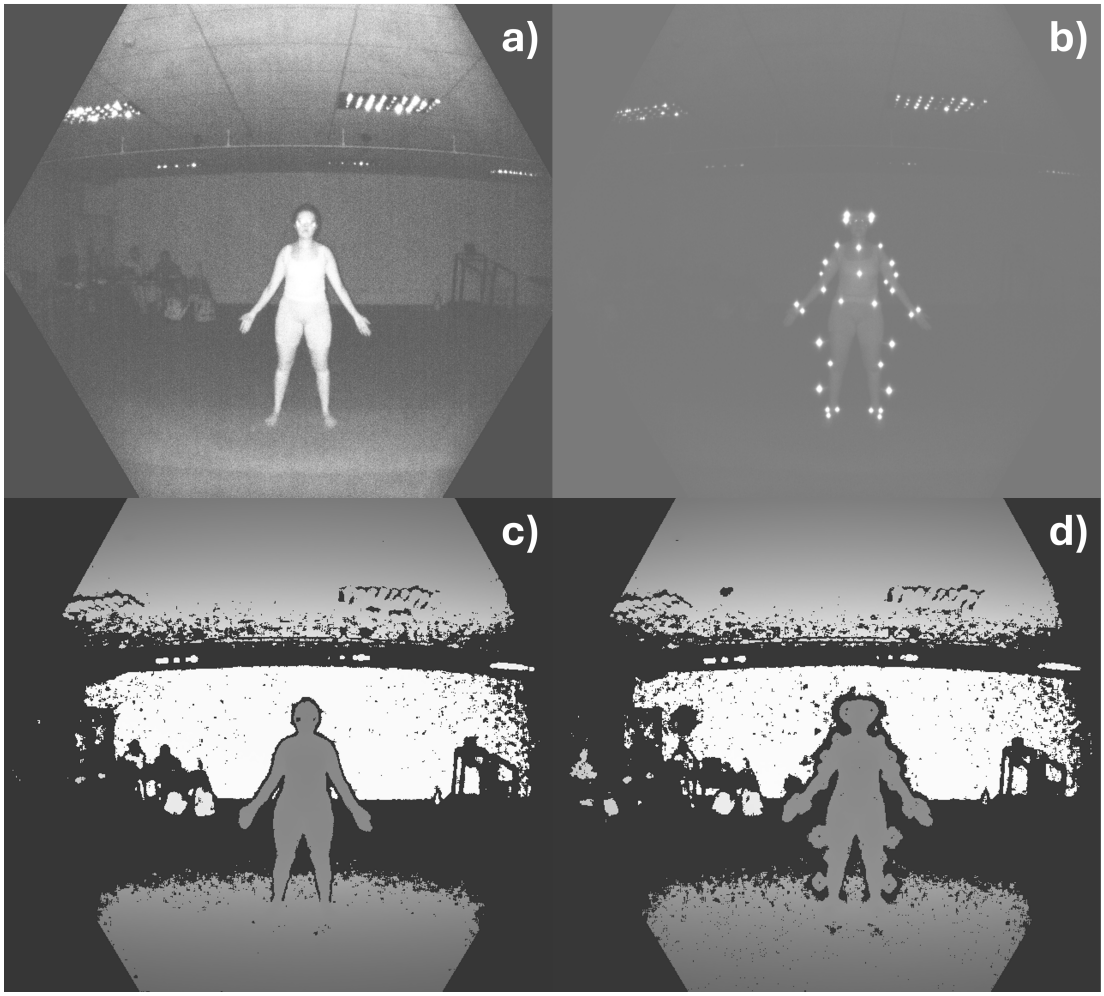

Figure 3.1: Effects of retroreflective markers on infrared (IR) and depth images. (a) IR channel without markers. (b) IR channel with markers: saturated, high-intensity reflections at marker locations. (c) Depth channel without markers. (d) Depth channel with markers: blobs caused by IR reflections during ToF depth computation.

3.2 Kinect validation studies

Multiple studies have attempted to validate the Azure Kinect, in combination with deep-learning methods, for clinical motion-capture applications such as gait analysis, sit to stand test, treadmill walking, with mixed results.

In 2020, Ma, Sheng, Hart, and Zhang [33] reported that, relative to a Vicon ground truth in gait analysis, a dual-Azure Kinect motion-capture system produced accurate knee angles (CMC $= 0.87 \pm 0.06$, RMSE $= 11.9° \pm 3.4°$), moderate agreement for hip sagittal angles (CMC $= 0.60 \pm 0.34$, RMSE $= 15.1° \pm 6.5°$), and poor validity for hip frontal/transverse and ankle angles.

In 2021, Ota *et al.* [34] showed that a single RGB-D camera combined with an OpenPose-based posture-tracking algorithm achieved large associations and moderate-to-excellent agreement with Vicon for sagittal-plane measures in treadmill *gait* analysis, especially for knee angles and ROM, and ankle ROM. Peak knee flexion during running (ICC $= 0.93$), knee ROM during slow walking (ICC $= 0.91$), and hip flexion-extension ROM during running (ICC $= 0.86$) exhibited predominantly fixed (non-proportional) biases, whereas pelvis metrics and hip frontal-plane angles showed non-significant R^2 and poor ICCs with both fixed and proportional biases.

In 2022, Chatzitofis *et al.* [14] presented a low-cost, real-time marker-based motion-capture system using a sparse set of 3-6 Azure Kinect RGB-D sensors, in which retroreflective markers are robustly segmented as saturated IR blobs (despite creating depth "blind spots"); multi-sensor data are fused after a simple single-marker wand calibration; and a Lite-HRNet variant of DeMoCap denoises ghosting/missing markers to infer joints at 30 FPS.

In 2022, Thomas *et al.* [35] found that the Azure Kinect exhibited high agreement with a 12-camera Vicon system in the sit-to-stand test for continuous kinematic and spatiotemporal waveforms (all $r > 0.711$, $R^2 > 0.660$). Despite relatively high RMSE, strong correlations were observed for several discrete parameters, including phase-point detection and total time ($r \approx 0.99$) and maximal knee extension between phases 3-4 ($r = 0.90$). Medial-lateral pelvic sway, however, showed poor agreement ($r = 0.44$, n.s.).

In 2022, Guess *et al.* [36] demonstrated that a single Azure Kinect (Body Tracking SDK) yields spatiotemporal gait parameters in strong to very strong agreement with a 12-camera Vicon system ($r! \geq! 0.87$ across stride length, stride time, step length, and step width), with mean stride-length differences of $35.6, \text{mm}$ (left) and $39.1, \text{mm}$ (right; 3% of average stride length) and biases for step width and stride time $< 2\%$ and $< 1\%$ of their respective averages, supporting clinically relevant over-ground walking assessment.

In 2023, Hesse *et al.* [15] showed that their custom RGB-D-based markerless tracker closely agrees with a marker-based Vicon reference across five motor tasks in 23 children and young adults (Mean Per Joint Position Error (MPJPE) $= 11.7, \text{mm}$;

Probability of Correct Keypoint PCK50mm = 98.4, %; Pearson $r > 0.95$ for moving joints and $r > 0.99$ for high-motion joints), whereas Azure Kinect Body Tracking (K4ABT) exhibited frequent short tracking failures, especially with reflective markers (no-body detections in 44/107 sequences), and larger errors (MPJPE 45.7, mm with markers, 26.9, mm without; PCK100mm = 95.5, % markerless), indicating improved performance without markers but persistent limitations for clinical motion analysis.

Across these studies, even when statistically significant agreement with marker-based gold standards is achieved, this agreement usually involve only a narrow set of joints or spatiotemporal parameters. Gains in accuracy for sagittal plane knee or hip measures often causes worse estimates for frontal or transverse plane angles, pelvic metrics, or ankle kinematics. Improvements in timing measures can come with larger errors in joint angles, and vice-versa.

The reported pipelines are typically designed for the specific task examined and include custom setups, fine tuned processing, custom machine learning algorithms, ad hoc solutions. These choices increase performance on the specific parameters measured for the target task, but reduce reproducibility. Transfer to different tasks is limited, and changes in capture spaces or patient populations were not taken into account.

To date, the potential of denoising diffusion models to remove artifacts in Kinect depth data has not been systematically investigated.

Chapter 4

Theoretical Background: Generative AI

4.1 Diffusion Probabilistic Models

Denoising Diffusion Probabilistic Models (DDPMs) [20, 37, 38] are a class of generative probabilistic models that operate in a latent space to denoise data, such as images, by gradually reversing a diffusion process. Statistical diffusion processes, as used in Denoising Diffusion Probabilistic Models, are inspired by physical diffusion processes [39], such as the way particles disperse in space following to the laws of Brownian motion. In a statistical diffusion process, each sample in a dataset, that can be represented as a coordinate point in an n-dimensional space at time $t = 0$, gradually moves away from its original position as random noise with zero mean and small β variance is incrementally added over time. This process resembles the random movement of particles in physical diffusion, where the future position in space of each particle can be described probabilistically, given its previous position.

4.1.1 Reverse process: Denoising

The DDPM can be trained on a dataset to perform the reverse (denoising) process by maximizing the marginal likelihood on the training dataset. The marginal likelihood quantifies the probability that the model, with parameters θ, would generate the given data sample x_0, considering all possible ways x_0 could be obtained through the generative process. The diffusion probabilistic model operates over a set of latent variables $[x_0, \ldots, x_T]$ each with the same dimensionality as x_0, corresponding to the sequence of every progressively noisier intermediate versions of the original data that can be obtained adding random noise to x_0 in small

steps. Every possible sequence of latent space states $x_{0:T}$ is called a trajectory. The computation of the marginal likelihood of observing x_0 as the outcome of a given sequence of latent intermediates (latent trajectory), marginalized over every possible trajectory in the latent space, can be formulated as [37] integrating the joint probability distribution of the entire trajectory through the latent space over all possible values of the latent variables $x_{1:T}$ that can lead to x_0.

$$p_\theta(x_0) = \int p_\theta\left(x_{0:T}\right) dx_{1:T} \tag{4.1}$$

The reverse (denoising) process along the trajectory $x_{0:T}$, characterized by the joint probability distribution $p_\theta\left(x_{0:T}\right)$, is defined as a Markov chain with learned gaussian transitions [37]: each step of the process only depends on the internal state at the end of the previous iteration, and the probability of moving from the intermediate noisy step x_t to the slightly less noisy step $x_t - 1$ is a Gaussian function with learned parameters μ and Σ

$$p_\theta(x_{t-1} \mid x_t) := \mathcal{N}\Big(x_{t-1};\, \mu_\theta(x_t, t),\, \Sigma_\theta(x_t, t)\Big) \tag{4.2}$$

This characteristic distinguishes this models from other architectures frequently used for denoising, like Recurrent Neural Networks such as Long Short Term Memory (LSTM) networks, where the output is conditioned by an evolving hidden memory of several previous states.

The joint probability [37, 39] of the full reverse trajectory $x_{0:T}$, from random noise x_T to the original data sample x_0, can be expressed as the product of the conditional probabilities of each sequential transition of the Markov chain:

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t) \tag{4.3}$$

4.1.2 Forward process: Adding noise

The forward diffusion process [37], which incrementally adds Gaussian noise to x_0 over T steps, can be represented as a Markov chain. At each step, a small quantity of noise is added according to a predefined variance schedule $\beta_1, \ldots, \beta_T$ with $\beta \in [0,1]$. The conditional probability of obtaining x_t given x_{t-1} is defined as:

$$q(x_t \mid x_{t-1}) := \mathcal{N}\left(x_t;\, \sqrt{1-\beta_t}\, x_{t-1},\, \beta_t \mathcal{I}\right) \tag{4.4}$$

By chaining together all T forward transitions in a trajectory $x_{0:T}$, we obtain the approximate posterior $q(x_{1:T} \mid x_0)$, which defines the full forward process probability distribution [37, 39]. This distribution expresses the probability of the entire latent

trajectory $x_{1:T}$, given the original data x_0:

$$q(x_{1:T} \mid x_0) := \prod_{t=1}^{T} q(x_t \mid x_{t-1}) \tag{4.5}$$

Equation (4.4) implies that at each step t of the forward process, we can obtain x_t from x_{t-1} by sampling from a normal distribution with mean $\sqrt{1-\beta_t}\, x_{t-1}$ and variance β_t. The sampling operation from a Gaussian distribution with known mean μ and variance σ^2 can be interpreted as adding a bias equal to μ to a random sample ε drawn from a standard (zero mean, unit variance) normal distribution $\mathcal{N}(0, \mathcal{I})$, scaled by σ. Specifically, drawing a sample from $\mathcal{N}(\mu, \sigma^2\mathcal{I})$ is equivalent to sampling $\varepsilon \sim \mathcal{N}(0, \mathcal{I})$ and computing:

$$x_{\text{sample}} = \mu + \sigma\varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \mathcal{I}) \tag{4.6}$$

Given an initial data sample x_0 and a predefined variance schedule $\beta_{1:T}$, the forward diffusion process of adding Gaussian noise at each timestep can be recursively defined as follows:

$$\begin{aligned}
x_1 &= \sqrt{1-\beta_1}\, x_0 + \sqrt{\beta_1}\, \varepsilon_1 \\
x_2 &= \sqrt{1-\beta_2}\, x_1 + \sqrt{\beta_2}\, \varepsilon_2 \\
x_3 &= \sqrt{1-\beta_3}\, x_2 + \sqrt{\beta_3}\, \varepsilon_3 \\
&\vdots \\
x_t &= \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \varepsilon_t
\end{aligned} \tag{4.7}$$

where $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_t \sim \mathcal{N}(0, \mathcal{I})$ are independent Gaussian noise samples.

Because each step in the forward diffusion process consists of a linear transformation followed by the addition of independent Gaussian noise, the composition of multiple steps remains Gaussian. As a result, we can analytically derive a closed-form expression for the arbitrary timestep t of the forward process:

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \mathcal{I}) \tag{4.8}$$

Here, $\alpha_t := 1 - \beta_t$ is the retained signal coefficient at step t, and $\bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$ is the cumulative product of these coefficients up to timestep t, representing the total amount of signal preserved after t steps of noise injection.

The expression in Eq. (4.8) describes the sampling x_t at an arbitrary timestep t by combining the original data sample x_0 with Gaussian noise ε in a closed form. Since ε is sampled from a standard normal distribution and both scaling operations are linear, this implies that x_t itself is distributed according to a Gaussian. The

closed-form expression for the marginal conditional distribution of x_t given x_0 can be written as:

$$q(x_t \mid x_0) = \mathcal{N}\left(x_t;\ \sqrt{\bar{\alpha}_t}\, x_0,\ (1 - \bar{\alpha}_t)\, \mathcal{I}\right) \tag{4.9}$$

which describes the probability density of x_t given the clean input x_0 at any timestep t, allowing x_t to be sampled directly from a Gaussian distribution whose parameters depend on x_0 and the variance schedule coefficients $\beta_{1:T}$, without recursively iterating through all previous diffusion steps.

By combining the expression of the marginal conditional distribution $q(x_t \mid x_0)$ derived in Eq. (4.9)[1], with the forward transition distribution $q(x_t \mid x_{t-1})$ defined in Eq. (4.4), Bayes theorem (Eq. (4.10)) can be applied to derive the closed-form expression of the true Bayesian posterior $q(x_{t-1} \mid x_t, x_0)$ of the forward Markov chain (Eq. (4.11))[2]:

$$q(z \mid x) = p(x \mid z) \cdot \frac{p(z)}{p(x)} \tag{4.10}$$

$$q(x_{t-1} \mid x_t, x_0) = q(x_t \mid x_{t-1}) \cdot \frac{q(x_{t-1} \mid x_0)}{q(x_t \mid x_0)} \tag{4.11}$$

To compute the posterior, we use the identity for the product of Gaussians:

$$\mathcal{N}(x;\ \mu_1, \Sigma_1) \cdot \mathcal{N}(x;\ \mu_2, \Sigma_2) = C \cdot \mathcal{N}(x;\ \mu_{\text{out}}, \Sigma_{\text{out}}) \tag{4.12}$$

$$\begin{aligned} C &= \mathcal{N}\Big(x = \mu_1;\ \mu_2,\ (\Sigma_1 + \Sigma_2)\Big) \\ \Sigma_{\text{out}} &= \left(\Sigma_1^{-1} + \Sigma_2^{-1}\right)^{-1} \\ \mu_{\text{out}} &= \Sigma_{\text{out}} \left(\Sigma_1^{-1} \cdot \mu_1 + \Sigma_2^{-1} \cdot \mu_2\right) \end{aligned}$$

Applying this to the likelihood $q(x_t \mid x_{t-1})$ and the two marginals $q(x_{t-1} \mid x_0)$ and $q(x_t \mid x_0)$ as they appear in the Bayes theorem formulation, we obtain:

$$q(x_{t-1} \mid x_t, x_0) = C \cdot \mathcal{N}\left(x_{t-1};\ \tilde{\mu}_t(x_t, x_0),\ \tilde{\beta}_t \mathcal{I}\right) \tag{4.13}$$

with the posterior distribution parameters defined as:

[1] Evaluated at timesteps t and $t-1$ to obtain $q(x_t \mid x_0)$ and $q(x_{t-1} \mid x_0)$.

[2] In this context, the likelihood term $q(x_t \mid x_{t-1})$ is used in place of the conditional $q(x_t \mid x_{t-1}, x_0)$ because the forward process is a Markov chain. This means that each state x_t depends only on its immediate predecessor x_{t-1}, and is conditionally independent of the original data x_0, i.e., $(q(x_t \mid x_{t-1}, x_0) = q(x_t \mid x_{t-1})$.

$$C = \mathcal{N}\left(x = x_t;\, \sqrt{\bar{\alpha}_t}x_0,\, (1-\bar{\alpha}_t)\mathcal{I}\right) \quad \text{C is exactly } q(x_t \mid x_{t-1})$$

$$\tilde{\beta}_t = \frac{(1-\bar{\alpha}_{t-1})\,\beta_t}{1-\bar{\alpha}_t} \tag{4.14}$$

$$\tilde{\mu}_t(x_t, x_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\,\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t \tag{4.15}$$

Given the initial data sample x_0, the predefined variance schedule $\beta_{1:T}$ and the intermediate latent sample x_t at timestep t, the latent state x_{t-1} at timestep $t-1$ can be sampled from the true posterior as:

$$\begin{aligned} x_{t-1} = &\left(\frac{\sqrt{\bar{\alpha}_{t-1}}\,\beta_t}{1-\bar{\alpha}_t}\,x_0 + \frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\,x_t\right) \\ &+ \sqrt{\frac{(1-\bar{\alpha}_{t-1})\,\beta_t}{1-\bar{\alpha}_t}} \cdot \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \mathcal{I}) \end{aligned} \tag{4.16}$$

4.1.3 Training

While the true posterior distribution derived in Eq. (4.13) is known in closed form and can be used to sample x_{t-1} deterministically, it depends on the original clean sample x_0, which is not accessible at inference time. The training goal is therefore to train a model with learned parameters θ that is able to infer the parameters $\left(\mu_\theta(x_t, t),\, \Sigma_\theta(x_t, t)\right)$ at each timestep$t$, without access to x_0, such that the reverse transition distribution $p_\theta(x_{t-1} \mid x_t)$, defined in Eq. (4.2), closely approximates the true posterior $q(x_{t-1} \mid x_t, x_0)$.

$$\theta^* : p_{\theta^*}(x_{t-1} \mid x_t) \;\approx\; q(x_{t-1} \mid x_t, x_0). \tag{4.17}$$

The training goal can be translated into optimizing the model parameters θ to maximize the marginal likelihood $p_\theta(x_0)$ of the data under the generative model. This corresponds to making the observed data sample x_0 highly probable under the generative process, independently of the latent trajectory $x_{1:T}$.

Since it is impossible to compute the marginal likelihood as the integral in Eq. (4.1) over the entire latent space, we apply Jensen's inequality (Eq. (4.18)) to the logarithm of the marginal likelihood (Eq. (4.19)) in order to derive a logarithmic variational lower bound of the marginal likelihood to use as computable training heuristic. This heuristic is the Evidence Lower Bound, or ELBO, given in Eq. (4.20):

$$\mathbb{E}[f(X)] \leq f(\mathbb{E}[X]) \quad \forall f : f(\lambda x + (1-\lambda)y) \geq \lambda f(x) + (1-\lambda)f(y) \tag{4.18}$$

$$\begin{aligned}\log p_\theta(x_0) &= \log \int p_\theta(x_0, x_{1:T})\, dx_{1:T} \\ &= \log \int \frac{q(x_{1:T} \mid x_0)}{q(x_{1:T} \mid x_0)}\, p_\theta(x_0, x_{1:T})\, dx_{1:T} \\ &= \log \mathbb{E}_{q(x_{1:T}|x_0)} \left[\frac{p_\theta(x_0, x_{1:T})}{q(x_{1:T} \mid x_0)}\right]\end{aligned} \tag{4.19}$$

$$\log\, \mathbb{E}_{q(x_{1:T}|x_0)} \left[\frac{p_\theta(x_{0:T})}{q(x_{1:T} \mid x_0)}\right] \geq \mathbb{E}_{q(x_{1:T}|x_0)} \left[\log \frac{p_\theta(x_{0:T})}{q(x_{1:T} \mid x_0)}\right] := \textbf{ELBO} \tag{4.20}$$

The ELBO formulation in Eq. (4.20) depends only on probability distributions that have already been defined in closed form: the forward process estimated posterior in Eq. (4.5) and the reverse process joint distribution in Eq. (4.3).

The training goal therefore becomes minimizing the negative ELBO, which provides an upper bound to the expected value of the negative log-likelihood of the data [37].

$$\mathcal{L} := \mathbb{E}_\theta \left[-\log p_\theta(x_0)\right] \leq \mathbb{E}_{q(x_{1:T}|x_0)} \left[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T} \mid x_0)}\right] \tag{4.21}$$

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_{q(x_{1:T}|x_0)} \left[-\log \frac{p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t)}{\prod_{t=1}^{T} q\left(x_t \mid x_{t-1}\right)}\right] \\ &= \mathbb{E}_{q(x_{1:T}|x_0)} \left[-\log p(x_T) - \log \frac{p_\theta(x_0 \mid x_1)}{q(x_1 \mid x_0)} - \log \prod_{t=2}^{T} \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_t \mid x_{t-1})}\right]\end{aligned} \tag{4.22}$$

By applying Bayes theorem (Eq. (4.10)), the single-step forward transition distribution in Eq. (4.4) can be rewritten in terms of the closed-form true posterior (Eq. (4.13)) and the marginals $q(x_t \mid x_0)$ and $q(x_{t-1} \mid x_0)$ as:

$$q(x_t \mid x_{t-1}) = q(x_{t-1} \mid x_t, x_0) \cdot \frac{q(x_t \mid x_0)}{q(x_{t-1} \mid x_0)}. \tag{4.23}$$

The logarithm of the products can therefore be expanded as:

$$\begin{aligned}\log \prod_{t=2}^{T} \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_t \mid x_{t-1})} &= \log \prod_{t=2}^{T} \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_{t-1} \mid x_t, x_0)} \cdot \frac{q(x_{t-1} \mid x_0)}{q(x_t \mid x_0)} \\ &= \log \prod_{t=2}^{T} \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_{t-1} \mid x_t, x_0)} + \log \prod_{t=2}^{T} \frac{q(x_{t-1} \mid x_0)}{q(x_t \mid x_0)} \\ &= \log \prod_{t=2}^{T} \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_{t-1} \mid x_t, x_0)} + \log \frac{q(x_1 \mid x_0)}{q(x_T \mid x_0)}\end{aligned} \tag{4.24}$$

The loss function can be expanded as a sum of logarithms equal to the logarithm of the products and regrouped as follows:

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_{q(x_{1:T}|x_0)}\left[-\log p(x_T) - \log\frac{p_\theta(x_0 \mid x_1)}{q(x_1 \mid x_0)} - \log\prod_{t=2}^{T}\frac{p_\theta(x_{t-1} \mid x_t)}{q(x_t \mid x_{t-1})}\right], \\ &= \mathbb{E}_{q(x_{1:T}|x_0)}\left[-\log p_\theta(x_0 \mid x_1) - \sum_{t=2}^{T}\log\frac{p_\theta(x_{t-1} \mid x_t)}{q(x_{t-1} \mid x_t, x_0)} - \log\frac{p(x_T)}{q(x_T \mid x_0)}\right] \quad (4.25)\end{aligned}$$

The parametric loss function can be rewritten using the Kullback-Leibler divergence $\mathcal{D}_{\mathrm{KL}}$, which is a measure of the discrepancy between two probability distributions $q(x)$ and $p(x)$, and is defined as:

$$\begin{aligned}\mathcal{D}_{\mathrm{KL}}\Big(q(x) \,\|\, p(x)\Big) &= \mathbb{E}_{q(x)}\left[\log\frac{q(x)}{p(x)}\right] \quad (4.26)\\ &= \frac{1}{2}\left[k\left(\frac{\sigma_1^2}{\sigma_2^2} - 1 - \log\frac{\sigma_1^2}{\sigma_2^2}\right) + \frac{1}{\sigma_2^2}\left\|\mu_1 - \mu_2\right\|^2\right] \quad (4.27)\end{aligned}$$

Using this definition, the loss can be decomposed as the sum of three contributions: [37]

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_{q(x_{1:T}|x_0)}\left[-\log p_\theta(x_0 \mid x_1) + \sum_{t=2}^{T}\log\frac{q(x_{t-1} \mid x_t, x_0)}{p_\theta(x_{t-1} \mid x_t)} + \log\frac{q(x_T \mid x_0)}{p(x_T)}\right] \\ &= \mathcal{L}_0 + \mathcal{L}_{t-1} + \mathcal{L}_T \quad (4.28)\end{aligned}$$

where:

$\mathcal{L}_0$ is the **reconstruction term**, corresponding to the negative log-likelihood of the original data sample x_0 conditioned on the first latent x_1:

$$\mathcal{L}_0 = \mathbb{E}_{q(x_{1:T}|x_0)}\left[-\log p_\theta(x_0 \mid x_1)\right] \quad (4.29)$$

$\mathcal{L}_0$ **depends on the trainable parameters** θ through the conditional distribution $p_\theta(x_0 \mid x_1)$, and its role is similar to that of a **decoder**, mapping latent representations back to the original sample space.

$\mathcal{L}_{t-1}$ is the **regularization term**, defined as the sum of Kullback-Leibler divergences between the true posterior and the learned reverse transition distribution at each intermediate timestep:

$$\mathcal{L}_{t-1} = \sum_{t=2}^{T}\mathcal{D}_{\mathrm{KL}}\Big(q(x_{t-1} \mid x_t, x_0) \,\|\, p_\theta(x_{t-1} \mid x_t)\Big) \quad (4.30)$$

$\mathcal{L}_{t-1}$ **depends on the trainable parameters** θ and represent the agreement between the parametric reverse transition distribution $p_\theta(x_{t-1} \mid x_t)$ and the true posterior $(q(x_{t-1} \mid x_t, x_0)$. Its optimization $\theta^* = \arg\min_\theta \mathcal{D}_{\mathrm{KL}}\Big(q(x_{t-1} \mid x_t, x_0) \,\|\, p_\theta(x_{t-1} \mid x_t)\Big)$ fulfills the training objective defined in Eq. (4.17)

$\mathcal{L}_T$ is the **prior-matching term**, which enforces consistency between the terminal forward distribution and the prior $p(x_T)$:

$$\mathcal{L}_T = \mathcal{D}_{\mathrm{KL}}\Big(q(x_T \mid x_0) \,\|\, p(x_T)\Big) \tag{4.31}$$

$\mathcal{L}_T$ **does not depend on** θ, since both $q(x_T \mid x_0)$ and the prior $p(x_T) = \mathcal{N}(0, I)$ are fixed, and can be ignored during training.

4.1.4 Simplified training

A simplified training objective can be defined by focusing on minimizing $\mathcal{L}_{t-1}$, rather than optimizing the full set of components $\mathcal{L}_0$, $\mathcal{L}_{t-1}$, $\mathcal{L}_T$.

As stated before in Eq. (4.17), the training aims is to learn a reverse transition distribution $p_\theta(x_{t-1} \mid x_t) := \mathcal{N}(x_{t-1};\, \mu_\theta(x_t, t),\, \Sigma_\theta(x_t, t))$ (Eq. (4.2)), that closely matches the true posterior $q(x_{t-1} \mid x_t, x_0) := \mathcal{N}\Big(x_{t-1};\, \tilde{\mu}_t(x_t, x_0),\, \tilde{\beta}_t \mathbf{I}\Big)$ (Eq.(4.13)).

Firstly, $\Sigma_\theta(x_t, t)$ is set to match the true posterior variance $\tilde{\beta}_t$ (Eq. (4.14)) at timestep t:

$$\Sigma_\theta(x_t, t) = \tilde{\beta}_t \mathcal{I} = \frac{(1 - \bar{\alpha}_{t-1})\,\beta_t}{1 - \bar{\alpha}_t} \tag{4.32}$$

This means that, having defined a variance schedule $\beta_{1:T}$ for the forward process, the corresponding posterior variances $\tilde{\beta}_{1:T}$ are obtained in closed form, mirroring the schedule structure of $\beta_{1:T}$. They are not learned and depend only on the timestep t, since each $\tilde{\beta}_t$ is directly determined by β_t through Eq. (4.32).

For two distributions with the same variance Eq. (4.27) becomes:

$$\mathcal{D}_{\mathrm{KL}}\Big(q(x) \,\|\, p(x)\Big) = \frac{1}{2}\left[\frac{1}{\sigma_2^2}\,\|\mu_1 - \mu_2\|^2\right] \tag{4.33}$$

The $\mathcal{L}_{t-1}$ (Eq.(4.30)) loss can therefore be written as

$$\mathcal{L}_{t-1} = \sum_{t=2}^{T} \mathbb{E}_q\left[\frac{1}{2\tilde{\beta}_t}\left\|\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t)\right\|^2\right] \tag{4.34}$$

According to this parameterization, a model $\mu_\theta(x_t, t)$ with parameters θ could be trained to predict the true posterior mean $\tilde{\mu}_t(x_t, x_0)$ from x_t by minimizing $\mathcal{L}_{t-1}$.

An alternative parametrization can be obtained by constructing an unbiased estimator of $\mathcal{L}_{t-1}$. This is done by uniformly sampling a single random timestep $t \sim \text{Uniform}(1, T)$, and generating x_t according to Eq. (4.8), which defines x_t as a function of the clean sample x_0 and a noise sample $\varepsilon \sim \mathcal{N}(0, \mathcal{I})$.

The true posterior mean $\tilde{\mu}_t(x_t, x_0)$ (Eq. (4.15)) can be rewritten as a function of $x_t(x_0, \varepsilon)$:

$$\begin{aligned}
\tilde{\mu}_t(x_t, x_0) &= \tilde{\mu}_t\Big(x_t(x_0,\varepsilon),\ \tfrac{1}{\sqrt{\bar{\alpha}_t}}\Big(x_t(x_0,\varepsilon) - \sqrt{1-\bar{\alpha}_t}\,\varepsilon\Big)\Big) \\
&= \frac{\sqrt{\bar{\alpha}_{t-1}}\,\beta_t}{1-\bar{\alpha}_t}\ \tfrac{1}{\sqrt{\bar{\alpha}_t}}\Big(x_t(x_0,\varepsilon) - \sqrt{1-\bar{\alpha}_t}\,\varepsilon\Big)\ +\ \frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\ x_t(x_0,\varepsilon) \\
&= \left[\frac{\beta_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}} + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\right] x_t(x_0,\varepsilon) - \frac{\beta_t}{\sqrt{\alpha_t}\sqrt{1-\bar{\alpha}_t}}\,\varepsilon \\
&= \frac{1}{\sqrt{\alpha_t}}\ x_t(x_0,\varepsilon) - \frac{\beta_t}{\sqrt{\alpha_t}\sqrt{1-\bar{\alpha}_t}}\,\varepsilon \\
&= \frac{1}{\sqrt{\alpha_t}}\left(x_t(x_0,\varepsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\varepsilon\right)
\end{aligned} \tag{4.35}$$

The loss function can also be rewritten as a function of $x_t(x_0, \varepsilon)$ (up to a constant C that is independent of the parameters θ):

$$\mathcal{L}_{t-1} - C = \mathbb{E}_{x_0,\varepsilon}\left[\frac{1}{2\tilde{\beta}_t}\left\|\frac{1}{\sqrt{\alpha_t}}\left(x_t(x_0,\varepsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\varepsilon\right) - \mu_\theta\Big(x_t(x_0,\varepsilon), t\Big)\right\|^2\right] \tag{4.36}$$

Instead of training $\mu_\theta(x_t, t)$ to match the forward process posterior mean $\tilde{\mu}_t$, the estimator μ_θ can be parameterized to match the formulation of $\tilde{\mu}_t(x_t, x_0)$:

$$\begin{aligned}
\tilde{\mu}_\theta(x_t, t) &= \tilde{\mu}_\theta\Big(x_t(x_0,\varepsilon),\ \tfrac{1}{\sqrt{\bar{\alpha}_t}}\Big(x_t(x_0,\varepsilon) - \sqrt{1-\bar{\alpha}_t}\,\varepsilon_\theta(x_t(x_0,\varepsilon), t)\Big)\Big) \\
&= \frac{1}{\sqrt{\alpha_t}}\left(x_t(x_0,\varepsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\varepsilon_\theta(x_t(x_0,\varepsilon), t)\right)
\end{aligned} \tag{4.37}$$

The newly parameterized estimator can be substituted into Eq. (4.36) to obtain

$$\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left[-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\varepsilon + \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\varepsilon_\theta(x_t(x_0,\varepsilon), t)\right] \tag{4.38}$$

$$= \frac{-\beta_t}{\sqrt{\alpha_t}\,\sqrt{1-\bar{\alpha}_t}}\,(\varepsilon - \varepsilon_\theta(x_t(x_0,\varepsilon), t)) \tag{4.39}$$

Therefore,

$$\|\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t)\|^2 = \frac{\beta_t^2}{\alpha_t\,(1-\bar{\alpha}_t)}\,\|\varepsilon_\theta(x_t, t) - \varepsilon\|^2 \tag{4.40}$$

and the loss becomes

$$\mathcal{L}_{t-1} - C = \mathbb{E}_{x_0,\,\varepsilon}\left[\frac{1}{2\tilde{\beta}_t}\,\frac{\beta_t^2}{\alpha_t\,(1-\bar{\alpha}_t)}\,\|\varepsilon_\theta(x_t(x_0,\varepsilon), t) - \varepsilon\|^2\right] \tag{4.41}$$

$$= \mathbb{E}_{x_0,\,\varepsilon}\left[\frac{\beta_t^2}{2\,\tilde{\beta}_t\,\alpha_t\,(1-\bar{\alpha}_t)}\,\left\|\varepsilon - \varepsilon_\theta\Big(\sqrt{\bar{\alpha}_t}\,x_0 + \sqrt{1-\bar{\alpha}_t}\,\varepsilon,\; t\Big)\right\|^2\right] \tag{4.42}$$

In this parameterization, a model $\varepsilon_\theta(x_t(x_0,\varepsilon), t)$ with parameters θ is trained to predict the noise sample ε from $x_t(x_0,\varepsilon)$. With t, x_0, ε known during training. The loss function in Eq. (4.42) is equivalent to the mean squared error (MSE) between the true noise sample ε and the model prediction $\varepsilon_\theta(x_t, t)$, scaled by a timestep-dependent constant factor $\frac{\beta_t^2}{2\,\tilde{\beta}_t\,\alpha_t\,(1-\bar{\alpha}_t)}$. In practice, the constant weighting factor in Eq. (4.42) can be omitted, leading to the simplified objective [37]:

$$\mathcal{L}_{\text{simple}}(\theta) := \mathbb{E}_{t,\,x_0,\,\varepsilon}\left[\left\|\varepsilon - \varepsilon_\theta\Big(\sqrt{\bar{\alpha}_t}\,x_0 + \sqrt{1-\bar{\alpha}_t}\,\varepsilon,\; t\Big)\right\|^2\right] \tag{4.43}$$

which corresponds to minimizing the mean squared error between the true noise sample ε and the model prediction ε_θ at randomly chosen timesteps t uniformly sampled between 1 and T.

A prediction of the intermediate denoised step x_{t-1} can be sampled from the learned reverse transition distribution $p_\theta(x_{t-1} \mid x_t)$ by computing

$$\hat{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\varepsilon_\theta(x_t, t)\right) + \sqrt{\tilde{\beta}_t}z \qquad z \sim \mathcal{N}(0, I) \tag{4.44}$$

The additional term $\sqrt{\tilde{\beta}_t}$ z represents **sampling noise**, introduced to model the stochasticity of the reverse diffusion process as a Gaussian process so that $p_\theta(x_{t-1} \mid x_t)$ remains a Gaussian distribution.

4.2 Variational Autoencoders

A Variational Autoencoder (VAE) [38, 40] is a statistical model composed of a matched encoder decoder pair (E_ϕ, D_θ) that learns a lower dimensional representation of data in a latent space. The encoder E_ϕ maps the data samples $x \in \mathbb{R}^{H\times W\times C}$ into a latent space representation $z \in \mathbb{R}^{H_L\times W_L\times C_L}$, while the decoder D_θ reconstructs the data from the latent variables, i.e. $\hat{x} = \mathbf{B}(z)$.

4.2.1 Encoding process

The encoders learns a gaussian approximate posterior [40]:

$$q_\phi(z \mid x) = \mathcal{N}\Big(z; \mu_\phi, {\sigma_\phi}^2 \mathcal{I}\Big) \tag{4.45}$$

parametrized by neural networks with parametersϕ predicting the mean μ_ϕ and the variance ${\sigma_\phi}^2$. A latent representation z of the datasample x can be sampled from $q_\phi(z \mid x)$ by reparameterization:

$$z = \mu_\phi(x) + \sigma_\phi(x)\varepsilon \quad \varepsilon \sim \mathcal{N}(0, I) \tag{4.46}$$

Equation (4.46) express z as a function of the estimated mean μ_ϕ and variance σ_ϕ^2 and a random sample ε.

4.2.2 Decoding process

The decoder learns a conditional likelihood [40]:

$$p_\theta(z \mid x) = \mathcal{N}\Big(z; \mu_\theta, {\sigma_\theta}^2 \mathcal{I}\Big) \tag{4.47}$$

parametrized by neural networks with parameters θ predicting the mean μ_θ and the variance ${\sigma_\theta}^2$.

4.2.3 Training

The logarithm of the marginal likelihood is the sum of the log likelihood of each sample, which is generally intractable:

$$\log p_\theta(x_{1:N}) = \sum_{i=1}^{N} \log p_\theta(x^{(i)}) \tag{4.48}$$

$$p_\theta(x^{(i)}) = \int p_\theta(z)\, p_\theta(x^{(i)} \mid z)\, dz \tag{4.49}$$

Applying Jensen's inequality (Eq. (4.18)) and using Bayes theorem (Eq. (4.10)) to derive the true posterior for z, the logarithm of the marginal likelihood of datapoint $x^{(i)}$ becomes

$$\log p_\theta\Big(x^{(i)}\Big) = \underbrace{\mathbb{E}_{q_\phi(z|x^{(i)})}\Big[\log p_\theta\Big(x^{(i)}, z\Big) - \log q_\phi\Big(z \mid x^{(i)}\Big)\Big]}_{\mathcal{L}(\theta, \phi; x^{(i)})} \tag{4.50}$$

$$+ \; D_{\mathrm{KL}}\Big(q_\phi(z \mid x^{(i)}) \,\Big\|\, p_\theta(z \mid x^{(i)})\Big) \tag{4.51}$$

Since the D_{KL} is a non negative quantity, the variational lower bound of the log likelihood can be quantified as [40]:

$$\mathcal{L}(\theta, \phi; x^{(i)}) = \mathbb{E}_{q_\phi(z|x^{(i)})}\Big[\log p_\theta(x^{(i)} \mid z)\Big] - D_{\mathrm{KL}}\Big(q_\phi(z \mid x^{(i)}) \,\|\, p_\theta(z)\Big) \tag{4.52}$$

The training objective becomes minimizing the negative lower bound:

$$-\mathcal{L}(\theta, \phi; x^{(i)}) = -\,\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x \mid z)] + D_{\mathrm{KL}}\Big(q_\phi(z \mid x) \,\|\, p_\theta(z)\Big) \tag{4.53}$$

The prior distribution of z is set to $p_\theta(z) = \mathcal{N}(0, \mathcal{I})$ to fit the latent space representation to a standard normal distribution. Having parametrized the posterior $q_\phi(z \mid x)$ (Eq. (4.45)) to a gaussian distribution, and having sampled z according to Eq.(4.46), the Variational Autoencoder loss $\mathcal{L}_{\mathrm{VAE}}$ becomes [40]:

$$\begin{aligned} \mathcal{L}_{\mathrm{VAE}}(x^{(i)}) \;=\; -\,\mathcal{L}(\theta, \phi; x^{(i)}) \;\approx\; & \underbrace{\frac{1}{2}\sum_{j=1}^{J}\Big((\mu_j^{(i)})^2 + (\sigma_j^{(i)})^2 - \log(\sigma_j^{(i)})^2 - 1\Big)}_{\mathrm{KL}\left(q_\phi(z|x^{(i)}) \,\|\, \mathcal{N}(0,I)\right)} \\ & - \underbrace{\frac{1}{L}\sum_{l=1}^{L}\log p_\theta\Big(x^{(i)} \mid z^{(i,l)}\Big)}_{\text{negative expected log-likelihood}} \end{aligned} \tag{4.54}$$

Chapter 5

Methods

5.1 Image dataset and preprocessing

All images used for this thesis were previously collected at the VUB Rehabilitation Research Center (RERE) [16] in Jette, Belgium. Eight able-bodied young adults (four females and four males) aged 18 to 25 years were selected as test subjects. Each subject was asked to perform four simple exercises, executed at a slow pace of 20 bpm [17]. Exercise descriptions are displayed in Table 5.1.

SAB	Bilateral shoulder abduction
KFB	Squat (Knee flexion)
HABR	Right hip abduction with right arm abduction
HABL	Left hip abduction with left arm abduction

Table 5.1: Exercises performed by the test subjects

Two acquisitions were performed for each exercise, one acquired using only the Azure Kinect device and without markers and the other acquired during a concurrent acquisition with the Vicon system, with marker applied on the subject's body. For the concurrent acquisition a total of 49 markers were placed by undergraduate Physiotherapy students in relevant positions of the subject's body: 39 placed accordingly to the conventional Optitrack marker set positions, 4 wand markers on the Tight and the Tibia, and 6 additional markers following the Vicon upper-body marker set.

The experimental setup (fig. 5.1) involved a Azure Kinect RGBD camera with

ToF depth estimation technology placed 200 cm in front of the subject and 130 cm from the ground. Concurrent acquisitions also involved 14 Vicon Vero cameras evenly spaced around the subject for motion capture, and 2 Vicon Vue cameras capturing high-definition images of the subject in the frontal and sagittal plane. Both Vicon Vero and Vicon Vue cameras were set to a 100 Hz acquisition frame-rate.

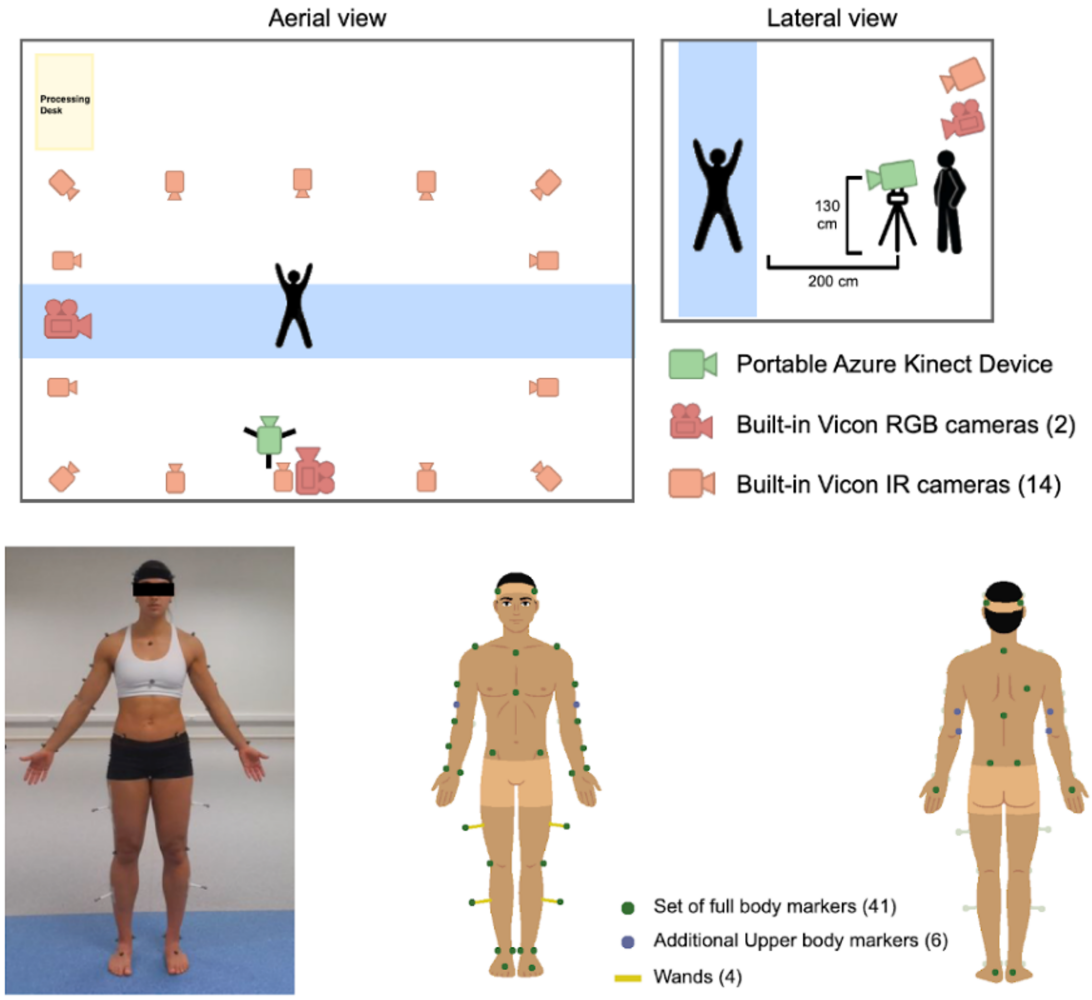

Figure 5.1: Experimental setup: aerial and lateral view of the lab with the 14+2 Vicon system cameras, marker placement. Courtesy of the RERE Center (VUB).

5.1.1 Preprocessing steps

A series of preprocessing steps has been implemented (Fig. 5.2.a), aiming to obtain a set of clean images for training the denoising model.

The main focus of preprocessing was to separate the subject from the background and to optimize the dynamic range of the images.

Step 1: **Thresholding**
The pixel values of the depth image (Fig. 5.2.b) have been thresholded between the values of 1,000 mm and 4,000 mm to remove non-zero pixels that were too close to the camera and objects in the background. This step was necessary to avoid the presence of confusing elements during the training process of the diffusion model. Pixels outside the aforementioned interval, as well as clusters of pixels too small to belong to the subject, were set to zero and assigned to the background.

Step 2: **Zeroing out Rows and Columns outside the Subject's silhouette**
The image intensity histograms along the x and y axes were computed by summing the intensity of the pixels across the columns and the rows, respectively. These histograms were used to estimate the bounding box of the subject's silhouette. Pixels outside the silhouette of the subject were zeroed out to reduce the background noise (Fig. 5.2.c).

Step 3: **Z-Score Normalization and Intensity scaling**
The depth values, originally measured in millimeters, were normalized to dimensionless values using the Z-score normalization, according o the formula:

$$Z_{score} = \frac{x - \mu}{\sigma} \tag{5.1}$$

The Z-score values were then clipped to a range going from -2 to +2 and finally scaled to the [0, 1] interval to match the expected input format of the Python libraries used for training the models (Fig. 5.2.d).

Step 4: **ROI identification and extraction**
The center of the Region of Interest has been identified as the pixel whose row index is the center value of all rows containing non-zero pixels, and whose column index is the center value of all columns containing non-zero pixels. A square ROI large enough to contain all the pixels that belong to the subject was extracted from the image and then resized to a fixed 256×256 size (Fig. 5.2.e).

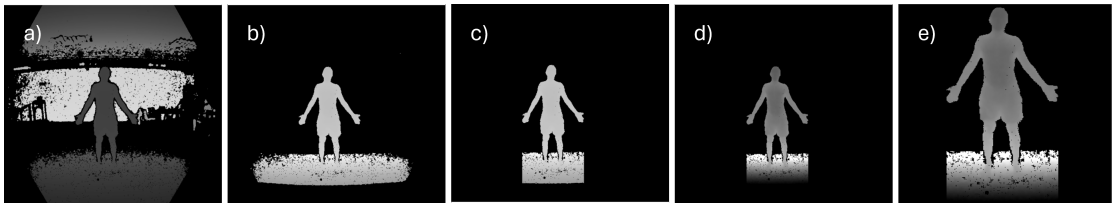

Figure 5.2: Preprocessing steps: **a)**Original Image **b)**Background removed applying a threshold **c)**Silhouette of the subject, zeroed out lateral columns **d)**Intensity range Normalization **e)**ROI Extraction

Exclusion Criteria

Some of the images presented a higher number of missing pixel values, falling below the quality level we considered necessary for training an image-based generative model. For that reason we implemented an exclusion criterion computed as the sum of the pixel values divided by the total number of pixels.

$$R = \frac{\Sigma_{i \in rows} \Sigma_{j \in columns} [I_{i,j}]}{n_{rows} \times n_{columns}} \tag{5.2}$$

Outlier images, whose ratio fell below the distribution of the other pictures in the dataset, were excluded from the construction set. We concluded that in the Kinect-only, marker-less acquisition this affected 3.36% of the images. In the concurrent Vicon + Kinect acquisitions, where markers were applied on the subject, this percentage rose to 24.58% of the images.

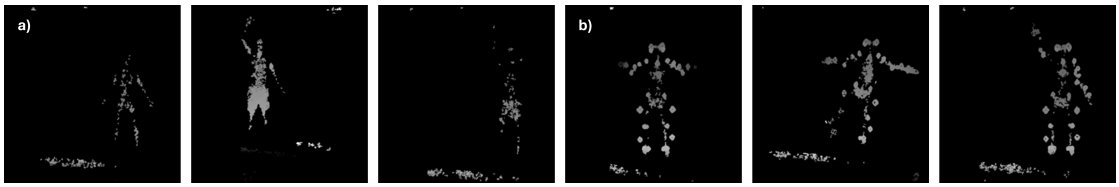

Figure 5.3: Excluded images **a)** Images without markers excluded from the training set **b)** Images with markers excluded from the test set

5.1.2 Dataset partitioning into Construction Set and Test Set

Six of the original eight subjects were included in the Construction Set (three males and three females), while the remaining two (one male and one female) were used as a test set. For each of the training sessions, 80% of the images of each subject in

the construction set were used as the training set, while the other 20% were used as a validation set during the training process.

Construction Set	6 Subjects	26 103 Images	80% Training set 20% Validation set	20 882 Images 5 221 Images
Test Set	2 Subjects	10 667 Images		

Table 5.2: Dataset partitioning into construction and test sets.

5.2 AI denoising framework development

5.2.1 Model architecture

The architecture proposed [38, 20] comprises an autoencoder, consisting of an encoder designed to produce a reduced latent-space representation of the image and a symmetric decoder capable of reverting back to the original image from its latent representation, and a diffusion model, which has been trained to iteratively remove noise from the depth images, also operating in the latent space.

The autoencoder is implemented as a three-layer neural network with 64, 128, and 256 channels respectively that accepts a $1 \times 256 \times 256$ grayscale depth image as input and outputs a $3 \times 64 \times 64$ three-channels latent space representation. The down-sampling is obtained with strided convolution and group normalization is performed at each layer [41].

The diffusion model is implemented as a Monai Diffusion U-Net model: it receives the $3 \times 64 \times 64$ autoencoder output with added random Gaussian noise pattern of the same size, processes it through a contracting path of two convolutional layers of 64 and 128 channels for feature extraction, processes it through a 128 channel bottleneck to predict the added noise contribution to the noised image, and then up-sample it through a symmetric diverging path to obtain a $3 \times 64 \times 64$ prediction of the noise pattern [19].

5.2.2 Variational Autoencoder training

The autoencoder was trained for 400 epochs employing an adversarial strategy in which the autoencoder was used as an image generator and paired with a discriminator concurrently trained to distinguish images from the original training set from images reconstructed using only their latent-space representation. This strategy was adopted to ensure meaningful feature selection and minimize loss of information during the encoding process.

For each iteration, a batch $\hat{B}$ of 32 preprocessed grayscale depth images was fed to the model to extract an output consisting of:

B: The output of the decoder branch of the autoencoder U-net, consisting in a set of reconstructed images obtained from the latent space representation of the original batch $\hat{B}$ of images.

μ: The mean value of the latent space distribution.

σ: The standard deviation of the latent space distribution.

The generator loss was computed as a linear combination of four terms, each one representing a different performance parameter [38, 20, 41]:

$$\mathcal{L}^G = \mathcal{L}_1 + w_1 \cdot KL_{div} + w_2 \cdot \mathcal{L}^G_{\text{perc}} + w_3 \cdot \mathcal{L}^G_{\text{adv}} \tag{5.3}$$

where:

$\mathcal{L}_1$ is the pixel-wise L_1 loss computed as the mean absolute error between the pixel intensity values of B and $\hat{B}$

$$L_1 \;=\; \frac{1}{B_S \times m \times n} \sum_{b=1}^{B_S} \sum_{i=1}^{m} \sum_{j=1}^{n} \left| \mathcal{I}^{\in B}_{b,i,j} - \mathcal{I}^{\in \hat{B}}_{b,i,j} \right| \quad \text{with } B_S = 32 \text{ (batch size)}$$

KL_{div} is the Kullback-Leibler divergence that measures the deviation of the latent space distribution from a zero-mean, unit-variance normal distribution

$$\begin{aligned} \mathrm{D_K L}(q(z|x) \,\|\, \mathcal{N}(0,I)) &= \int q(z|x) \log\left(\frac{q(z|x)}{\mathcal{N}(0,I)}\right) dz \\ &= \mathbb{E}_{z\sim q(z|x)} \left[\log q(z|x) - \log \mathcal{N}(0,I)\right] \\ &= \frac{1}{2}\Big(\|\mu\|^2 + \mathrm{Tr}(\Sigma) - \log\det\Sigma - d\Big) \end{aligned}$$

$\mathcal{L}^G_{\text{perc}}$ is the generator perceptual loss, that measure the perceptual differences between the original and the reconstructed image. It is evaluated using a pretrained AlexNet convolutional neural network for feature extraction. It is computed as the mean square error (MSE) between the feature map extracted from the original images and the feature map extracted from the reconstructed images:

$$\mathcal{L}_{\text{perceptual}} = \sum_{l=1}^{L} \frac{1}{N_l} \left\| \phi_l(x_{\text{recon}}) - \phi_l(x_{\text{true}}) \right\|^2 \quad (N_l\text{: features in layer } l)$$

$\mathcal{L}_{adv}^{G}$ is the generator adversarial loss, that measures how well the generator can fool the discriminator by producing synthetic images that are indistinguishable from real ones (False Positives). It is computed as the Binary Cross-Entropy (BCE) between the discriminator's output on synthetic samples B and a target label of 1:

$$\mathcal{L}_{\text{adv}}^{G} = \text{BCE}(D(B), 1) = -\log(D(B))$$

w_1, w_2, w_3 are the scalar weights that balance the contribution of each metric to the total loss

On the other hand, the discriminator loss was computed as the mean of two adversarial loss terms. The first is the Binary Cross-Entropy (BCE) between the discriminator's output on real images $\hat{B}$ and a target label of 1; this parameter measures how well the discriminator classifies real images as real. The second is the BCE between the discriminator's output on synthetic images B and a target label of 0; this parameter measures how well the discriminator recognizes fake images as fake.

$$\begin{aligned}
\mathcal{L}^{D} &= \frac{1}{2}\left(\mathcal{L}_{D}^{\text{real}} + \mathcal{L}_{D}^{\text{fake}}\right) \\
&= \frac{1}{2}\Big(\text{BCE}(D(\hat{B}), 1) + \text{BCE}(D(B), 0)\Big) \\
&= \frac{1}{2}\Big(-\log(D(\hat{B})) - \log(1 - D(B))\Big)
\end{aligned}$$

The generator and discriminator losses were then backpropagated by computing the gradients of each loss with respect to their corresponding network trainable parameters. During the generator update, the discriminator's weights were kept constant, and vice versa. These gradients were then used to update the model weights using the Adam optimizer.

Every 5 epochs a validation step was performed, in which the performances of the current iteration of the model was tested by computing the generator loss on the validation set. If the validation loss obtained during any validation check happened to be lower than any previously recorded value, the corresponding state of the model was saved in memory.

After the training was completed, the model training state corresponding to the checkpoint on which the lowest validation loss was achieved, was serialized in a JSON dictionary to be saved to disk.

5.2.3 Denoising Diffusion Probabilistic Model training

The Denoising Diffusion Probabilistic Model (DDPM) was trained for 600 epochs on batches of 16 images. At each iteration, Gaussian noise patterns matching the encoded images' spatial dimensions were generated by sampling random values from a standard normal distribution with zero mean and unit variance. These noise patterns were used to degrade the original encoded images simulating a diffusion process that follows the timestep dependent formula (Eq. (4.8)) described in paragraph 4.1.2:

$$X_t = \sqrt{\bar{\alpha}_t} \cdot X_0 + \sqrt{1 - \bar{\alpha}_t} \cdot \varepsilon_n, \quad \varepsilon_n \sim \mathcal{N}(0, I)$$

In this case the timesteps t used to compute the cumulative product of the corresponding noise schedule coefficients $\bar{\alpha}_t = \prod_{s=1}^{t}(1 - \beta_s)$ (where $\beta_s \in (0, 1)$ are the values of the noise variance schedule) were uniformly sampled between 0 and the maximum number T_{max} of denoising steps we want the model to be able to handle. The term ε_n is the Gaussian noise pattern generated at the current iteration of the training process.

The actual training step consisted of feeding the degraded images, the corresponding known noise patterns, and the timesteps into the diffusion model inferer to obtain the model output, consisting of a prediction $\hat{\varepsilon}_n$ of the original noise pattens ε_n.

The loss is then computed as the Mean absolute Error between the known noise patterns and the model's prediction, averaged on whole the batch [37, 19]:

$$\text{MSE} = \frac{1}{B_s \times n \times m} \sum_{b=1}^{B_s} \sum_{i=1}^{n} \sum_{j=1}^{m} (\varepsilon_{b,i,j} - \hat{\varepsilon}_{b,i,j})^2 \quad \text{with } B_S = 16 \text{ (batch size)}$$

The loss was finally backpropagated through the network to compute gradients, which were then used to update the model parameters via the Adam optimizer. Validation checks were performed every 5 epochs, and the checkpoint with the lowest validation loss was saved to disk.

5.2.4 Guided Denoising Probabilistic Diffusion Model inference for inpainting missing pixels

The trained Denoising Diffusion Probabilistic Model is capable of transforming random noise into meaningful synthetic images that resemble those included in the training set. This means that, by feeding a random noise pattern of the appropriate size into the model, the output will be a depth image depicting a subject performing one of the four exercises represented in the training set. The specific exercise depicted is not predetermined either, but emerges from the stochastic nature of the generative process and the random noise input instead [38, 20].

In contrast, our goal is to perform inpainting on images affected by artifacts caused by retro-reflective markers, characterized by the presence of blob-like clusters of missing pixel values. For that reason, we need to be able to control the generation process, guiding the model to infer only the missing data using the undamaged areas of the image as fixed boundary conditions without overwriting them.

The algorithm used to perform inpainting is based on the assumption that each image to be inpainted can be partitioned into two distinct and complementary subsets: the first containing all and only the known values of the pixels that are unaffected by artifacts, and the second containing all and only the pixels with unknown values [19, 42].

These areas can be identified by a binary mask M of the same size as the image, where pixels corresponding to the undamaged regions are marked as 1, and those corresponding to the missing areas to be inpainted are marked as 0.

Denoting by P the float matrix representation of the image pixels and by M the binary mask, we can define the matrices P_K and P_U as the subsets of known and unknown pixels, respectively.

$$P_K = P \odot M \tag{5.4}$$

$$P_U = P \odot (1 - M) \tag{5.5}$$

The inpainting algorithm proceeds as follows:

Step 1: Define a diffusion schedule by generating a sequence of β_t coefficients for $t = 1, \ldots, T_{max}$, where each $\beta_t \in (0, 1)$ controls the amount of noise added at diffusion step t.

Step 2: Initialize a random matrix P_I with the same size as the image P by sampling a random Gaussian noise distribution $\varepsilon_u \sim \mathcal{N}(0, I)$. This matrix will be the template upon which the inpainted image will be generated.

Step 3: Feed P_I as input to the diffusion model to obtain an estimate of ε_u at timestep t and to perform a small denoising step.

The model assumes a noise variance consistent with the cumulative product of the forward process noise schedule at timestep t, defined as $\bar{\alpha}_t = \prod_{s=1}^{t}(1 - \beta_s)$. Using this, the model estimates the noise as:

$$\hat{\varepsilon}_u^{(t)} = \text{DDPM}(P_I^{(t)}, t) \quad \varepsilon_u \sim \mathcal{N}(0, I)$$

This prediction is then used to compute the intermediate synthetic image:

$$\begin{aligned} \hat{P}_U^{(t-1)} &= \mu_t(P_I^{(t)}, t) + \sigma_t \cdot \varepsilon_s, \quad \varepsilon_s \sim \mathcal{N}(0, I) \\ &= \frac{1}{\sqrt{\alpha_t}} \left(P_I^{(t)} - \left(\frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \right) \cdot \hat{\varepsilon}_u \left(P_I^{(t)}, t \right) \right) + \sqrt{\left(\frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \cdot \beta_t \right)} \cdot \varepsilon_s, \end{aligned}$$

In this formula, the first term represents the predicted pixel values after partially removing noise from the current inpainted image, while the second term introduces sampling noise to account for the uncertainty resulting from the use of the estimated $\hat{\varepsilon}_u(P_I^{(t)}, t)$, given that the true noise value $\varepsilon_u(P_I^{(t)}, t)$ is unknown. This is because the observed noisy image, originally initialized as random noise, is assumed to be the outcome of an unknown forward diffusion process.

Step 4: Degrade the known pixel image P_K by adding an amount of noise corresponding to the timestep t-1 of the scheduler, as defined by the time-dependent scheduler equation:

$$\hat{P_K}^{(t-1)} = \sqrt{\bar{\alpha}_t} \cdot P_K + \sqrt{1 - \bar{\alpha}_t} \cdot \varepsilon_k, \quad \varepsilon_k \sim \mathcal{N}(0, I)$$

where $\bar{\alpha}_t = \prod_{s=1}^{t}(1 - \beta_s)$ represents the cumulative product of the noise schedule coefficients from 0 to t, $\beta_s \in (0, 1)$ are the values of the noise variance schedule, and ε_k is sampled from standard Gaussian noise.

Step 5: Combine the two intermediate results using the known pixel mask M to obtain the inpainted image at step $t - 1$ as follows:

$$P_I^{(t-1)} = P_K^{(t-1)} \odot M + P_U^{(t-1)} \odot (1 - M)$$

Step 6: Reiterate steps 3 to 5 for all the scheduler steps to obtain the denoised inpainted image P_I.

The proposed inpainting procedure can be seen as the combination of two opposite processes: one that progressively denoises an initial sample of random noise, and another that incrementally degrades the known regions of the original image by adding noise in accordance with the diffusion schedule. These two processes converge at each timestep $t - 1$, where information from the noisy sample obtained is combined with the denoising sample.

In the early stages of the denoising process, the synthetic sample P_U generated by trying to denoise the random noise sample does not carry any meaningful information about the target image yet. However, by blending in the pixels from the degraded known image, according to the binary mask M, we are able to introduce a bias that guides the denoising model toward reconstructing the missing parts of the image in a way that remains coherent with the undamaged parts of the image. This is possible because the degraded known image P_k at timestep $t - 1$ retains information from the known part of the original image, but represents this information at a noise level consistent with the expected output of the diffusion model when attempting to perform a denoising step from timestep t to timestep $t - 1$.

This helps to avoid unexpected discontinuities in the intermediate inpainted images, which could otherwise confuse the model when trying to estimate the noise pattern in subsequent denoising steps and ultimately lead to inconsistent outputs and an incoherent final reconstruction.

5.3 AI denoising framework validation

The validation pipeline aims to evaluate the performances of the developed models and to assess their effectiveness in the proposed task of removing the retro-reflective markers blobs from the depth images. To this end, three experiments were conducted:

1. To evaluate the performance of the variational autoencoder in accurately representing the markerless depth images in the latent space and reconstruct the original images from their latent representation.

2. To evaluate the diffusion model's ability to perform inpainting on clean markerless images during a simulated task.

3. To evaluate the full pipeline by applying inpainting to real images affected by markers, with the objective of producing partially synthetic outputs more closely resembling the training set.

5.3.1 Validation of the Variational Autoencoder

The autoencoder was tested on the markerless acquisition images of the two test subjects. The validation experiment consisted of an encoding-decoding cycle: each normalized grayscale 256×256 image was fed into the autoencoder to produce a corresponding three-channel 64×64 latent representation. The decoder then reconstructed the image by decoding this latent representation back into the original 256×256 grayscale space. The output of the decoder was then compared to the original image by evaluating the following metrics:

- The **mean square error** (MSE) was computed to quantify the distance between the original and reconstructed image in the 256×256 grayscale space. A lower MSE indicates a more accurate reconstruction, as it reflects smaller pixel-level deviations.

- The **Perceptual difference** was estimated by feeding the original image and then the reconstructed image into the same standard pretrained AlexNet CNN used during training, and computing the MSE between the two feature maps obtained. A lower perceptual difference suggests that the two images are more

similar in terms of high-level 'semantic' features (this metric is more similar to a human perception of similarity).

5.3.2 Validation of the Denoising Probabilistic Diffusion Model on images without markers: Checkerboard experiment

Since obtaining ground truth for the synthetically generated images was not possible, a simulated inpainting task was chosen to evaluate the performance of the diffusion model. This was done using markerless depth images from the two test subjects, where a known and fixed binary mask was applied to all images in order to define the missing regions to be inpainted.

The binary mask used in the experiment was generated by creating a checkerboard pattern composed of repeated $n \times n$ pixel squares with the same dimensions as the original image, alternating between squares set to 1 and squares set to 0.

This pattern was then applied to the original image through element-wise multiplication, deleting the regions corresponding to the squares set to 0. The inpainting algorithm proposed in paragraph 5.2.4 was subsequently applied to reconstruct the missing areas.

The inpainted images were compared with the original ground-truth images by computing the Mean Squared Error (**MSE**) over the synthetically generated regions.

The following grid configurations were tested on the 256×256 test images:

- 8×8 grid with blocks of 32×32 pixels,
- 16×16 grid with blocks of 16×16 pixels,
- 32×32 grid with blocks of 8×8 pixels.

The results were compared with those obtained on a subset of the construction set, in order to provide a reference baseline.

5.3.3 Validation of the Denoising Probabilistic Diffusion Model on images with markers

To fulfill the task of removing infrared markers from depth images, two main objectives must be achieved in sequence:

1. Obtain a binary mask that corresponds as accurately as possible to the position of the markers in the depth image.
2. Apply the inpainting algorithm exclusively to the affected areas.

The final goal of applying the AI pipeline to marker-affected images is to produce inpainted images in which only the pixels corresponding to the markers are synthetically generated, while ensuring that the overall result resembles as closely as possible the markerless images used to train the diffusion model.

Obtaining the binary mask

The algorithm used to generate the binary mask of the areas to be inpainted requires both the depth and infrared channels extracted from the Azure Kinect recordings acquired during the simultaneous Azure Kinect-Vicon acquisitions for accurately localizing the marker regions.

Step 1: The depth and infrared (IR) frames are first loaded into memory. The depth image is then preprocessed according to the steps described in Paragraph 5.1.1 to obtain a normalized 256×256 Region of Interest (ROI). A corresponding ROI is extracted from the IR image using the same coordinates, maintaining pixel coordinates alignment between the two channels.

Step 2: The IR ROI is segmented twice by applying a threshold to obtain two binary masks:

- **Mask 1** highlighting the position of the retro-reflective markers, obtained with a single threshold set at the 99th percentile of the pixel values.
- **Mask 2** highlighting the subject's body, obtained with a double threshold defined between the 90th and 98th percentiles included.

Step 3: A conditional binary dilation is applied to the blobs corresponding to the markers by adding the pixels that:

- have an **8-neighbor** that is an edge pixel of the blob (determined by a L1 binary dilation of the blobs edge pixels)
- are not already part of the blobs (are not already set to 1 in Mask 1)
- are not part of the background (are not set to 0 in the depth ROI)
- are not part of the subject's body in the IR image (are not set to 1 in Mask 2)

The dilated blob mask provides the pixel coordinates of the regions to be inpainted in the depth ROI.

Performing inpainting

The Depth ROI is inpainted using the algorithm proposed in paragraph 5.2.4 to remove the blobs caused by the retro-reflective markers, using the binary mask obtained following the previous steps as a template to guide the generation of synthetic pixels.

Evaluation metrics

To assess the quality of generation and the similarity between the generated images and the training distribution, three complementary metrics were computed, comparing a set of images from the training set, named T, a set of images affected by marker artifacts before cleaning, named D, a set of inpainted images, named I:

1. **Kernel Inception Distance (KID)** [43]: computed on features extracted by the same AlexNet used to compute perceptual loss. The discrepancy between two features distributions D_1 and D_2 can be estimated by extracting a sample X from D_1 and Y from D_2 and computing the Maximum Mean Discrepancy with kernel k:

$$\begin{aligned}\mathrm{KID}(D_1, D_2) &= \mathrm{MMD}_k^2(D_1, D_2) \\ &= \mathbb{E}_{X,X'\sim D_1}\Big[k(X, X')\Big] + \mathbb{E}_{Y,Y'\sim D_2}\Big[k(Y, Y')\Big] \\ &- 2\,\mathbb{E}_{X\sim D_1,\, Y\sim D_2}\Big[k(X, Y)\Big]\end{aligned}$$

   Where k is the Radial Basis Function kernel, which is a similarity metric that estimate the similarity between the feature vectors x and y:

$$k_{\mathrm{RBF}}(x, y) = \exp\left(-\frac{\|x - y\|_2^2}{2\sigma^2}\right)$$

   Global-average-pooled activation values were extracted from the convolutional layers of the AlexNet, z-scored using the training set T as standard. The mismatch between the set of inpainted images I and the training set T and between the dirty images D and the training set T were measured computing $\Delta_{\mathrm{KID}}(I - T)$ and $\Delta_{\mathrm{KID}}(D - T)$ respectively. Lower KID values indicate better similarity between the set analyzed (I or D) and the training set T.

   If the distance $\Delta_{\mathrm{KID}}(I - T)$ between I and T is lower than the distance $\Delta_{\mathrm{KID}}(D - T)$ between D and T, improvement can be quantified by computing $\Delta_{\mathrm{KID}}(D - I)$. This metric shows whether inpainted images I move closer to the training distribution T than the dirty corrupted images D.

2. **Mahalanobis Distance** [44]: per-image Mahalanobis distance was computed between each image's feature vector and the training mean under the training covariance. Lower values indicate greater compatibility with the training distribution. Results are summarized as boxplots.

3. **Reconstruction L1 Loss** [45]: computed in pixel space using the same autoencoder employed by the latent DDPM. Lower values mean that the autoencoder can encode and decode the image more faithfully, reflecting greater consistency with the training distribution.

5.4 Integration of the AI framework in the Kinect SDK pipeline using the Pykinect Azure Library

The AI framework was integrated into a pipeline that enables the offline analysis of `.mkv` files acquired with an Azure Kinect device. The pipeline uses the native skeletal tracking provided by the Azure Kinect SDK, accessed through the `Pykinect Azure` wrapper, while also allowing frame-level modifications of the depth channel when required.

Each frame of the `.mkv` file is split into its constituent channels (RGB, IR, Depth). Each channel is preprocessed according to the steps described in paragraph 5.1.1, but this time the preprocessing steps are made reversible by saving the normalization parameters and the ROI coordinates in a temporary dictionary. The Inpainting pipeline described in paragraph 5.3.3 is applied exclusively to the depth channel and the inpainted ROI is then injected back into the depth channel of the `.mkv` frame being processed.

The Azure Kinect Body Tracking SDK **(`K4ABT`)** [21] is called through the `Pykinect Azure` [46] wrapper, to retrieve body segmentation and skeleton data corresponding to the frame's timestamp, such as joint positions and orientations. The body-tracking sequence is constructed frame-by-frame, by storing the extracted skeletal data in a per-frame dictionary together with associated metadata, including the frame identifier, the number of detected bodies, and the corresponding timestamp.

When the entire `.mkv` file has been parsed, an output `.c3d` file is generated, containing the skeletal tracking data of all frames, including joint positions and orientations, synchronized with the timestamp of the Azure Kinect recordings. This ensures that the modified depth data and the motion capture information are preserved in a standardized format for post-processing.

5.4.1 Evaluation of skeletal tracking performances after applying inpainting on the depth channel

The effects of the inpainting pipeline are evaluated by comparing the body-tracking sequences obtained through the K4ABT applied to the original recording, affected by marker noise, with those obtained from the inpainted recordings. Two metrics are used for evaluation:

- **Skeleton tracking success rate**, computed as the percentage of missing frames over the total frame count, with lower values indicating better noise rejection.
- **Stability of the estimated bone lengths**, computed as the variance of bone lengths across body segments, with lower values indicating higher tracking accuracy, since bone length is considered a kinematic invariant.

5.5 Insights into Kinect SDK: Evaluation of skeletal tracking performances after applying inpainting on the IR channel

The tests described in paragraph 5.4 showed that, although the proposed method is able to perform inpainting on depth images, it does not improve the performance of the SDK. To further investigate this limitation, an algorithm similar to the one described in Paragraph 5.4 was applied, in collaboration with a team of PhD researchers from ETRO [47], to Kinect `.mkv` recordings obtained from concurrent Vicon-Kinect acquisitions affected by marker noise.

Each session involved a single subject performing a slow exercise, comparable to those presented in Paragraph 5.1, specifically designed to assess the range of motion (`ROM`) of the hip, knee, and shoulder joints.

Unlike the approach followed in this work, where only the depth channel was taken into account, this new study also considered the IR modality. This broader analysis revealed that the IR channel has a much stronger influence on skeleton reconstruction and on the quality of joint tracking than the depth one. The research team demonstrated this effect by repeatedly applying flipping transformations to one channel while leaving the other unchanged. In all cases where skeletal tracking was successful, the output orientation consistently followed the IR frame orientation, regardless of the state of the depth channel. This confirmed the predominant role of the IR channel in driving the body tracking algorithm.

Following this insight, a decision was made with the research team to perform inpainting exclusively on the IR channel, filling in the missing blobs by using Telea's

fast marching method [48], which reconstructs missing regions by propagating intensity information inward from the surrounding boundary pixels.

The stability of skeletal tracking was assessed as in 5.4.1 by evaluating the percentage of missing frames and the bone length variability. Results obtained adopting this strategy were published by the ETRO Research team in proceedings of the 2025 DSP conference [18].

Chapter 6

Results and Discussion

6.1 AI denoising framework training

Figures 6.1 and 6.2 depict the training and validation loss curves for the autoencoder and DDPM, respectively. In both cases, the loss exhibits a steep initial decrease followed by an almost monotonic decay toward a plateau.

For the autoencoder, the reconstruction loss drops rapidly in the first tens of epochs and stabilizes around 3×10^{-3}, with training and validation closely overlapping.

For the DDPM, the MSE loss steadily declines across the full training run and levels off near 2.4×10^{-2}; the losses show stochastic oscillations, but no persistent gap between curves. The convergence and alignment of the two curves in both models suggest limited overfitting and good generalization.

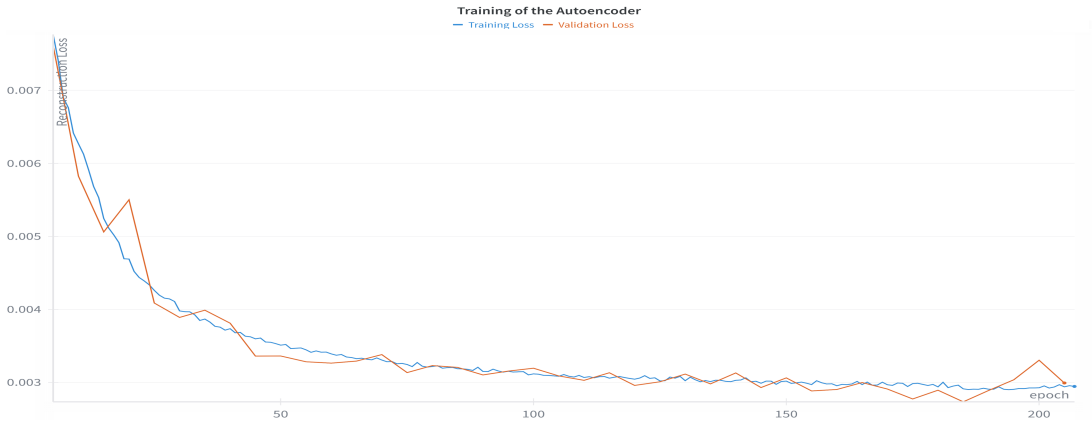

Figure 6.1: Training and validation loss of the Variational Autoencoder as a function of epoch. The blue training loss curve tracks the reconstruction loss computed on the training set at each epoch, while the orange validation loss curve tracks the loss computed on the validation set every five epochs. Lower values indicate better performance.

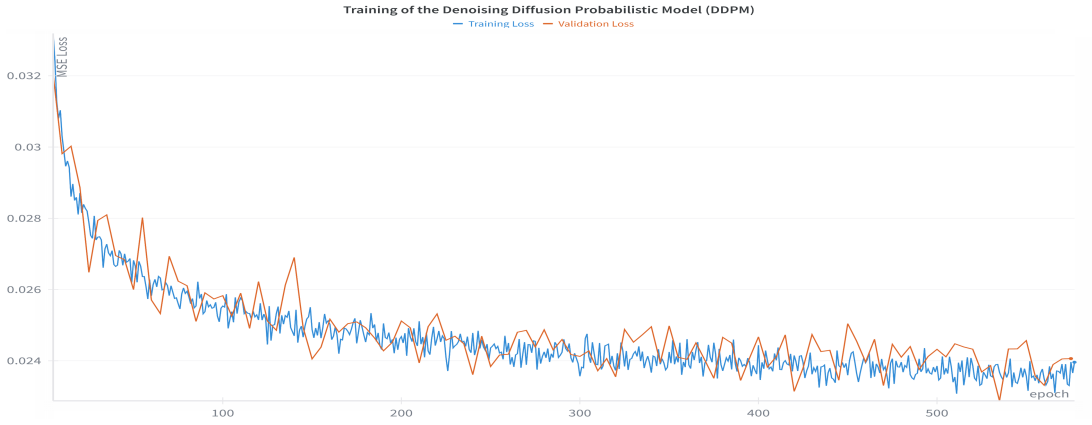

Figure 6.2: Training and validation loss of the Denoising Diffusion Probabilistic Model as a function of epoch. The blue training loss curve tracks the MSE loss computed on the training set at each epoch, while the orange validation loss curve tracks the MSE computed on the validation set every five epochs. Lower values indicate better performance.

6.2 AI denoising Framework validation

6.2.1 Validation of the Variational Autoencoder

Figure 6.3 shows a comparison example that highlights the similarity between a test-set image and its corresponding reconstruction obtained from the decoder, while Fig. 6.4 shows a visual representation of the three latent space channels of a batch of four images, providing insight into how the feature map is generated during the encoding process.

In Fig. 6.5, the distributions of the MSE and the Perceptual Loss computed on the test subjects are presented as boxplots and compared with the results obtained on the construction set, which serve as a baseline. The MSE values are centered around $3.21 \times 10^{-4} \pm 3 \times 10^{-6}$, while the Perceptual Loss values are centered around $8.37 \times 10^{-3} \pm 6 \times 10^{-5}$.

The low values obtained for the Mean Squared Error and the Perceptual Loss in the autoencoder test indicate good performance in encoding depth images into the latent space and subsequently decoding them back into the original pixel space. In particular, the low MSE reflects high local spatial accuracy, meaning that pixels at the same coordinates in the original and reconstructed images have very similar values. At the same time, the low perceptual loss suggests that the autoencoder was able to preserve the global fidelity of the image, maintaining its high-level features and visual content in the reconstructed version, beyond pixel-wise agreement.

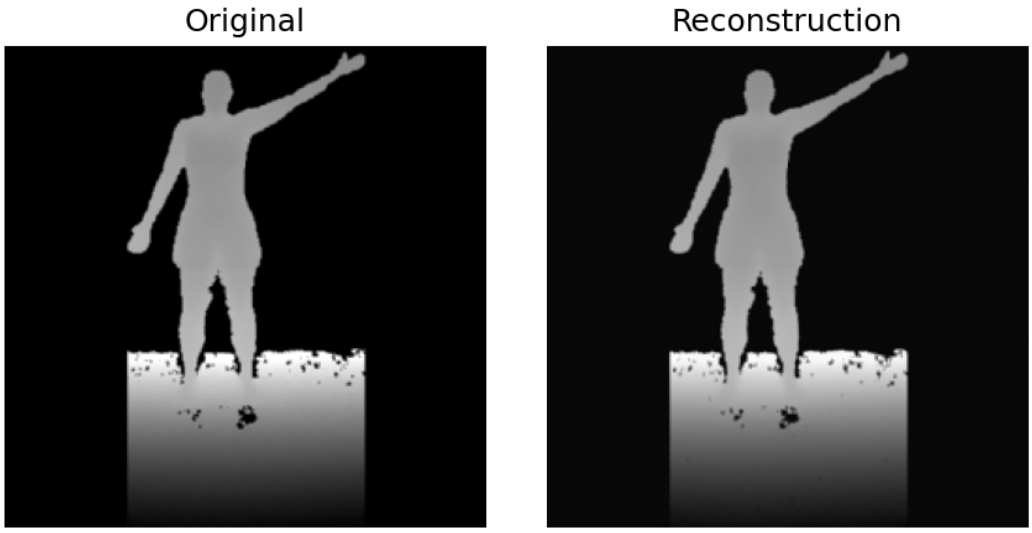

Figure 6.3: Example of an image reconstruction using the autoencoder. The original image is shown on the left, and the reconstruction from its latent space representation is shown on the right.

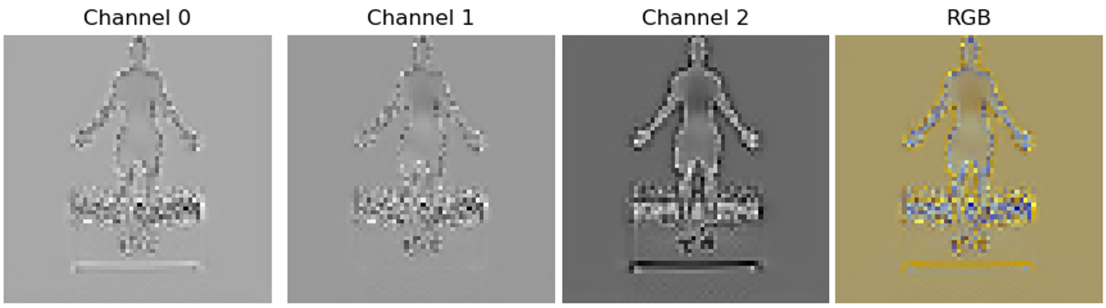

Figure 6.4: Examples of encoded latent space representations of a 256×256 image across the three 64×64 latent channels (feature maps). The three latent channels are shown in grayscale (values are normalized for visualization reasons), while the last image on the right provides an RGB visualization of the entire feature map, with the three channels mapped to red, green, and blue.

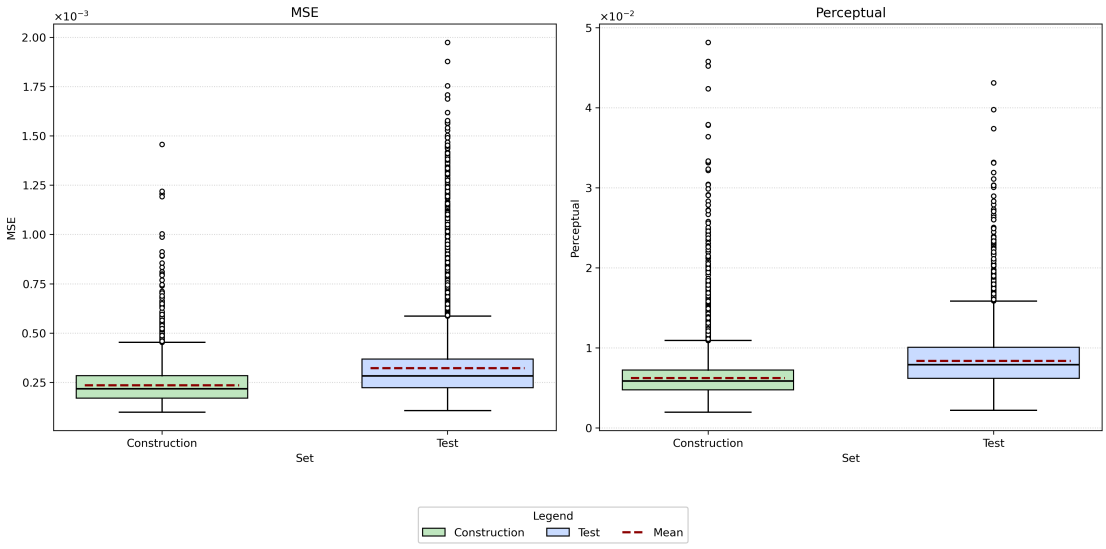

Figure 6.5: Boxplots of the Mean Squared Error (MSE) and perceptual loss between original images and the images generated by the autoencoder. Results obtained on the construction and test sets are shown, respectively, in green and blue.

6.2.2 Validation of the Denoising Probabilistic Diffusion Model on images without markers: Checkerboard experiment

The experiment of testing the inpainting algorithm with a known binary mask, using the image itself as ground truth, demonstrated that the diffusion model is capable of reconstructing missing regions in depth images. When the test was repeated with binary masks containing progressively larger voids, the results (shown in figure 6.6) exhibit an increase in loss for grids with larger missing areas. This behavior can be explained by considering that, among the pixels on which the DDPM is performing inference (pixels that are blacked out when applying the binary mask) the inference outcome becomes progressively less accurate for pixels located farther away from the boundary conditions. This is due to the fact that the surrounding pixels are themselves inferred rather than deterministically set to match the original image, so the farther a pixel lies from the known boundaries, the more its reconstruction is affected by uncertainty. The MSE values are centered around $5.45\times10^{-3}\pm2.02\times10^{-3}$ for the 32×32 grid of 8×8 pixel squares $6.17 \times 10^{-3} \pm 1.49 \times 10^{-3}$ for the 16×16 grid of 16×16 pixel squares $8.25 \times 10^{-3} \pm 2.4 \times 10^{-3}$ for the 8×8 grid of 32×32 pixel squares. Figure 6.7 shows samples of inpainted images for different grid sizes.

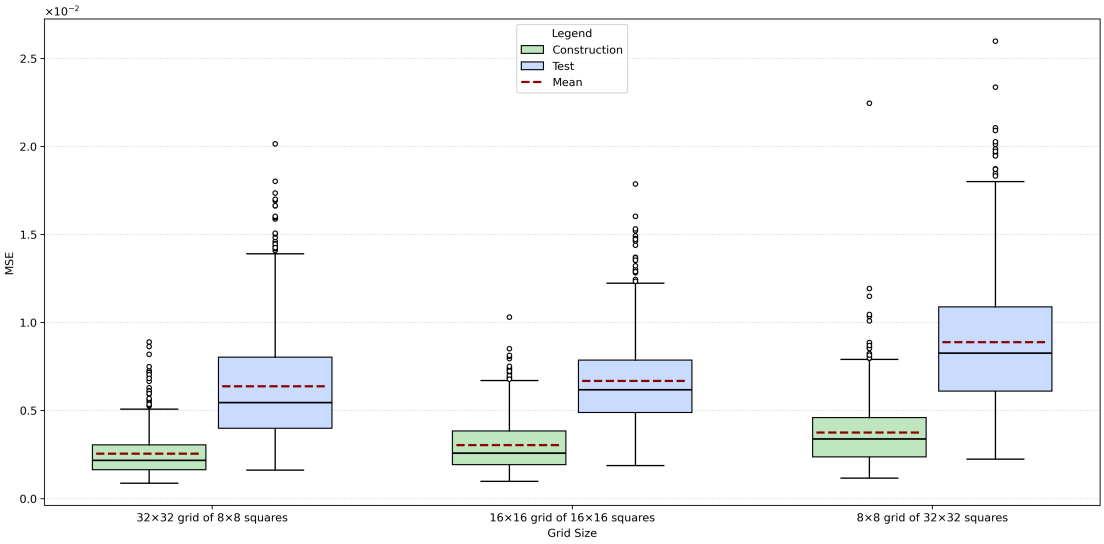

Figure 6.6: Boxplots of the mean squared error (MSE) values between original images and the inpainted images in the simulated checkerboard mask filling task. Results obtained on the construction and test sets are shown, respectively, in green and blue. The plots are grouped by grid size: from left to right, a 32×32 grid of 8×8 pixel squares, a 16×16 grid of 16×16 pixel squares, and a 8×8 grid of 32×32 pixel squares.

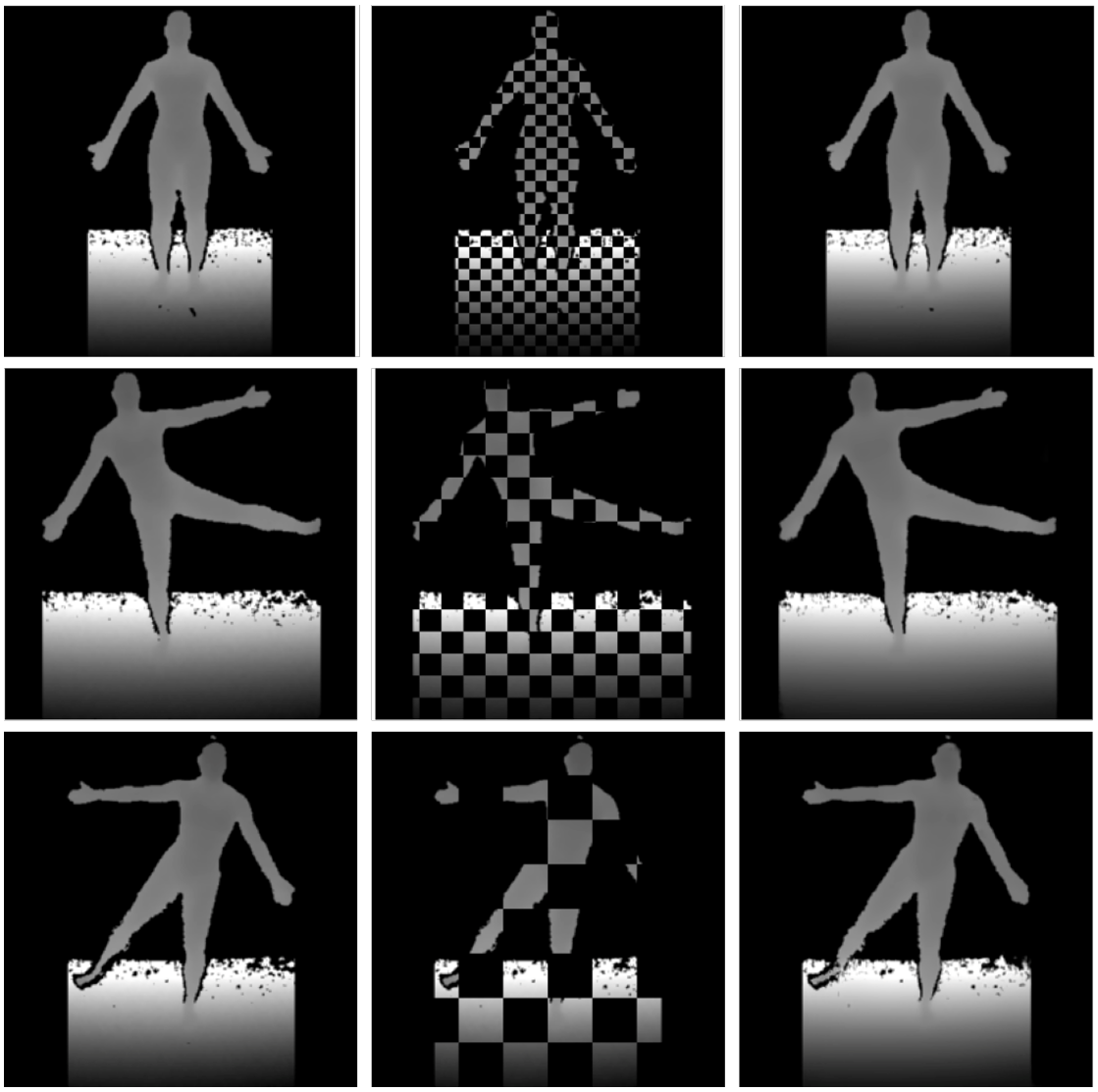

Figure 6.7: Example from the simulated inpainting task: each row corresponds to a different masking pattern: a 32×32 grid of 8×8 squares (top), a 16×16 grid of 16×16 squares (middle), and an 8×8 grid of 32×32 squares (bottom). Within each row, the columns show (from left to right) the original ground-truth image, the image with missing pixels, and the inpainted result.

6.2.3 Validation of the Denoising Probabilistic Diffusion Model on images with markers

The segmentation algorithm was able to generate an accurate binary mask of the marker positions, as shown in Fig. 6.8.b, where the segmentation mask is overlaid on the IR channel.

Through multimodal dilation (step 3 of paragraph 5.3.3), this information was transferred to the depth channel as shown in Fig. 6.8.c. However, it was not possible to completely prevent undamaged pixels belonging to the subject's body from being included in the inpainting mask, nor to avoid pixels corresponding to marker-induced artifacts being excluded from it. For this reason, the inpainting algorithm produced noisier reconstructions than those obtained in the checkerboard inpainting task. Figure 6.8.d shows an example of an inpainted image, in which and residual artifacts remain visible around the subject.

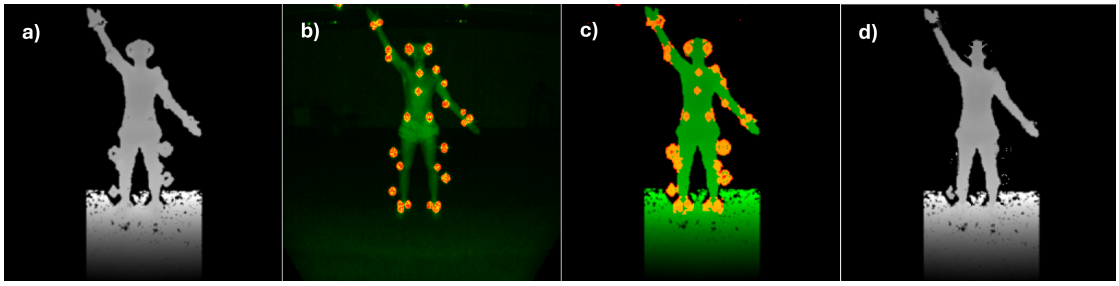

Figure 6.8: SDK Frame inpainting example: **a)** Depth ROI before inpainting **b)** Segmentation mask of the retro-reflective markers position overlaid on the IR channel **c)** Binary segmentation mask of retro-reflective marker artifacts overlaid on the depth channel **d)** Inpainted ROI, where holes and residual artifacts are still visible in the final reconstruction

.

The outcomes for each metric introduced in 5.4.1 are reported below:

- **Kernel Inception Distance (KID)** (Fig. 6.9): Inpainted images achieved a lower KID values than the Images affected by markers: (0.395×10^{-3} vs. 0.457×10^{-3}). Moreover, a bootstrap test was executed by repeatedly sampling each set (D, I, T) and computing the Kernel Inception Distances $\Delta_{\text{KID}}(D-T)$ $\Delta_{\text{KID}}(I-T)$ and their difference $\Delta_{\text{KID}}(D-I)$.

  The test results showed that the 95% bootstrap confidence intervals of $\Delta_{\text{KID}}(D-T)$ and $\Delta_{\text{KID}}(I-T)$ did not overlap ($[0.383,0.410] \times 10^{-3}$ vs. $[0.444,0.471] \times 10^{-3}$) indicating a statistically significant reduction in distributional discrepancy between the inpainted images and the training set compared to the images affected by markers.

- **Mahalanobis Distance** (Fig. 6.10-left panel): The median distance of the Inpainted images was slightly higher than that of the Images affected by markers (3.02×10^7 vs. 2.38×10^7), and the paired Wilcoxon test confirmed the difference was not significant ($z = -1.40, p = 0.162$). Therefore, no improvement was observed according to this metric.

- **Reconstruction L1** (Fig. 6.10-right panel): Inpainted images obtained lower reconstruction error compared to Images affected by markers (0.0227 ± 0.0104 vs. 0.0252 ± 0.0143), although both remain well above the training baseline (0.0040 ± 0.0007). This suggests that the inpainted images are marginally easier to encode and decode, but still far from the clean training distribution.

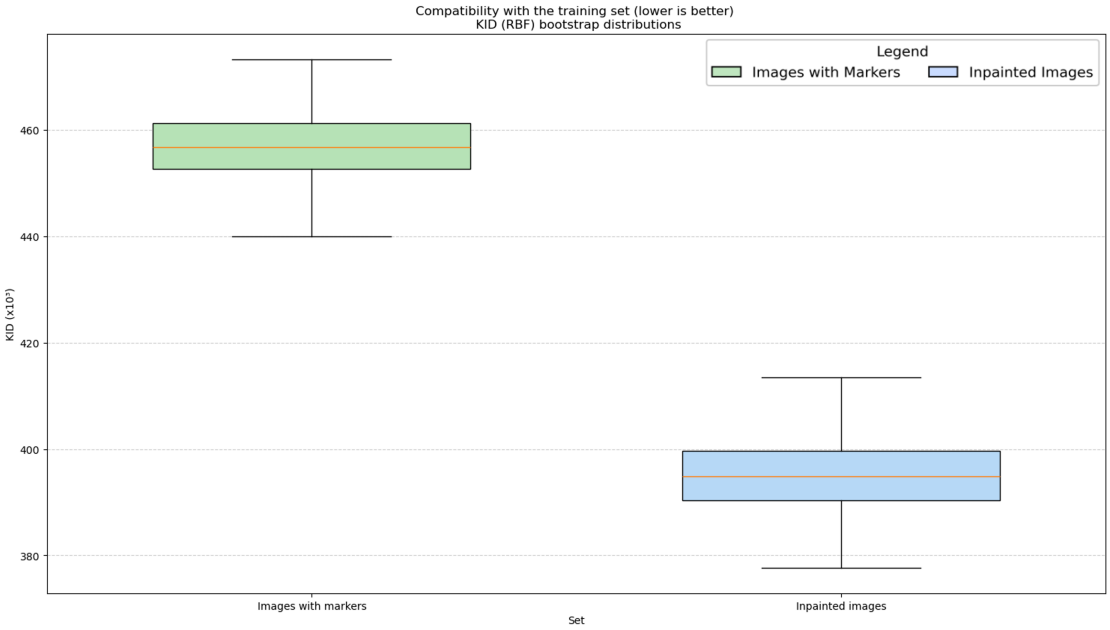

Figure 6.9: Boxplots of the Kernel inception distance used to assess the similarity of the inpainted images with the training set, compared to the images affected by markers. The KID bootstrap distributions show a statistically significative difference

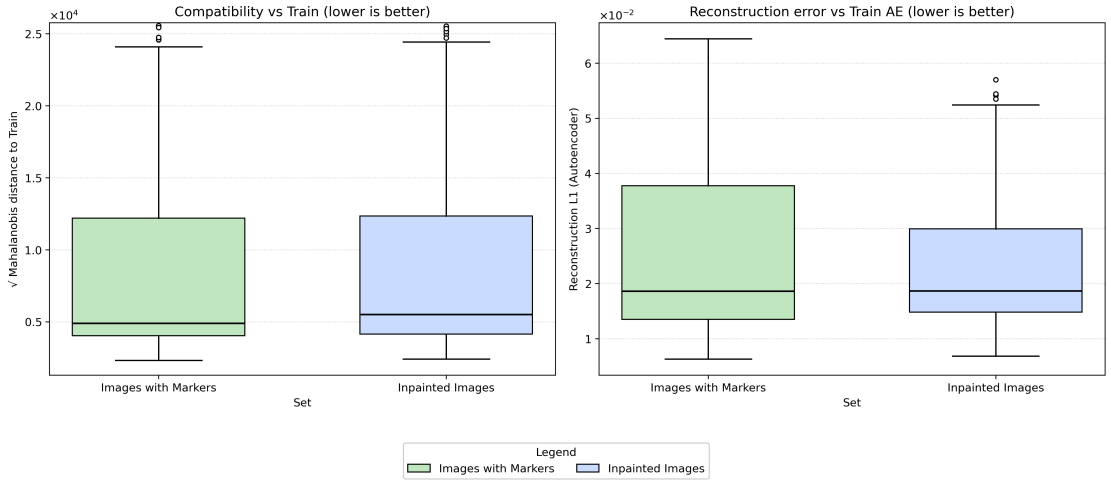

Figure 6.10: Boxplots of the Mahalanobis and L1 distances used to assess the similarity of the inpainted images with the training set, compared to the images affected by markers. Mahalanobis and L1 distributions do not show a statistically significative difference

6.3 Results obtained inpainting the Depth channel

As anticipated in Paragraph 5.5, the fact that the body tracking model (K4ABT) relies more heavily on IR images explains why the attempt to integrate the depth-image modification pipeline into the K4ABT post-processing analysis did not lead to improvements in the quality or stability of skeletal tracking. Seven out of the eight recordings used for testing failed to output a continuous body tracking sequence, with an average 3.30% of missing frames.

As shown in Fig. 6.11, the model fails to correctly track the subject's skeleton, and overlaps between body segments are also evident. Similarly, in Fig. 6.12, it can be observed that the intervention on the depth channel did not reduce bone length variability; in fact, the standard deviation was found to be statistically significantly greater for two of the four exercises (SAB and KFB).

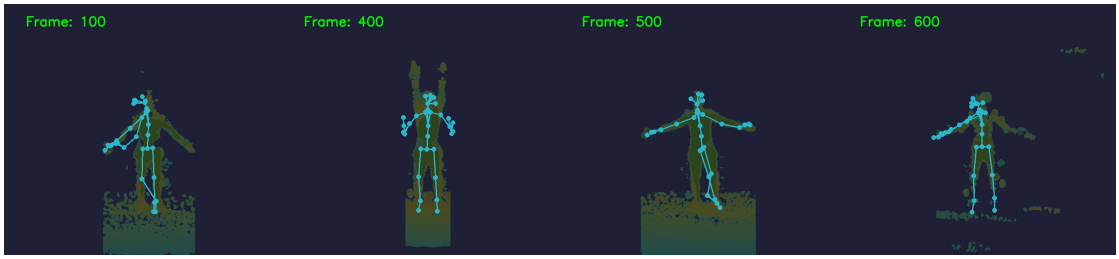

Figure 6.11: Failed skeleton reconstruction in different frames: inpainting the depth channel alone did not improve skeletal tracking

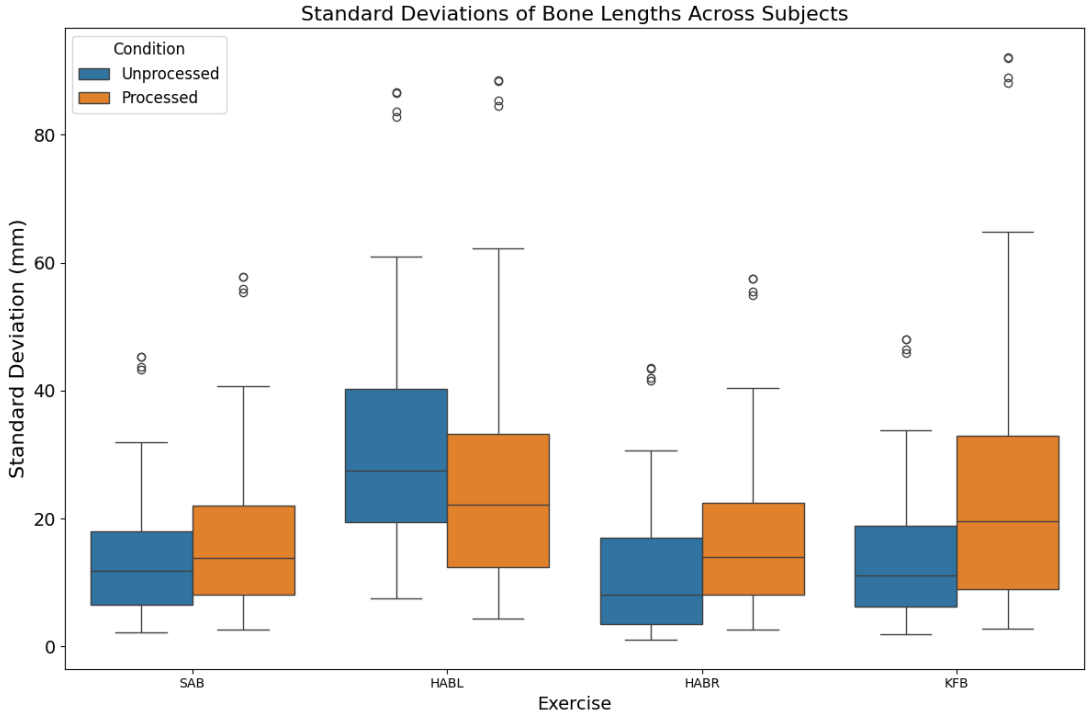

Figure 6.12: Boxplots of the measured bone length variability obtained adopting the Depth inpainting strategy, grouped by exercise

6.4 Results obtained adopting the IR inpainting strategy

As demonstrated by the channel flipping experiments discussed in Section 5.5, the Kinect body tracking model (K4ABT) primarily relies on the infrared (IR) channel for generating skeletal output, as illustrated in Fig. 6.13.

The application of Telea's fast marching inpainting method [48] to the IR channel reduced the occurrence of missing skeletal data to only two recordings, with an average of 0.16% missing frames (Fig. 6.14). Moreover, the proposed IR inpainting method [18] significantly reduced bone length variability compared to the standard K4ABT output (Shapiro-Wilk test; $p \leq 0.005$), consistently across all subjects and all exercises (Fig. 6.15).

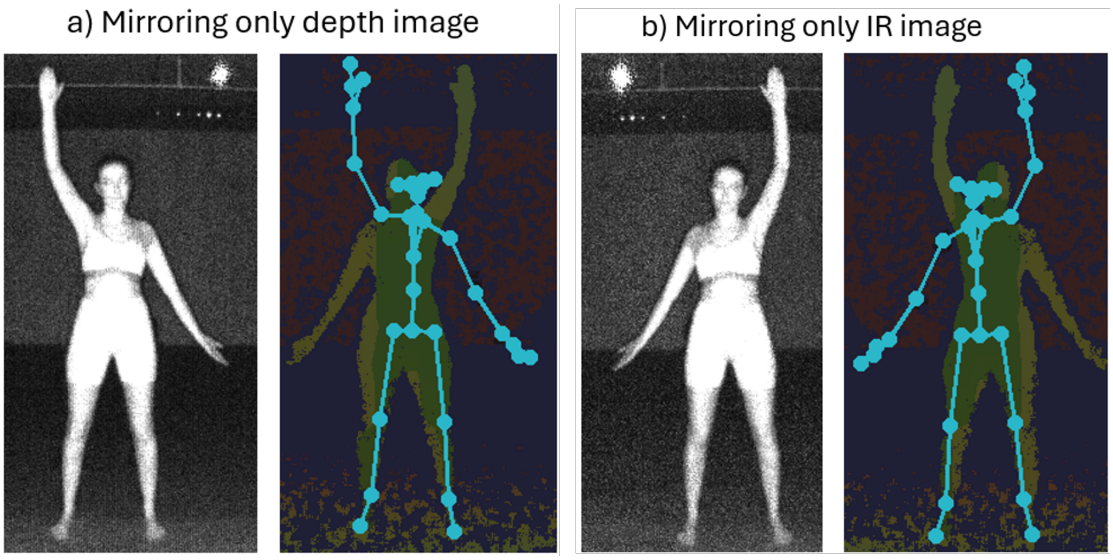

Figure 6.13: The IR flipping experiments performed on markerless images: on the left, the IR channel; on the right, the output of the body tracking model K4ABT, overlaid to the Depth channel. The estimated skeleton always matches the IR orientation, regardless of the depth.

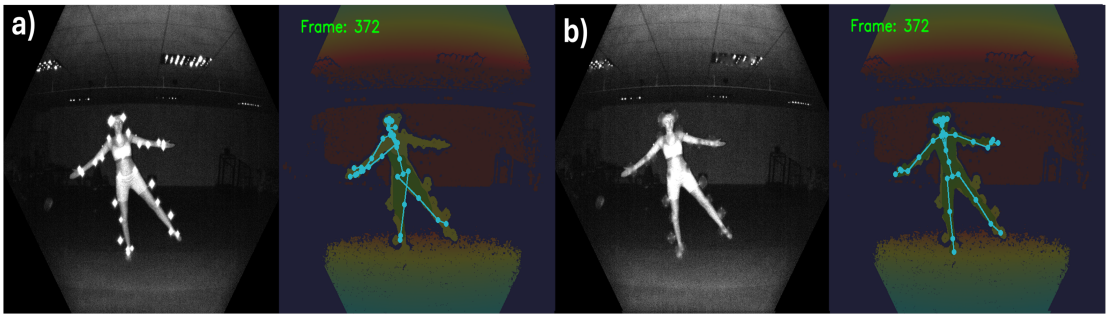

Figure 6.14: Results obtained a) before and b) after applying Telea's Fast marching method [48] on IR images. Body segment overlapping is greatly reduced.

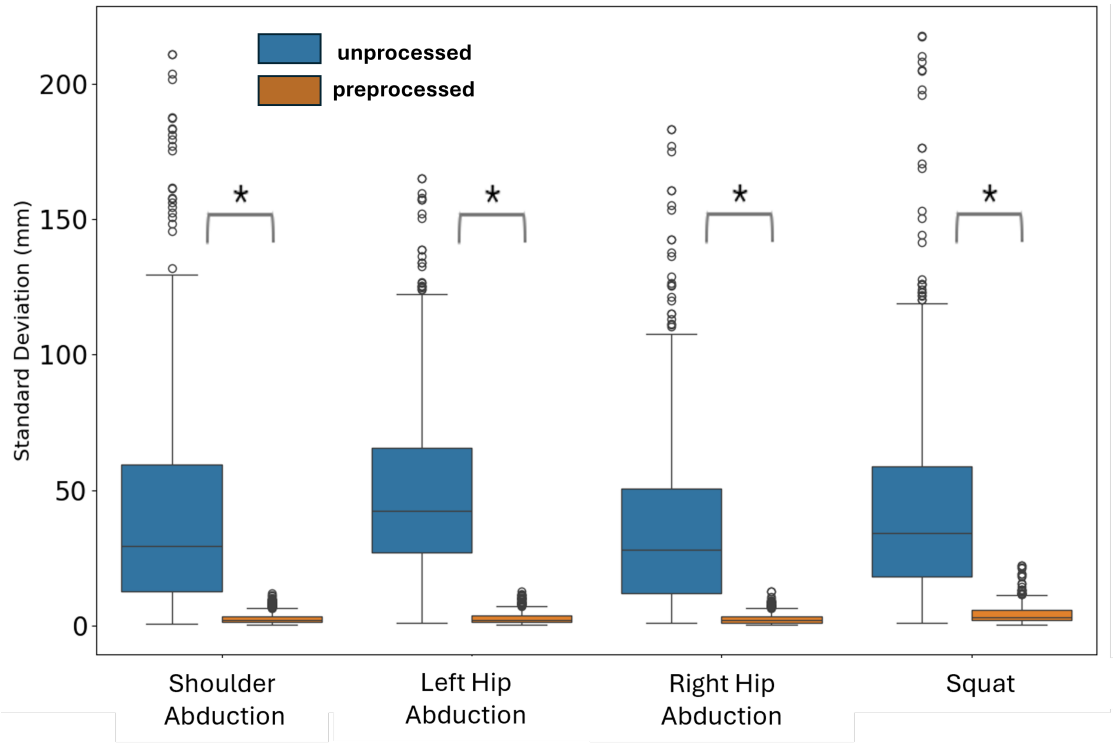

Figure 6.15: Boxplots of the measured bone length variability obtained adopting the IR inpainting strategy, grouped by exercise.

Chapter 7

Conclusions

7.1 Contributions

Analyzing the results obtained, as described in chapter 6, this thesis established the following contributions: we demonstrated that a diffusion probabilistic model, paired with an autoencoder that is capable of encoding depth images to and from the DDPM latent space, can generate synthetic depth images that are consistent with the clean training set. That the same latent-space deep learning framework, previously applied mainly in diagnostic medical imaging, successfully inpainted depth images by reconstructing areas degraded by marker artifacts. Inpainted outputs were shown to be more perceptually coherent with the clean-image distribution than the corresponding artifact-affected inputs; this could enable the developing of an automated method for building datasets of paired images (with and without markers), that could enable direct comparisons between Kinect and Vicon recordings obtained during concurrent acquisitions. However, we also demonstrated that inpainting only the depth channel yields limited to no improvement in skeletal tracking when using the default Azure Kinect Body Tracking SDK (K4ABT).

In collaboration with the ETRO research team, we demonstrated that applying a simpler inpainting procedure on the infrared (IR) channel significantly improves the stability and quality of the estimated skeletons.

7.2 Limitations and future improvement

The present study also presents several limitations that should be considered:

The data acquisition involved a small cohort of healthy subjects performing simple exercises at a slow pace, and all recordings were acquired from a single point of view. This may limit repeatability and validity, as Azure Kinect accuracy depends on field of view and speed of motion [33, 36].

The experiments also relied on Microsoft's Azure Kinect hardware and software: because this platform has been discontinued, future replications should validate the proposed pipeline on Orbbec's Femto Bolt, which is built to the same specifications, together with the OrbbecSDK K4A Wrapper [49].

The marker-segmentation procedure produced a non-negligible number of false negatives, leaving residual artifacts that increased output noise. This could be mitigated by testing alternative masking patterns and bounding boxes during inpainting; for example, a square box to replicate the checkerboard inpainting setup that yielded satisfactory results.

Moreover, the inference time of the DDPM is relatively high, which limits its applicability in real-time settings.

Since the Kinect's body tracking model accuracy has been shown to be almost entirely dependent on the quality of the IR images, a more sophisticated inpainting technique, that involves applying the diffusion generative models capabilities to both the depth and the IR channels, could be implemented in future developments.

Bibliography

[1] Bradley Scott, Martin Seyres, Fraser Philp, Edward K. Chadwick, and Dimitra Blana. «Healthcare applications of single camera markerless motion capture: a scoping review». In: *PeerJ* 10 (May 2022), e13517. ISSN: 2167-8359. DOI: 10.7717/peerj.13517. URL: https://doi.org/10.7717/peerj.13517 (cit. on pp. ii, 1, 2).

[2] Hyungtai Kim, Yun-Hee Kim, Seung-Jong Kim, and Mun-Taek Choi. «Pathological gait clustering in post-stroke patients using motion capture data». In: *Gait & Posture* 94 (2022), pp. 210–216. ISSN: 0966-6362. DOI: https://doi.org/10.1016/j.gaitpost.2022.03.007. URL: https://www.sciencedirect.com/science/article/pii/S0966636222000790 (cit. on pp. ii, 1, 2).

[3] Judy Seesahai, Maureen Luther, Paige Terrien Church, Patricia Maddalena, Elizabeth Asztalos, Thomas Rotter, and Rudaina Banihani. «The assessment of general movements in term and late-preterm infants diagnosed with neonatal encephalopathy, as a predictive tool of cerebral palsy by 2 years of age-a scoping review». In: *Systematic Reviews* 10.1 (2021), p. 226 (cit. on pp. ii, 2).

[4] Maria do Carmo Vilas-Boas, Ana Patrícia Rocha, Márcio Neves Cardoso, José Maria Fernandes, Teresa Coelho, and João Paulo Silva Cunha. «Clinical 3-D Gait Assessment of Patients With Polyneuropathy Associated With Hereditary Transthyretin Amyloidosis». In: *Frontiers in Neurology* Volume 11 - 2020 (2020). ISSN: 1664-2295. DOI: 10.3389/fneur.2020.605282. URL: https://www.frontiersin.org/journals/neurology/articles/10.3389/fneur.2020.605282 (cit. on pp. ii, 2).

[5] Sina Mehdizadeh, Mohammadreza Faieghi, Andrea Sabo, Hoda Nabavi, Avril Mansfield, Alastair J. Flint, Babak Taati, and Andrea Iaboni. «Gait changes over time in hospitalized older adults with advanced dementia: Predictors of mobility change». In: *PLOS ONE* 16.11 (Nov. 2021), pp. 1–13. DOI: 10.1371/journal.pone.0259975. URL: https://doi.org/10.1371/journal.pone.0259975 (cit. on pp. ii, 2).

[6] Winnie WT Lam, Yuk Ming Tang, and Kenneth NK Fong. «A systematic review of the applications of markerless motion capture (MMC) technology for clinical measurement in rehabilitation». In: *Journal of NeuroEngineering and Rehabilitation* 20.1 (2023), p. 57 (cit. on pp. ii, 2).

[7] Adam Chromy, Petr Sopak, and Hynek Cigler. «Validated low-cost standardized VICON configuration as a practical approach to estimating the minimal accuracy of a specific setup». In: *Scientific Reports* 15.1 (2025), p. 23351 (cit. on pp. ii, 1, 5).

[8] Gregorij Kurillo, Evan Hemingway, Mu-Lin Cheng, and Louis Cheng. «Evaluating the Accuracy of the Azure Kinect and Kinect v2». In: *Sensors* 22.7 (2022). ISSN: 1424-8220. DOI: 10.3390/s22072469. URL: https://www.mdpi.com/1424-8220/22/7/2469 (cit. on pp. ii, 1, 6).

[9] *Vicon Vero | Advanced Super Wide Motion Capture Camera*. Vicon Motion Systems Ltd. URL: https://www.vicon.com/hardware/cameras/vero/ (visited on 09/15/2025) (cit. on pp. ii, 1, 3, 5).

[10] Michal Tölgyessy, Martin Dekan, Ľuboš Chovanec, and Peter Hubinský. «Evaluation of the Azure Kinect and Its Comparison to Kinect V1 and Kinect V2». In: *Sensors* 21.2 (2021). ISSN: 1424-8220. DOI: 10.3390/s21020413. URL: https://www.mdpi.com/1424-8220/21/2/413 (cit. on pp. ii, 1, 6).

[11] Tanwi Mallick, Partha Pratim Das, and Arun Kumar Majumdar. «Characterizations of noise in Kinect depth images: A review». In: *IEEE Sensors journal* 14.6 (2014), pp. 1731–1740 (cit. on pp. ii, 1, 9).

[12] MReza Naeemabadi, Birthe Dinesen, Ole Kæseler Andersen, and John Hansen. «Investigating the impact of a motion capture system on Microsoft Kinect v2 recordings: A caution for using the technologies together». In: *PloS one* 13.9 (2018), e0204052 (cit. on pp. ii, 1, 9).

[13] Mreza Naeemabadi, Birthe Dinesen, Ole Kæseler Andersen, and John Hansen. «Influence of a Marker-Based Motion Capture System on the Performance of Microsoft Kinect v2 Skeleton Algorithm». In: *IEEE Sensors Journal* 19.1 (2019), pp. 171–179. DOI: 10.1109/JSEN.2018.2876624 (cit. on pp. ii, 9).

[14] Anargyros Chatzitofis, Georgios Albanis, Nikolaos Zioulis, and Spyridon Thermos. «A Low-Cost & Real-Time Motion Capture System». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 21453–21458 (cit. on pp. ii, 10, 12).

[15] Nikolas Hesse, Sandra Baumgartner, Anja Gut, and Hubertus J. A. van Hedel. «Concurrent Validity of a Custom Method for Markerless 3D Full-Body Motion Tracking of Children and Young Adults Based on a Single RGB-D Camera». In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 31 (2023), pp. 1943–1951. DOI: 10.1109/TNSRE.2023.3251440 (cit. on pp. ii, 10, 12).

[16] *Rehabilitation Research / RERE*. Vrije Universiteit Brussel (VUB). URL: https://rere.research.vub.be/ (visited on 09/15/2025) (cit. on pp. ii, 26).

[17] Silvia Zaccardi, Redona Brahimetaj, Erick Rodriguez, Sven Van Den Bergh, Mona Ibrahim, Eva Swinnen, David Beckwée, and Bart Jansen. «Impact of infrared interference on Azure Kinect's motion tracking performance during validation studies against marker-based gold standard». In: *Gait & Posture* 113 (2024), pp. 262–263 (cit. on pp. iv, 26).

[18] Silvia Zaccardi, Redona Brahimetaj, Federico Trovalusci, Reinhard Claeys, Rossana Lovecchio, David Beckwée, Eva Swinnen, and Bart Jansen. «Insights Into Azure Kinect Skeletal Tracking: A Simple Approach to Reduce IR Passive Noise». In: *2025 25th International Conference on Digital Signal Processing (DSP)*. IEEE. 2025, pp. 1–5 (cit. on pp. iv, 42, 53).

[19] *2D DDPM Tutorial (GenerativeModels)*. Project MONAI. URL: https://github.com/Project-MONAI/GenerativeModels/tree/main/tutorials/generative/2d_ddpm (visited on 09/15/2025) (cit. on pp. iv, 30, 33, 34).

[20] Walter H. L. Pinaya, Petru-Daniel Tudosiu, Jessica Dafflon, Pedro F da Costa, Virginia Fernandez, Parashkev Nachev, Sebastien Ourselin, and M. Jorge Cardoso. *Brain Imaging Generation with Latent Diffusion Models*. 2022. arXiv: 2209.07162 [eess.IV]. URL: https://arxiv.org/abs/2209.07162 (cit. on pp. iv, 14, 30, 31, 33).

[21] *Azure Kinect DK*. Microsoft. URL: https://azure.microsoft.com/en-us/products/kinect-dk (visited on 09/15/2025) (cit. on pp. 1, 6, 40).

[22] Wesley Niswander, Wei Wang, and Kimberly Kontson. «Optimization of IMU Sensor Placement for the Measurement of Lower Limb Joint Kinematics». In: *Sensors* 20.21 (2020). ISSN: 1424-8220. DOI: 10.3390/s20215993. URL: https://www.mdpi.com/1424-8220/20/21/5993 (cit. on pp. 3, 4).

[23] Chang June Lee and Jung Keun Lee. «Inertial Motion Capture-Based Wearable Systems for Estimation of Joint Kinetics: A Systematic Review». In: *Sensors* 22.7 (2022). ISSN: 1424-8220. DOI: 10.3390/s22072507. URL: https://www.mdpi.com/1424-8220/22/7/2507 (cit. on p. 3).

[24] Sofia Scataglini, Eveline Abts, Cas Van Bocxlaer, Maxime Van den Bussche, Sara Meletani, and Steven Truijen. «Accuracy, Validity, and Reliability of Markerless Camera-Based 3D Motion Capture Systems versus Marker-Based 3D Motion Capture Systems in Gait Analysis: A Systematic Review and Meta-Analysis». In: *Sensors* 24.11 (2024). ISSN: 1424-8220. DOI: 10.3390/s24113686. URL: https://www.mdpi.com/1424-8220/24/11/3686 (cit. on p. 3).

[25] *Motion Capture Technology and Systems | Qualisys*. Qualisys. URL: https://www.qualisys.com/ (visited on 09/25/2025) (cit. on p. 3).

[26] *Theia3D Documentation*. Theia Markerless. URL: https://docs.theiamarkerless.com/ (visited on 09/25/2025) (cit. on p. 3).

[27] Filipe Conceição, Martin Lewis, Hernâni Lopes, and Elza M. M. Fonseca. «An Evaluation of the Accuracy and Precision of Jump Height Measurements Using Different Technologies and Analytical Methods». In: *Applied Sciences* 12.1 (2022). ISSN: 2076-3417. DOI: 10.3390/app12010511. URL: https://www.mdpi.com/2076-3417/12/1/511 (cit. on pp. 3, 4).

[28] John F. Drazan, William T. Phillips, Nidhi Seethapathi, Todd J. Hullfish, and Josh R. Baxter. «Moving outside the lab: Markerless motion capture accurately quantifies sagittal plane kinematics during the vertical jump». In: *Journal of Biomechanics* 125 (2021), p. 110547. ISSN: 0021-9290. DOI: https://doi.org/10.1016/j.jbiomech.2021.110547. URL: https://www.sciencedirect.com/science/article/pii/S0021929021003274 (cit. on p. 5).

[29] Robert M. Kanko, Elise K. Laende, Elysia M. Davis, W. Scott Selbie, and Kevin J. Deluzio. «Concurrent assessment of gait kinematics using marker-based and markerless motion capture». In: *Journal of Biomechanics* 127 (2021), p. 110665. ISSN: 0021-9290. DOI: https://doi.org/10.1016/j.jbiomech.2021.110665. URL: https://www.sciencedirect.com/science/article/pii/S0021929021004346 (cit. on p. 5).

[30] Gerda Strutzenberger, Robert Kanko, Scott Selbie, Hermann Schwameder, and Kevin Deluzio. «Assessment of kinematic CMJ data using a deep learning algorithm-based markerless motion capture system». In: *ISBS Proceedings Archive* 39.1 (2021), p. 236 (cit. on p. 5).

[31] Ross A. Clark, Benjamin F. Mentiplay, Emma Hough, and Yong Hao Pua. «Three-dimensional cameras and skeleton pose tracking for physical function assessment: A review of uses, validity, current developments and Kinect alternatives». In: *Gait & Posture* 68 (2019), pp. 193–200. ISSN: 0966-6362. DOI: https://doi.org/10.1016/j.gaitpost.2018.11.029. URL: https:

//www.sciencedirect.com/science/article/pii/S0966636218311913 (cit. on p. 5).

[32] *Azure Kinect DK documentation*. Microsoft. URL: https://learn.microsoft.com/en-us/previous-versions/azure/kinect-dk/ (visited on 09/15/2025) (cit. on pp. 6, 7).

[33] Yunru Ma, Bo Sheng, Rylea Hart, and Yanxin Zhang. «The validity of a dual Azure Kinect-based motion capture system for gait analysis: a preliminary study». In: *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. 2020, pp. 1201–1206 (cit. on pp. 12, 56).

[34] Megumi Ota, Hiroshige Tateuchi, Takaya Hashiguchi, and Noriaki Ichihashi. «Verification of validity of gait analysis systems during treadmill walking and running using human pose tracking algorithm». In: *Gait & Posture* 85 (2021), pp. 290–297. ISSN: 0966-6362. DOI: https://doi.org/10.1016/j.gaitpost.2021.02.006. URL: https://www.sciencedirect.com/science/article/pii/S0966636221000448 (cit. on p. 12).

[35] Jacob Thomas, Jamie B. Hall, Rebecca Bliss, and Trent M. Guess. «Comparison of Azure Kinect and optical retroreflective motion capture for kinematic and spatiotemporal evaluation of the sit-to-stand test». In: *Gait & Posture* 94 (2022), pp. 153–159. ISSN: 0966-6362. DOI: https://doi.org/10.1016/j.gaitpost.2022.03.011. URL: https://www.sciencedirect.com/science/article/pii/S0966636222000832 (cit. on p. 12).

[36] Trent M. Guess, Rebecca Bliss, Jamie B. Hall, and Andrew M. Kiselica. «Comparison of Azure Kinect overground gait spatiotemporal parameters to marker based optical motion capture». In: *Gait & Posture* 96 (2022), pp. 130–136. ISSN: 0966-6362. DOI: https://doi.org/10.1016/j.gaitpost.2022.05.021. URL: https://www.sciencedirect.com/science/article/pii/S0966636222001539 (cit. on pp. 12, 56).

[37] Jonathan Ho, Ajay Jain, and Pieter Abbeel. «Denoising Diffusion Probabilistic Models». In: *arXiv preprint arxiv:2006.11239* (2020) (cit. on pp. 14, 15, 19, 20, 23, 33).

[38] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2022. arXiv: 2112.10752 [cs.CV]. URL: https://arxiv.org/abs/2112.10752 (cit. on pp. 14, 23, 30, 31, 33).

[39] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. «Deep Unsupervised Learning using Nonequilibrium Thermodynamics». In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 2256–2265. URL: https://proceedings.mlr.press/v37/sohl-dickstein15.html (cit. on pp. 14, 15).

[40] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2022. arXiv: 1312.6114 [stat.ML]. URL: https://arxiv.org/abs/1312.6114 (cit. on pp. 23–25).

[41] *2D AutoencoderKL Tutorial (GenerativeModels)*. Project MONAI. URL: https://github.com/Project-MONAI/GenerativeModels/tree/main/tutorials/generative/2d_autoencoderkl (visited on 09/15/2025) (cit. on pp. 30, 31).

[42] *2D DDPM Inpainting (GenerativeModels)*. Project MONAI. URL: https://github.com/Project-MONAI/GenerativeModels/blob/main/tutorials/generative/2d_ddpm/2d_ddpm_inpainting.ipynb (visited on 09/15/2025) (cit. on p. 34).

[43] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. «Demystifying mmd gans». In: *arXiv preprint arXiv:1801.01401* (2018) (cit. on p. 39).

[44] Henryk Maciejewski, Tomasz Walkowiak, and Kamil Szyc. «Out-of-Distribution Detection in High-Dimensional Data Using Mahalanobis Distance - Critical Analysis». In: *Computational Science – ICCS 2022*. Ed. by Derek Groen, Clélia de Mulatier, Maciej Paszynski, Valeria V. Krzhizhanovskaya, Jack J. Dongarra, and Peter M. A. Sloot. Cham: Springer International Publishing, 2022, pp. 262–275. ISBN: 978-3-031-08751-6 (cit. on p. 40).

[45] Nishant Khare, Poornima Singh Thakur, Pritee Khanna, and Aparajita Ojha. «Analysis of Loss Functions for Image Reconstruction Using Convolutional Autoencoder». In: *Computer Vision and Image Processing*. Ed. by Balasubramanian Raman, Subrahmanyam Murala, Ananda Chowdhury, Abhinav Dhall, and Puneet Goyal. Cham: Springer International Publishing, 2022, pp. 338–349. ISBN: 978-3-031-11349-9 (cit. on p. 40).

[46] Ibai Gorordo. *pyKinectAzure*. Version 0.0.4. June 22, 2024. URL: https://github.com/ibaiGorordo/pyKinectAzure (visited on 09/15/2025) (cit. on p. 40).

[47] *Healthcare & Biotech*. ETRO-Vrije Universiteit Brussel (VUB). URL: https://www.etrovub.be/research/themes/hb/overview/ (visited on 09/15/2025) (cit. on p. 41).

[48] Alexandru Telea. «An Image Inpainting Technique Based on the Fast Marching Method». In: *Journal of Graphics Tools* 9 (Jan. 2004). DOI: 10.1080/10867651.2004.10487596 (cit. on pp. 42, 53, 54).

[49] Orbbec. *OrbbecSDK K4A Wrapper*. Version v1.10.3. URL: https://github.com/orbbec/OrbbecSDK-K4A-Wrapper (visited on 09/29/2025) (cit. on p. 56).